# รายงานสหกิจศึกษาฉบับสมบูรณ์

## การพัฒนาอุปกรณ์และการแก้ไขปัญหาของ IoT

## IoT Device & IoT Solution Development


นายจิรพัฒน์ ศิรสีห์

สาขาวิชาวิศวกรรมสารสนเทศ

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง


ปีการศึกษา 2562

| | |
|---|---|
| **ชื่อโครงงานสหกิจศึกษา** | พัฒนาอุปกรณ์และแก้ไขปัญหา IoT |
| **ชื่อ-สกุล นักศึกษา** | นายจิรพัฒน์ ศิรสีห์ |
| **ระดับปริญญา** | วิศวกรรมศาสตร์บัณฑิต |
| **สาขาวิชา** | วิศวกรรมสารสนเทศ |
| **ภาควิชา** | วิศวกรรมคอมพิวเตอร์ |
| **ชื่อ-สกุล อาจารย์นิเทศ** | ผศ.ดลชัย สุขเจริญผล |
| **ชื่อ-สกุล ผู้นิเทศงาน** | คุณสิริกันยา จงเจริญพรชัย |
| **สถานประกอบการ** | บริษัทแอดวานซ์ อินโฟร์ เซอร์วิส จำกัด (มหาชน) |

## บทคัดย่อ

โครงงานนี้เสนอการพัฒนาอุปกรณ์โมดูล ESP32 ให้ทำหน้าที่เป็นเกตเวย์ในการรับ–ส่งข้อมูลทางกายภาพที่ได้รับมาจากเซนเซอร์ของบริษัท Omron จากนั้นจึงนำไปแสดงบน IoT Platform ของทางบริษัท หรือ IoT Platform อื่น ๆ เพื่อให้ผู้ที่อยู่บริเวณรอบ ๆ หรือเอาไว้ใช้กับผู้สูงอายุ หรือโรงพยาบาล ได้ตระหนักถึงมลภาวะของสภาพแวดล้อมในบริเวณนั้น เช่น อุณหภูมิ ความชื้น ความสว่างของแสง ความดัน ความดังของเสียงรอบบริเวณนั้น เป็นต้น ช่วยให้ผู้ที่อยู่บริเวณโดยรอบ หรือนำเสนอข้อมูลไว้ใช้ทางการแพทย์กับผู้สูงอายุ หรือโรงพยาบาลสามารถแจ้งเตือน และป้องกันอันตรายที่อาจจะเกิดขึ้นจากสภาพแวดล้อมที่ไม่สามารถคาดการณ์ได้ โดย ESP32 มีข้อดีในการใช้งานที่มีพื้นที่จำกัด เหมาะกับงานที่ไม่ต้องใช้การประมวลผลสูง

| | |
|---|---|
| **Cooperative Title** | IoT Device & IoT Solution Development |
| **Student intern name** | Mr. Jirapat Sirasri |
| **Faculty** | Bachelor of Engineering |
| **Program** | Information Engineering |
| **Department** | Computer Engineering |
| **Advisor name** | Asst.Prof. Dolchai Sookcharoenphol |
| **Mentor name** | Ms. Sirikanya Jongjaroenpornchai |
| **Company** | Advanced Info Service Public Company Limited |

## ABSTRACT

This project present developed of ESP32 module device to act as a gateway for physical data transmitted from Omron company's multi type sensors and then displayed on the company's IoT Platform or also the other's IoT Platform.

All environment data from the sensors such as temperature, humidity, brightness of light, air pressure and sound noise of the surrounding area are provided awareness information of the pollution environment for a private medical care or hospital. The warning information of the hazard pollution is usefully applied for who take care the patient or elder.

The ESP32 has advantages in applications where space is limited, suitable for tasks that did not require high process.

# กิตติกรรมประกาศ

รายงานฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี ด้วยคำแนะนำ และคำปรึกษาจากหลาย ๆ ฝ่ายด้วยกัน โดยเฉพาะการให้คำปรึกษาเรื่องหัวข้องาน ความเอาใจใส่แนะนำแนวคิดต่าง ๆ ที่เป็นประโยชน์ต่อการทำงานนี้ให้สำเร็จผ่านลุล่วงไปได้ด้วยดี คือ คุณสิริกันยา จงเจริญพรชัย ตำแหน่ง Engineering Specialist ผู้ที่เป็นคนนิเทศงาน และรับผิดชอบตรวจสอบการทำงานทั้งหมด อีกทั้งฝ่ายดูแลรายละเอียดของงาน คอยให้ความรู้ในเชิงเทคนิค คือ คุณพิษณุพงษ์ มีประกอบ ตำแหน่ง Senior Programmer Analyst คุณสิรภัทร สิริปทุมรัตน์ ตำแหน่ง Engineer และพี่ ๆ ทุกท่านที่ให้ความรู้ ความเข้าใจในการทำงานครั้งนี้ คอยให้คำปรึกษา และแนวทางการแก้ไขปัญหาต่าง ๆ ซึ่งต้องขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณอาจารย์ที่ปรึกษา คณะวิศวกรรมศาสตร์ สาขาวิศวกรรมสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่คอยให้คำแนะนำในเรื่องการทำสหกิจศึกษาในครั้งนี้

สุดท้ายนี้ขอขอบคุณรุ่นพี่ และเพื่อน ๆ ทุกคนที่คอยให้คำปรึกษาในเรื่องการทำงาน การแก้ไขปัญหาต่าง ๆ ตั้งแต่เริ่มทำสหกิจศึกษาจนสำเร็จรายงานฉบับนี้ไปได้ด้วยดี

จิรพัฒน์ ศิรสีห์

# สารบัญ

หน้า

# สารบัญ (ต่อ)

หน้า

# สารบัญ (ต่อ)

หน้า

# สารบัญตาราง

หน้า

# สารบัญรูป

หน้า

# สารบัญรูป (ต่อ)

หน้า

# บทที่ 1
# บทนำ

## 1.1 ข้อมูลสถานประกอบการที่เข้าร่วมปฏิบัติงานสหกิจศึกษา

เริ่มแรกจดทะเบียนก่อตั้งเป็น บริษัท แอดวานซ์ อินโฟร์ เซอร์วิส จำกัด เมื่อวันที่ 24 เมษายน พ.ศ. 2529 เปิดให้บริการครั้งแรกเมื่อเดือนตุลาคม พ.ศ. 2533 โดยเอไอเอสทำสัญญากับองค์การโทรศัพท์แห่งประเทศไทย ให้ดำเนินการโครงการบริการระบบโทรศัพท์เคลื่อนที่ ais2100 เมกะเฮิรตซ์ เป็นระยะเวลา 30 ปี ถึง พ.ศ. 2561

เอไอเอสเข้าจดทะเบียนในตลาดหลักทรัพย์แห่งประเทศไทย เมื่อวันที่ 13 พฤศจิกายน พ.ศ. 2534 และแปรสภาพเป็นบริษัทมหาชนจำกัด หลังจากนั้นบริษัทขยายกิจการโดยการเข้าซื้อกิจการในเครือชินวัตร เช่น ชินวัตร ดาต้าคอม (ปัจจุบันคือ บริษัท แอดวานซ์ ดาต้าเน็ตเวิร์ค คอมมิวนิเคชั่นส์ จำกัด) , ชินวัตร เพจจิ้ง เป็นต้น บริษัทเปิดบริการโทรศัพท์เคลื่อนที่ในระบบจีเอสเอ็ม ในชื่อ Digital GSM ในเดือนตุลาคม พ.ศ. 2537 และได้ขยายเวลาร่วมสัญญาเป็น 25 ปี (หมดสัญญาปี พ.ศ. 2558) เมื่อ พ.ศ. 2539

ปัจจุบันเอไอเอสมีการแบ่งธุรกิจหลัก ๆ ออกเป็น 4 ประเภท ดังนี้

1. Mobile: ดูแลส่วนของสัญญาณโทรศัพท์ และ Super WIFI
2. Fixed Broadband: ดูแลส่วนของอินเทอร์เน็ตบ้านแบบออปติคไฟเบอร์
3. Digital Service: ดูแลในส่วนของ Digital Content ทั้งที่เป็น Mobile Application รวมถึง AIS Play Box เพื่อตอบโจทย์ทั้ง การทำงาน และความบันเทิง
4. Quality Service and Lifestyle: ดูแลในส่วนของการบริการลูกค้าให้ได้รับบริการที่มีคุณภาพที่สุด

## 1.2 ที่มา และความสำคัญของโครงงาน

เนื่องจากปัญหามลภาวะ และสภาพแวดล้อมที่เราใช้ชีวิตอยู่มีคุณภาพลดลงซึ่งอาจจะส่งผลต่อสุขภาพร่ายกาย และคนรอบข้าง โดยที่ไม่ได้ตระหนักถึงโดยสาเหตุมาจากการขาดข้อมูลและการเข้าถึงข้อมูลของสภาพแวดล้อมดังกล่าว ผู้จัดทำจึงได้เล็งเห็นถึงปัญหาดังกล่าว ทำให้มีการนำเซนเซอร์ตรวจวัดข้อมูลทางกายภาพของสภาพแวดล้อม เช่น อุณหภูมิ ความชื้น ความสว่างของแสง แรงกดดันอากาศ รวมทั้งความดังของเสียงรอบบริเวณนั้น นำมาแสดงให้เห็นถึงคุณภาพของสภาพแวดล้อมรอบตัวในขณะนั้น และเพื่อความสะดวกสบายในการเข้าถึง, การตรวจสอบ และวิเคราะห์ดูข้อมูลเหล่านี้ จึงได้มีการนำข้อมูลเหล่านี้ไปแสดงบน IoT Platform ของบริษัท หรือ IoT Platform อื่น ๆ เพื่อให้ผู้ที่อยู่บริเวณรอบ ๆ สามารถตรวจสอบ หรือจัดการกับข้อมูลเหล่านี้ได้อย่างสะดวกสบายมากยิ่งขึ้น

## 1.3 วัตถุประสงค์ของการปฏิบัติงาน

1. เพื่อให้ผู้ที่อาศัยอยู่โดยรอบรับรู้ถึงข้อมูล และปัญหามลภาวะของสภาพแวดล้อม
2. เพื่อให้ผู้ที่อาศัยอยู่โดยรอบสามารถเข้าถึง และวิเคราะห์คุณภาพถึงสภาพแวดล้อม
3. ลดการเกิดปัญหาสุขภาพที่จะส่งผลตามมาเนื่องจากมลภาวะสภาพแวดล้อม

## 1.4 ขอบเขตของโครงงาน

1. ศึกษาระบบการเชื่อมต่อไร้สายในรูปแบบของ Bluetooth Low Energy
2. ศึกษาการทำงานของ ESP32
3. ศึกษาระบบโครงข่าย NB-IoT
4. ศึกษาวิธีการส่งข้อมูลแบบ CoAP (Constrained Application Protocol)
5. ศึกษาวิธีการใช้งาน AT Command
6. พัฒนา Code Arduino เพื่อให้ ESP32 รับข้อมูลจากเซนเซอร์
7. พัฒนา Gateway เพื่อส่งข้อมูลไปยัง IoT Platform

## 1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. ผู้ที่อยู่อาศัยบริเวณโดยรอบสามารถรับรู้ได้ถึงปัญหาทางมลภาวะของสภาพแวดล้อม และสามารถเตรียมการแก้ไขปัญหาได้ตรงจุด
2. ผู้ประกอบการ หรือนักพัฒนาสามารถที่จะนำข้อมูลที่ได้นำมาคำนวณหรือทำการวิเคราะห์ต่อยอดเพื่อนำไปเข้าสู่กระบวนการต่าง ๆ

## 1.6 ขั้นตอนการดำเนินงาน

ในการดำเนินการจัดการจัดทำโครงการ จะประกอบด้วยขั้นตอน และระยะเวลาดังนี้

**ตารางที่ 1.1** ตารางแสดงขั้นตอนการดำเนินการ

| ตารางแสดงขั้นตอนการดำเนินงาน | | | | | | |
|---|---|---|---|---|---|---|
| ลำดับ | หัวข้องาน | ส.ค | ก.ย | ต.ค | พ.ย | ธ.ค |
| 1 | ศึกษาการเชื่อมต่อ BLE | → | | | | |
| 2 | ศึกษาและทดลองการติดต่อระหว่าง ESP32 กับ NB-IoT on board | | → | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 3 | ศึกษาและทดลองการรับ-ส่งข้อมูลเซนเซอร์กับ ESP32 ผ่าน BLE | | | → | → | |
| 4 | ส่งข้อมูลขึ้น IoT Platform | | | | | → |

## 1.7 อุปกรณ์ที่ใช้ในการพัฒนา

1.7.1 ฮาร์ดแวร์

1. DEVIO NB-Devkit 1
2. Omron sensor: 2jcie-bu01
3. Omron sensor: 2jcie-bl01
4. คอมพิวเตอร์

1.7.2 ซอฟต์แวร์

1. Arduino ide
2. Sublime
3. Docklight

1.7.3 ภาษาที่ใช้พัฒนา

1. Arduino
2. C++

# บทที่ 2
# แนวคิด ทฤษฎี และงานวิจัยที่เกี่ยวข้อง

## 2.1 Bluetooth Low Energy

Bluetooth คือ มาตรฐานการสื่อสารของอุปกรณ์อิเล็กทรอนิคแบบสองทางที่เชื่อมต่อในรูปแบบ ไร้สาย สำหรับการรับ-ส่งข้อมูลขนาดเล็กภายในข่ายงานส่วนบุคคล (Personal Area Network : PAN) โดยผ่านทางคลื่นวิทยุระยะสั้นความถี่สูง 2.4 GHz ระยะการทำงานของ Bluetooth จะอยู่ที่ 5-100 เมตร ซึ่งจะขึ้นอยู่กับ class ที่ใช้ ซึ่งแบบได้ดังนี้

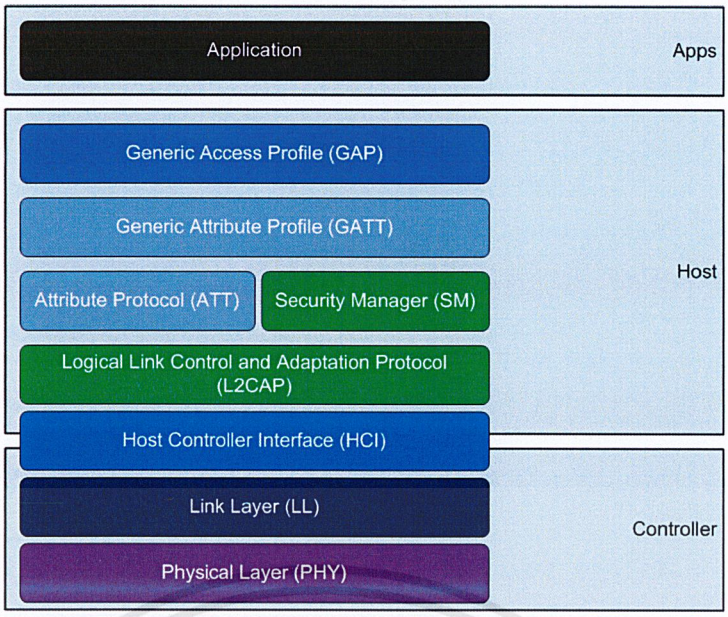
- Class 1 จะมีกำลังส่ง 100 มิลลิวัตต์ จะมีระยะในการส่งประมาณ 100 เมตร
- Class 2 จะมีกำลังส่ง 2.5 มิลลิวัตต์ จะมีระยะในการส่งประมาณ 10 เมตร
- Class 3 จะมีกำลังส่ง 1 มิลลิวัตต์ จะมีระยะในการส่งประมาณ 1 เมตร
- Class 4 จะมีกำลังส่ง 0.5 มิลลิวัตต์ จะมีระยะในการส่งประมาณ 0.5 เมตร

Bluetooth Low Energy (BLE บางแห่งจะเรียกว่า Bluetooth Smart) เป็นคุณลักษณะของเทคโนโลยีบลูทูธ 4.0 ได้ถูกออกแบบมาเพื่อให้ใช้งานสำหรับอุปกรณ์ไร้สายรุ่นใหม่ที่มีการใช้งานแบบประหยัด เช่น พลังงานต่ำ, แบนด์วิดธ์ต่ำ และความซับซ้อนต่ำ เริ่มต้นการพัฒนาจาก Wibree โดย Nokia ซึ่งปัจจุบันได้รับการดูแล การใช้งาน และกำหนดการสร้างมาตรฐานด้วย Bluetooth Special Interest Group (SIG)

Bluetooth Low Energy ได้ถูกออกแบบมาเพื่อให้เหมาะสมกับสถานการณ์ที่มีการใช้อัตราพลังงานต่ำ หรืออายุการใช้งานของแบตเตอรี่มีความสำคัญมากกว่าการที่ต้องใช้ความเร็วสูงในการแลกเปลี่ยนข้อมูล

### 2.1.1 Generic Access Profile (GAP)

เป็นเลเยอร์ที่อยู่ด้านบนสุดของ Host Protocol Stack ที่กำหนดสถานะของอุปกรณ์ที่จะทำการสื่อสารผ่าน BLE GAP จะทำการอธิบายการสร้างเส้นทางการสื่อสาร และขั้นตอนความปลอดภัยของแต่ละกระบวนการ GAP APIs จะสามารถเรียกใช้งานได้จากแอปพลิเคชันเลเยอร์




**รูปที่ 2.1** GAP Layer ที่มีความสัมพันธ์กับ Layer อื่นในซอฟต์แวร์ต่าง ๆ

| บทบาทของ GAP | คำอธิบาย |
|---|---|
| BROADCASTER | อุปกรณ์ที่ทำการส่ง advertising events |
| OBSERVER | อุปกรณ์ที่ทำการรับ advertising events |
| PERIPHERAL | อุปกรณ์ที่ยอมรับการสร้างใน LE physical link |
| CENTRAL | อุปกรณ์ที่รองรับหน้าที่ในการเป็นศูนย์กลางในการเชื่อมต่อทางกายภาพ |

**ตารางที่ 2.1** บทบาทต่าง ๆ ของ GAP

BLE สามารถสื่อสารกันได้ 2 รูปแบบ

1. Broadcasting & Observing เป็นรูปแบบการสื่อสารแบบทางเดียว ไม่มีการเชื่อมต่อระหว่างอุปกรณ์อย่างสมบูรณ์ เหมาะกับการสื่อสารที่มีข้อมูลขนาดเล็ก และต้องการความเร็วในการสื่อสารที่สูง
2. Connections เป็นรูปแบบการสื่อสารแบบสองทาง มีการสร้างเส้นทางในการสื่อสารระหว่างอุปกรณ์ที่ชัดเจน จะเหมาะกับการใช้ข้อมูลที่มีขนาดใหญ่กว่าในรูปแบบแรก

BLE จะใช้การส่งข้อมูลออกไปเป็นระยะจาก Broadcaster หรือ Peripheral ไปยัง Observer หรือ Central ซึ่งจะทำให้เกิดการประหยัดของพลังงาน รูปแบบการสื่อสารที่กล่าวมานี้เป็นรายละเอียดที่กำหนดไว้ในชื่อ Generic Access Profile (GAP)

### 2.1.2 Bluetooth Address

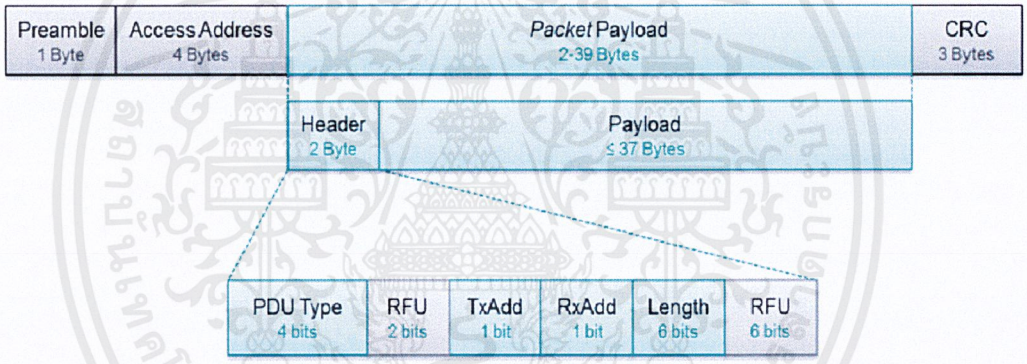
เป็นตัวบ่งบอก Address ที่ใช้ในการสื่อสาร คล้ายการสื่อสารในรูปแบบ internet จะแบ่งออกเป็น 2 รูปแบบ

1. Static Address (Public Address) มีขนาด 6 bytes (48bits) ซึ่งจะถูกกำหนดมาโดยผู้ผลิตอุปกรณ์ address นี้จะต้องไปจดขึ้นทะเบียนไว้กับ SIG และจะไม่มีการเปลี่ยนแปลงตลอดอายุของอุปกรณ์.
2. Random Address จะเป็น address ที่กำหนดได้ขณะใช้งานอุปกรณ์ ในทางปฏิบัติจะไม่ค่อยมีการใช้งาน

### 2.1.3 Data Packets

BLE จะมีโครงสร้างของข้อมูลแบบเดียว แต่จะมีการใช้งานที่สามารถแบ่งออกเป็นสองลักษณะ

1. Advertising packet
2. Scan Response packet




**รูปที่ 2.2** Data Packets BLE

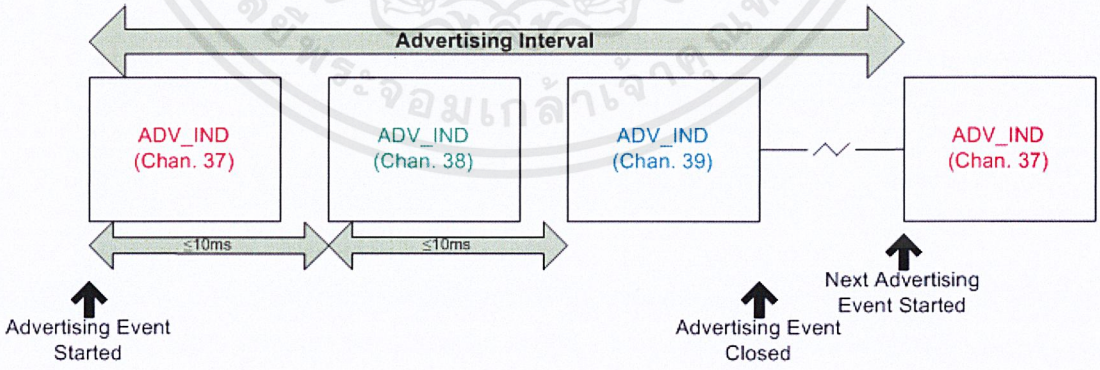
### 2.1.4 Advertising

คือการส่งข้อมูลออกของอุปกรณ์ (Broadcasting) ที่เป็น Peripheral หรือ Broadcaster (Advertiser) โดยมีวัตถุประสงค์เพื่อให้ค้นพบอุปกรณ์ (Device discovery) หรือเพื่อส่งข้อมูล (data Publishing) ด้วยระยะห่างของเวลาที่คงที่ ซึ่งจะมีค่าระหว่าง 20 Millisecond ถึง 10.24 Second Data packet ที่ถูกส่งออกไปจะถูกเรียกว่า Advertising Packet โดยจะมีขนาดสูงสุด 31 bytes โดยจะสามารถแบ่งออกได้ทั้งหมด 4 ประเภท ตามตารางต่อไปนี้

| Advertising PDU | คำอธิบาย | Max Adv Data Length | Max Scan Response Length | อนุญาตให้มีการเชื่อมต่อ |
|---|---|---|---|---|
| ADV_IND | ใช้สำหรับการส่งข้อมูลแบบกระจาย โดยสามารถเชื่อมต่อแบบไร้ทิศทางได้ | 31 bytes | 31 bytes | Yes |
| ADV_DIRECT_IND | ใช้สำหรับการส่งข้อมูลแบบกระจาย โดยสามารถเชื่อมต่อแบบมีทิศทางได้ | N/A | N/A | Yes |
| ADV_SCAN_IND | ใช้สำหรับการส่งข้อมูลแบบกระจายที่ได้จากการ Scan Request แบบไร้ทิศทาง | 31 bytes | 31 bytes | No |
| ADV_NONCONN_IND | ใช้สำหรับการส่งข้อมูลแบบกระจาย โดยที่ไม่มีการเชื่อต่อแบบไร้ทิศทาง | 31 bytes | N/A | No |

**ตารางที่ 2.2** Advertising packet ประเภทต่างๆ

การ Advertises จะใช้ channels อยู่ด้วยกัน 3 channels คือ 1. Channel 37 (2402 MHz) 2. Channels 38 (2426 MHz) 3. Channels 39 (2480 MHz) ซึ่งแต่ละ channel จะถูกเลือกมาจาก channel ที่จะส่งผลรบกวนกับ channel ของ Wi-Fi ให้น้อยที่สุด



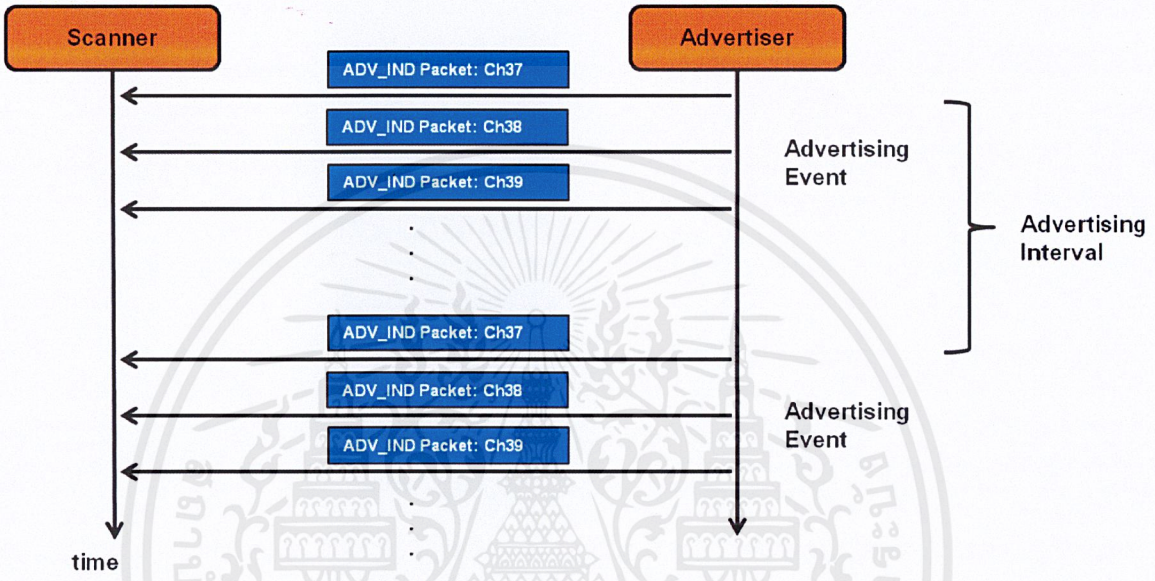

**รูปที่ 2.3** Channel ที่ถูกใช้งานในการ Advertise

### 2.1.5 Scanning

การสแกนเป็นหน้าที่สำคัญที่จะทำให้กระบวนการการค้นหาอุปกรณ์สำเร็จ ซึ่งการสแกนจะถูกแบ่งออกเป็น 2 ประเภท คือ Passive Scanning และ Active Scanning

#### 2.1.5.1 Passive Scanning

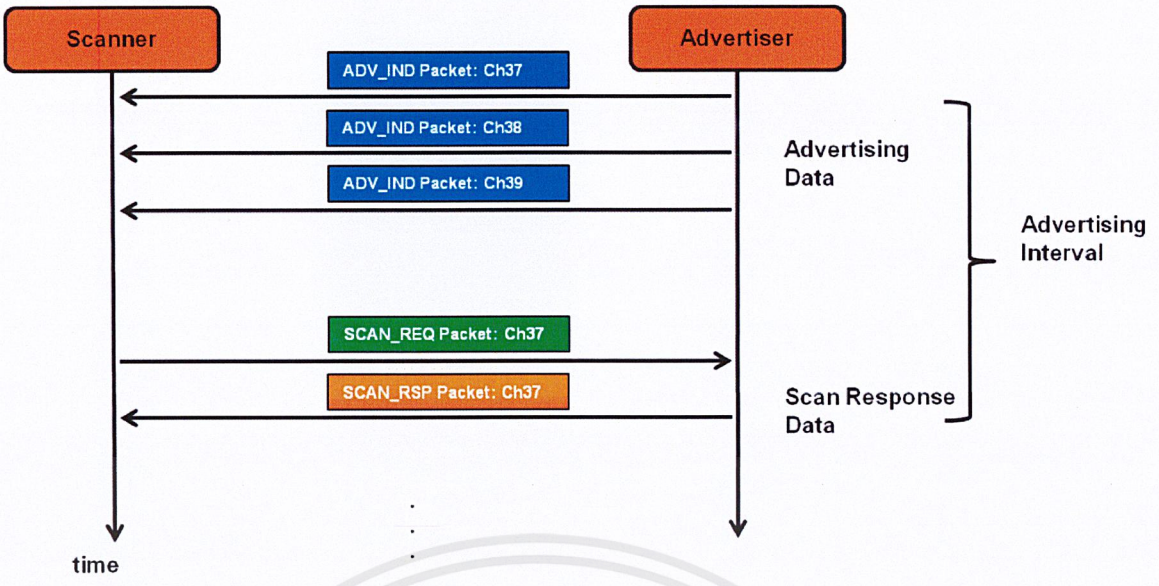
การสแกนประเภทนี้จะเป็นการสแกนหาอุปกรณ์ที่ทำการส่งข้อมูลในรูปแบบ Advertiser เท่านั้น




**รูปที่ 2.4** การสแกนในรูปแบบของ Passive

#### 2.1.5.2 Active Scanning

การสแกนประเภทนี้จะเป็นการที่อุปกรณ์ที่เป็น Central ต้องการข้อมูลเพิ่มเติมจาก Advertise packet โดย Scanner จะมีการส่ง Scan Request ไปยัง Advertiser ซึ่งทาง Advertiser จะมีการตอบกลับมาด้วย Scan Response packet ไปให้เพื่อให้ Scanner ได้ข้อมูลที่มากยิ่งขึ้น

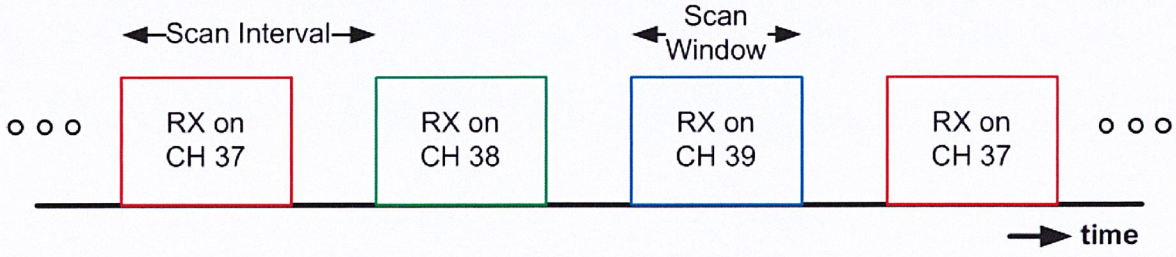



**รูปที่ 2.5** การสแกนในรูปแบบของ Active

ช่วงเวลาระหว่าง 2 packet ที่ติดต่อกันใน channel เดียวกัน จะเรียกว่า Time Inter Frame Space (ThIS) ซึ่งจะมีรายละเอียดตามตารางต่อไปนี้

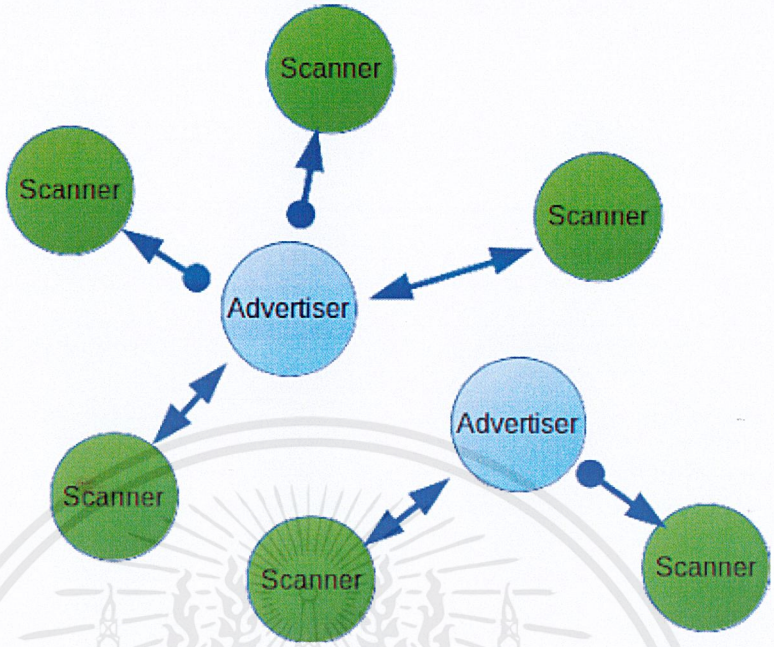
| Scan Parameter | คำอธิบาย | ระยะ |
|---|---|---|
| Scan Interval | ช่วงเวลาในการเริ่มต้นของ window scan ที่ต่อเนื่องกัน | 10ms – 10.24s |
| Scan Window | ระยะเวลาในชั้น Link layer ที่ทำการ scan ใน 1 channel | 10ms – 10.24s |
| Scan Duration | ระยะเวลาที่อุปกรณ์อยู่ในการดำเนินกระบวนการ scan | 10ms – infinity |

**ตารางที่ 2.3** รายละเอียดช่วงเวลาของกระบวนการ Scan




**รูปที่ 2.6** Diagram Parameter ของกระบวนการ scan

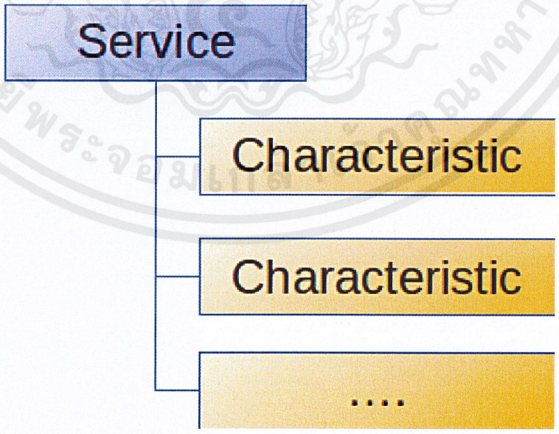
ในทางปฏิบัติแล้วในข่ายงานหนึ่ง ๆ สามารถมี Advertise และ Scanner ได้มากกว่า 1 ในช่วงเวลาเดียวกัน (Multiple Advertisers / Multiple Scanners)




**รูปที่ 2.7** การสื่อสารแบบ BLE ที่มากกว่า 1 ตัว

### 2.1.6 Attribute Protocol (ATT)

วิธีการสื่อสารของ Bluetooth คือรูปแบบเดียวกันกับ Client/Server ข้อมูลที่ใช้ในการแลกเปลี่ยนกันจะมีคุณลักษณะเป็นโครงสร้าง (จะคล้ายกับโครงสร้างของต้นไม้ ที่แยกกิ่งออกจากลำดับ)




**รูปที่ 2.8** โครงสร้างของ Attribute Protocol

Service คือ กลุ่มของข้อมูลที่มีลักษณะร่วมกันบางประการ เช่น ค่าที่ได้จากเซนเซอร์ซึ่งอาจจะมีมากกว่า 1 ค่า ทาง SIG ได้ทำการกำหนดลักษณะของ Service บางอย่างไว้แล้ว เพื่ออำนวยความสะดวกสำหรับผู้ผลิตเฟิร์มแวร์แต่ไม่ได้หมายความว่าจะเป็นการกำหนดตายตัว นักพัฒนาก็สามารถที่จะกำหนด Service สำหรับการใช้ในงานของตนเองได้ สำหรับ Characteristic คือที่ใช้จัดเก็บข้อมูลจริง ๆ เราอาจจะมองว่า Service เป็นตู้เก็บเอกสาร ส่วน Character จะถูกเปรียบเทียบได้เป็นลิ้นชักในการจัดเก็บเอกสาร ซึ่งเป็นข้อมูลเพื่อนำไปใช้งาน ในตู้เก็บเอกสารจะมีได้หลายลิ้นชัก แต่ละลิ้นชักจะมีเอกสารที่มีลักษณะเฉพาะที่แตกต่างกันไป

Attribute จะมีองค์ประกอบหลัก 3 องค์ประกอบ

1. Universal Unique ID (UUID) 2. Handle 3. Value

#### 2.1.6.1 Universal Unique ID (UUID)

เป็นข้อมูลที่มีความยาว 128 bits (16 Bytes) ใช้เพื่อแยก attribute จะไม่ใช้ซ้ำซ้อนกัน SIG ได้ทำการกำหนด UUID ไว้ล่วงหน้าสำหรับบาง Service โดยจะกำหนดความยาวไว้ 16 bit หรือ 32 bit เวลาใช้งานจะต้องนำมารวมกับ base UUID ซึ่งจะเป็นค่าคงที่ คือ 0000-1000-8000-00805F9B34FB โดยนำมาวางไว้ข้างหน้าของ base UUID ก็จะได้ความยาวเป็น 128 bit ตามมาตรฐาน การกำหนดไว้แบบนี้ก็เพื่ออำนวยความสะดวกแก่ผู้ผลิต Firmware ที่ไม่จำเป็นต้องใช้พื้นที่ยาวถึง 128 bit ในการเก็บข้อมูล เช่น กำหนดให้ service ที่ใช้งานมี UUID เป็น 0xaaa1 ในอุปกรณ์ที่ใช้งานก็จัดเก็บเฉพาะค่านี้ เมื่อส่งออกไป UUID นี้จะถูกตีความเป็น 0000aaa1-0000-1000-8000-00805F9B34FB โดยผู้รับจะนำเอา base UUID มาต่อท้ายให้

#### 2.1.6.2 Handle

เป็นข้อมูลความยาว 16 bits ใช้เป็นตัวระบุตัวตนสำหรับการอ้างอิงแต่ละ attribute ภายใน Connections Handle ที่ถูกสร้างขึ้นจะไม่มีการเปลี่ยนแปลงระหว่างการใช้งาน

#### 2.1.6.3 Value

ตัวข้อมูลจริง ไม่มีการกำหนดประเภท (none-type) แต่จะมีการกำหนดขนาดไว้ไม่เกิน 512 Bytes

### 2.1.7 Bluetooth Classic vs Bluetooth Low Energy

1.พลังงานที่ใช้งาน

สิ่งที่ทำให้ Bluetooth Low Energy (BLE) มีความพิเศษ คือ การกินพลังงาน BLE สามารถทำให้แบตเตอรี่ 1 เดือน สามารถใช้งานได้หลายเดือน หรือหลายปี ในขณะที่ Bluetooth classic จะให้ความเร็วของการแลกเปลี่ยนข้อมูลที่สูงกว่า ซึ่งนั้นจะทำให้แลกกับการที่ต้องใช้พลังงานที่มากยิ่งขึ้น

2.การใช้งาน

Bluetooth classic จะเหมาะสำหรับการใช้งานที่ต้องการความต่อเนื่องของข้อมูล เช่น การฟังเพลง ส่วน BLE จะเหมาะสำหรับการใช้งานที่ต้องการส่งข้อมูลเป็นระยะ จึงส่งผลให้ลดการกินพลังงาน ทำให้เหมาะสำหรับการใช้งาน IoT เป็นอย่างมาก

3.ความสามารถในการเชื่อมต่อพร้อมกัน

ความแตกต่างระหว่าง Bluetooth กับ Bluetooth Low Energy

| คุณสมบัติ | Bluetooth | Bluetooth Low Energy |
|---|---|---|
| ระยะทาง | < 30 เมตร | 50 เมตร (150 เมตร ในพื้นที่เปิด) |
| อัตราการส่งข้อมูล | 1-3 Mbps | 1 Mbps |
| การใช้พลังงาน | < 30mA | < 15mA |
| Latency | 100 ms | 6 ms |
| Application Throughput | 0.7–2.1 Mbps | 0.27 Mbps |
| Frequency Channels | 79 channels | 40 channels (3 advertising, 37 data channels) |
| ขนาดของข้อมูล (bytes) | สูงสุด 358 | 8 – 47 |
| ความปลอดภัย | 64/128 bits | 128 bits AES |
| การเชื่อมต่อ | 7 | ไม่จำกัด ขึ้นอยู่กับการออกแบบ |

**ตารางที่ 2.4** ความแตกต่างระหว่าง Bluetooth กับ BLE

## 2.2 Arduino

คือ โครงการที่ได้นำชิปไอซีไมโครคอนโทรลเลอร์ตระกูลต่าง ๆ มาใช้ร่วมกันในภาษา C ซึ่งภายในภาษา C นี้จะเป็นลักษณะที่เฉพาะ คือมีการเขียน Library ของ Arduino ขึ้นมาเพื่อให้การสั่งงานไมโครคอนโทรลเลอร์ที่แตกต่างกัน สามารถใช้งานได้ภายในโค้ดตัวเดียวกัน

### 2.2.1 ประวัติ Arduino

ในปี 2548 การสร้างผลงานของ Hernando Barragán (ผู้สร้าง Wiring), Massimo Banzi และ David Cuartielles ได้ทำการสร้าง Arduino ซึ่งเป็นอุปกรณ์ที่ใช้งานง่ายสำหรับโครงการที่มีการออกแบบเชิงโต้ตอบที่สถาบันออกแบบ Interrea Ivrea ใน Ivrea ประเทศอิตาลี David Mellis นักพัฒนาซอฟแวร์ Arduino ซึ่งมีการนำพื้นฐานมาจาก Wiring หลังจากนั้นไม่นานก็ได้มี Gianluca Martino และ Tom Igoe มาเข้าร่วมโครงการ และทั้งห้าคนก็ได้รู้จักในชื่อของผู้ก่อตั้งดั้งเดิมของ Arduino พวกเขาต้องการที่จะนำ

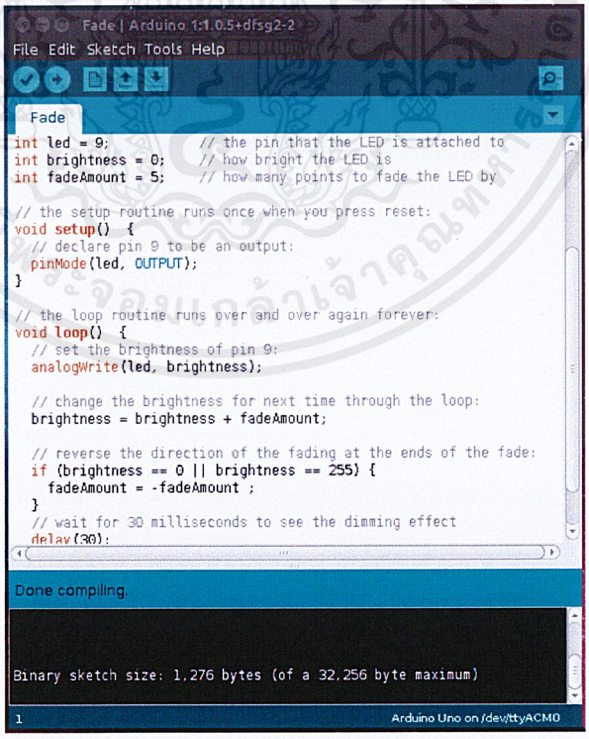
อุปกรณ์ที่เรียบง่าย และง่ายต่อการเชื่อมต่อกับสิ่งต่าง ๆ เช่น รีเลย์มอเตอร์ และเซนเซอร์ และสามารถนำไปตั้งโปรแกรมได้ง่าย นอกจากนี้จะต้องราคาไม่แพงเนื่องจากนักเรียนและผู้ชำนาญไม่มีเงินสดจำนวนมาก ทำให้พวกเขาเลือกตระกูล AVR ของอุปกรณ์ไมโครคอนโทรลเลอร์ 8 บิต (MCU หรือ µC) จาก Atmel และออกแบบแผงวงจรที่มีอยู่ในตัวเองพร้อมการเชื่อมต่อที่ใช้งานง่าย เขียนเฟิร์มแวร์ bootloader สำหรับไมโครคอนโทรลเลอร์และบรรจุทั้งหมดลงในสภาพแวดล้อมของการพัฒนาแบบรวมที่เรียบง่าย (IDE) ซึ่งโปรแกรมที่ใช้จะเรียกว่า “sketches” ทำให้ผลลัพธ์ที่ได้คือ Arduino

ตั้งแต่นั้นมา Arduino ได้เติบโตขึ้นในหลาย ๆ ทิศทาง บางรุ่นก็เล็กกว่าเดิม และบางตัวก็ใหญ่ขึ้น แต่ละคนมีความตั้งใจที่เฉพาะเจาะจงเพื่อเติมองค์ประกอบที่พบได้ทั่วไปคือ Library AVR-GCC ของ Arduino runtime ที่มาพร้อมสภาพแวดล้อมการพัฒนาของ Arduino และเฟิร์มแวร์ bootloader on-board ที่ติดตั้งมาพร้อมกับไมโครคอนโทรลเลอร์ของบอร์ด Arduino ทุกตัว

### 2.2.2 Arduino IDE

คือ เครื่องมือการเขียนโปรแกรมที่มีการใช้งานได้กับ Arduino ได้ทุกรุ่น โดยภายในจะมีเครื่องมือที่จะจำเป็นสำหรับการติดต่อกับ Arduino เช่น การค้นหา Arduino ที่ได้ทำการติดต่อกับคอมพิวเตอร์ การเลือกรุ่น Arduino ที่ติดต่ออยู่เพื่อตรวจสอบว่าขนาดของโปรแกรมที่เขียน หรือ Library ต่าง ๆ ที่ซับพอร์ตกับ Arduino รุ่นนั้น ๆ หรือไม่ อีกทั้งยังมีการติดต่อผ่านซีเรียลโดยตรงสำหรับคอมพิวเตอร์

จุดเด่นของโปรแกรม Arduino IDE คือเป็นโปรแกรมโอเพ่นซอส ทำให้สามารถนำไปใช้งานได้ฟรี ๆ อีกทั้งยังมีชุดคำสั่งตัวอย่างให้ทดสอบกับเนเซอร์ต่าง ๆ ได้ง่าย




**รูปที่ 2.9** หน้าโปรแกรม Arduino IDE

### 2.2.3 โครงสร้างภาษา C Arduino เบื้องต้น

โครงสร้างโปรแกรมภาษา C บน Arduino จะมีลักษณะแบบเดียวกับภาษา C ทั่ว ๆ ไป ซึ่งจะแบ่งออกเป็นส่วนต่าง ๆ ดังนี้

#### 2.2.3.1 ปรีโปรเซสเซอร์ไดเร็กทีฟ (Preprocessor directives)

คือ เป็นส่วนที่คอมไพลเลอร์จะมีการประมวลผลและทำตามคำสั่งก่อนที่จะมีการคอมไพล์โปรแกรมซึ่งจะเริ่มต้นด้วยเครื่องหมายไดเร็กทีฟ (Directive) หรือเครื่องหมายสี่เหลี่ยม (#) แล้วจึงตามด้วยชื่อคำสั่งที่ต้องการเรียกใช้หรือกำหนด โดยปกติส่วนนี้จะอยู่ในส่วนบนสุดหรือส่วนหัวของโปรแกรมและจะต้องอยู่นอกฟังก์ชั่นหลักใด ๆ ก็ตาม

#include จะเป็นคำสั่งที่ใช้อ้างอิงถึงไฟล์ภายนอก เพื่อเรียกใช้งานฟังก์ชั่นหรือตัวแปรที่มีการสร้าง หรือกำหนดไว้ในไฟล์นั้น ซึ่งรูปแบบการใช้งาน คือ #include <ชื่อไฟล์.h> เช่น #include<Wire.h>

#define เป็นคำสั่งที่ใช้ในการแทนข้อความที่กำหนดไว้ ด้วยข้อความที่กำหนดไว้ ซึ่งข้อดีคือจะไม่มีการอ้างอิงกับตัวโปรแกรม ซึ่งรูปแบบการใช้งาน คือ #define NAME VALUE เช่น #define LED 13

#### 2.2.3.2 ส่วนของการกำหนดค่า (Global Declarations)

คือ ส่วนที่ใช้ในการกำหนดชนิดตัวแปรแบบนอกฟังก์ชัน หรือประกาศฟังก์ชัน เพื่อให้ฟังก์ชันที่ประกาศสามารถกำหนด หรือเรียกใช้ได้จากทุกส่วนของโปรแกรม เช่น int pin = 13; , char point = A;

#### 2.2.3.3 ฟังก์ชั่น setup() และ ฟังก์ชั่น loop()

จะเป็นฟังก์ชั่นที่ถูกบังคับให้ต้องมีในทุกโปรแกรม โดยฟังก์ชัน setup() จะเป็นฟังก์ชันแรกที่ถูกเรียกใช้งาน จะนิยมใช้ในการกำหนดค่า หรือเริ่มต้นการใช้งาน Library ต่าง ๆ เช่น ในฟังก์ชัน setup() จะมีคำสั่ง pinMode() เพื่อกำหนดให้ขาใด ๆ ก็ตามที่เป็นดิจิทัลอินพุต หรือเอาต์พุต ส่วนฟังก์ชัน loop() จะเป็นฟังก์ชันที่ทำงานหลังจากฟังก์ชัน setup() ได้ทำงานเสร็จสิ้นไปแล้วและมีการวนรอบแบบไม่รู้จบ เมื่อฟังก์ชัน loop() ทำงานครบตามคำสั่งแล้วฟังก์ชัน loop() ก็จะถูกเรียกใช้งานขึ้นมาใหม่

ตัวอย่างของฟังก์ชั่น setup() และ loop()

```
Int pin = 13;
Void setup(){
        pinMode(pin, OUTPUT);
}
Void loop(){
        digitalWrite(pin, HIGH);
        delay(5000);
```

```
    digitalWrite(pin, LOW);
    delay(5000);;
}
```

จากตัวอย่าง จะเห็นว่ามีการประกาศตัวแปรแบบนอกฟังก์ชัน ทำให้สามารถกำหนด หรือจะเรียกใช้งานในฟังก์ชันใด ๆ ก็ได้ ซึ่งในฟังก์ชัน setup() ได้มีการกำหนดให้ขาที่ 13 เป็นดิจิทัลเอาต์พุต และในฟังก์ชัน loop() ก็ได้มีการกำหนดให้พอร์ต 13 มีลอจิกเป็น 1 (HIGH) และได้มีการใช้ฟังก์ชัน delay() ในการหน่วงเวลาใน 5 วินาทีแล้วจึงมีการกำหนดให้พอร์ต 13 มีลอจิกเป็น 0 (LOW) แล้วจึงหน่วงเวลา 5 วินาที จบฟังก์ชันการทำงานของ loop() และจะทำการเริ่มต้นการทำงานของฟังก์ชัน loop() ใหม่ ซึ่งผลลัพธ์ที่ได้คือจะมีไฟกระพริบบนบอร์ด Arduino ในพอร์ตที่ 13 ทำงานแบบไม่รู้จบ

ในทุก ๆ การทำงานของฟังก์ชันจะต้องมีการเริ่มด้วยการกำหนดค่าที่ส่งกลับ ตามด้วยชื่อของฟังก์ชันแล้วจะต้องตามด้วยเครื่องหมายปีกกาเปิด ({) และจะจบลงด้วยเครื่องหมายปีกกาปิด (}) ภายในฟังก์ชัน หากมีการเรียกใช้งานงานย่อยใด ๆ จะต้องมีเครื่องหมายเซมิโคลอน (;) ต่อท้ายเสมอ การกำหนดชนิดค่าที่จะส่งกลับเป็น void จะมีความหมายว่าไม่มีการส่งค่ากลับ แต่จะมีการใช้คำสั่ง return; ตรง ๆ ได้ เพื่อให้จบการทำงานของฟังก์ชันก่อนที่จะไปถึงบรรทัดสุดท้ายของฟังก์ชัน

#### 2.2.3.4 การสร้างฟังก์ชันและการใช้งานฟังก์ชัน (Users-defined function)

คำสั่งต่าง ๆ ที่อยู่ภายในฟังก์ชั่นจะต้องอยู่ภายใต้เครื่องหมายปีกกาเปิด และเครื่องหมายปีกกาปิด เท่านั้น ภายใต้เครื่องหมาย {} เราจะสามารถนำฟังก์ชันหรือคำสั่งใด ๆ ก็ได้มาใส่ไว้ แต่จะต้องคั่นด้วยเครื่องหมายเซมิโคลอน โดยจะนำคำสั่งทั้งหมดไว้บรรทัดเดียวกันเลย หรือแยกบรรทัดกันก็ได้เพื่อความสวยงามของชุดคำสั่ง (ไม่มีผลกับขนาดของโปรแกรมหลังคอมไพล์

#### 2.2.3.5 ส่วนอธิบายโปรแกรม (Program comments)

ส่วนอธิบายโปรแกรม หรือการแสดงความคิดเห็นของโปรแกรม จะเป็นส่วนที่สำคัญอย่างมากที่จะช่วยให้ผู้ที่ไม่ได้เขียนโปรแกรม หรือผู้เขียนโปรแกรมเข้าใจโปรแกรมได้ง่ายขึ้นโดยการอ่านข้อความจากส่วนแสดงความคิดเห็น **ส่วนนี้จะไม่มีผลใด ๆ กับขนาดของโปรแกรมหลังคอมไพล์ เนื่องจากจะถูกตัดทิ้งทั้งหมดเพราะไม่ได้ถูกนำไปใช้งาน จะมีผลแค่ไฟล์ชุดคำสั่งของโปรแกรมจะมีขนาดใหญ่ขึ้นมา** หากมีการแสดงความคิดเห็นของโปรแกรมเยอะ ๆ ขนาดไฟล์ก็จะเพิ่มขึ้นตามตัวอักษร ดังนั้นการแสดงความคิดเห็นของโปรแกรมจึงไม่คิดพื้นที่มากนัก แนะนำให้แสดงความคิดเห็นของโปรแกรมให้สั้น และกระชับเพื่อให้เกิดความรวดเร็วในการทำความเข้าใจ และจะต้องไม่ยาวจนต้องเลื่อนแถบเลื่อนหหน้าจอไปทางขวาเพื่ออ่านการแสดงความคิดเห็นของโปรแกรมเพิ่มเติม

การแสดงความคิดเห็นของโปรแกรมมีอยู่ 2 รูปแบบ

1. จะเปิดด้วย /* และจะทำการปิดด้วย */ ซึ่งเป็นการแสดงความคิดเห็นของโปรแกรมแบบข้ามบรรทัด คือ ถ้ายังไม่มี */ ตรงส่วนนั้นก็จะเป็นการแสดงความคิดเห็นของโปรแกรมทั้งหมด เช่น

```
/*
  Example comment
  Skip line comment
*/
```

2. เป็นการแสดงความคิดเห็นของโปรแกรมแบบบรรทัดเดียว คือ การเปิดด้วยเครื่องหมาย // และจะปิดด้วยการขึ้นบรรทัดใหม่ เช่น delay(1000); //เป็นการกำหนดการหน่วงเวลา 1 วินาที

### 2.2.4 ข้อแตกต่างระหว่าง Raspberry Pi (ราสเบอรี่ไพ) กับ Arduino

Raspberry Pi จัดว่าเป็นคอมพิวเตอร์ขนาดเล็กที่เรียกว่า Embedded Computer (เอมเบดเดด คอมพิวเตอร์) ที่สามารถต่ออุปกรณ์ได้เหมือนเครื่อง Micro Computer (ไมโคร คอมพิวเตอร์ ได้เลยเช่น ต่อออกจอภาพ ใช้เม้าส์, คีย์บอร์ด เป็นต้น และมีขา GPIO (จี พี ไอ โอ ซึ่งสามารถสั่งการอุปกร ณ์เหมือนขา I/O ของ Microcontroller (ไมโครคอนโทรลเลอร์ Embedded Computer สามารถประมวลผลได้โดยมีระบบปฏิบัติการ OS (โอ เอส แตกต่างกั บ Microcontroller ที่จะมีส่วนการจัดการในตัวเอง คือ bootloader (บูท โหลดเดอร์ ที่มีขนาดเล็กมากทำให้เราสามารถใช้งาน Micro-controller ที่ราคาไม่แพง ใช้ในการควบคุมที่มีการประมวลผลไม่ยุ่งยากซับซ้อนเท่ากับ Embedded Computer เช่น ใช้ให้ Arduino ทำการรับค่าเซ็นเซอร์จากแสงแดด LDR (แอล ดี อาร์ ว่ามีความเข้มเท่ากับที่เรากำหนดแล้วจึงสั่งให้ประมวลผลสั่งหลอดไฟให้ติดเป็นต้น

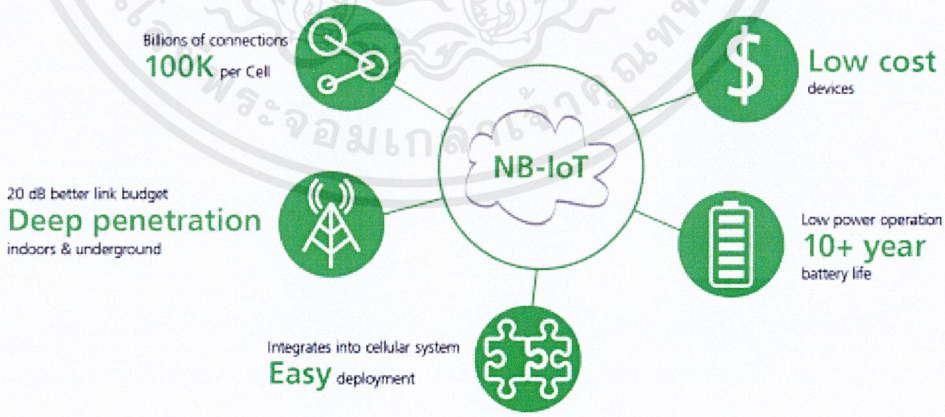
## 2.3 NB-IoT

ย่อมาจาก Narrowband IoT เป็นมาตรฐานระบบโครงข่ายที่ใช้พลังงานต่ำ (Low Power Wide Area Network (LPWAN)) ที่ถูกพัฒนามาเพื่อให้อุปกรณ์ต่าง ๆ สามารถเชื่อมต่อเข้าหากันได้ทุกที่โดยผ่านโครงข่ายของสัญญาณโทรศัพท์เคลื่อนที่ ด้วยความสามารถนี้ NB-IoT จึงถูกนำมาติดตั้งภายในพื้นที่ควบคุมหรือพื้นที่ที่ยากจะเข้าถึงในระยะไกลจากสถานีปล่อยสัญญาณโทรศัพท์ถัดไป หรือภายในพื้นที่หนาแน่นซึ่งสัญญาณนั้นยากที่จะทะลุผ่าน เช่น ในตึกสูงหรือใต้ดิน เมื่อมีสิ่งที่ป้องกันสัญญาณตามสถานการณ์ที่กล่าวมาข้างต้นแล้ว ความสามารถในการส่งสัญญาณก็จะต่ำลง และจำเป็นต้องพึ่งพาอุปกรณ์เครื่องส่งสัญญาณเพื่อทำงานร่วมกับระบบโครงข่ายที่ใช้พลังงานสูง และมีอัตราสิ้นเปลืองแบตเตอรี่สูง นอกเหนือจากนั้นเครือข่ายโทรศัพท์มือถือนั้นไม่สามารถนำไปใช้งานได้อย่างเหมาะสมสำหรับแอปพลิเคชันต่าง ๆ ที่สามารถส่งสัญญาณข้อมูลได้อย่างบางเบา และน้อยครั้ง และนอกเหนือจากนี้แล้ว มาตรฐานของเครือข่ายโทรศัพท์มือถือ Cellular ที่เราใช้อยู่ในปัจจุบันเช่น 3G, 4G หรือ 5G ในอนาคตนั้น ไม่สามารถรองรับอุปกรณ์ที่ประหยัดพลังงานได้ อย่างที่ NB-IoT สามารถทำได้ จึงทำให้มาตรฐาน cellular ดังกล่าวไม่เหมาะสมกับอุปกรณ์ที่มีราคาถูก เนื่องจากจำเป็นต้องมีแบตเตอรี่ที่มีอายุการใช้งานได้หลายปี

NB-IoT สามารถที่จะใช้เชื่อมต่ออินเตอร์เน็ตเข้ากับอุปกรณ์จำนวนมากได้ และทำให้แอปพลิเคชันใหม่ ๆ สามารถใช้งานได้ซึ่งจะเหมาะสมสำหรับการใช้งานแอปพลิเคชันที่ต้องใช้งานผ่านการสื่อสารด้วยสัญญาณข้อมูลขนาดเล็กในระยะเวลาที่นาน เนื่องจาก NB-IoT นั้นใช้งานผ่านย่านความถี่ที่ได้รับการอนุญาตแล้ว เราจึงมั่นใจได้ว่าเราจะได้รับทั้งความปลอดภัย และความน่าเชื่อถือ พร้อมกับได้รับประกันคุณภาพการบริการ การใช้งานคลื่นสัญญาณโทรศัพท์มือถือสำหรับแอปพลิเคชัน NB-IoT ทั่วไปจะมีราคาแพงเกินไป อีกทั้ง แอปพลิเคชัน NB-IoT ยังไม่สามารถใช้งานได้อย่างเต็มประสิทธิภาพ แอปพลิเคชั น NB-IoT จะเน้นไปที่การถ่ายโอนข้อมูลที่อัดแน่นด้วยความเร็วที่ช้า ซึ่งจะเหมาะสมกับระดับขีดความสามารถของอุปกรณ์ โดยอุปกรณ์ที่มีราคาถูกและมีอายุการใช้งานยาวนานนั้นจะสามารถตัดค่าใช้จ่ายการดำเนินงาน และลดความเสี่ยงต่อการถูกลักขโมยได้

### 2.3.1 คุณสมบัติหลักของ NB-IoT

1. มีอัตราสิ้นเปลืองของพลังงานที่ต่ำ (สามารถใช้ได้เกิน 10 ปี ต่อการชาร์จแบตหนึ่งครั้ง)
2. ขีดช่วงระยะของการส่งสัญญาณภายในตัวอาคาร และใต้ดินได้อย่างยอดเยี่ยม
3. สามารถติดตั้งเข้ากับอาคารสถานีปล่อยสัญญาณโทรศัพท์มือถือได้โดยง่าย
4. มีเครือข่ายที่ปลอดภัยและน่าเชื่อถือ
5. มีต้นทุนราคาชิ้นส่วนอุปกรณ์ที่ต่ำ



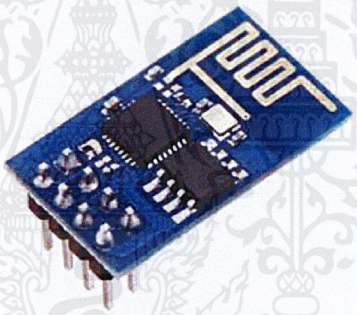

**รูปที่ 2.10** คุณสมบัติของ NB-IoT

## 2.4 ESP32

เป็นชื่อของไอซีไมโครคอนโทรคอนโทรลเลอร์ที่จะรองรับการเชื่อมต่อ WIFI และ Bluetooth 4.2 (BLE) ภายในตัว ผลิตโดยบริษี Espressif จากประเทศจีน รองรับการเขียนโปรแกรมโดยใช้โปรแกรม Arduino IDE และจะรองรับ Library ส่วนใหญ่ของ Arduino ทำให้สามารถใช้งานได้ง่าย

### 2.4.1 ประวัติความเป็นมาของ ESP32

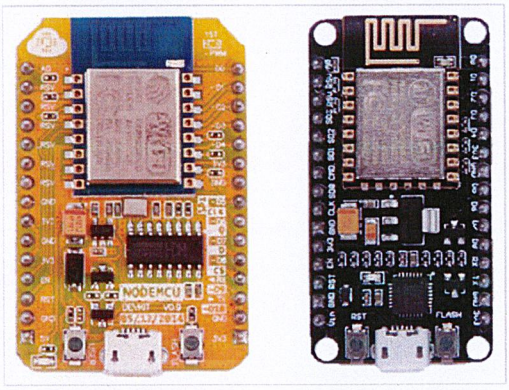
ก่อนที่ ESP32 จะถูกสร้างขึ้นได้มีไอซีไมโครคอนโทรลเลอร์ที่มี WIFI ในตัวและทำราคาที่ถูกมาก ๆ ในขณะนั้น (เพียง $5 หรือประมาณ 200 บาท) ออกมาปฏิวัติโลกของระบบสมองกลฝังตัว นั่นก็คือ ESP8266 ที่ผลิตโดยบริษัท Espressif จากประเทศจีน ในช่วงแรกนั้นไอซี ESP8266 สามารถทำงานได้โดยใช้การสื่อสารผ่าน UART เท่านั้น และจะพูดคุยสั่งงานผ่าน AT command ไม่สามารถทำการอัพเดทหรือแก้ไขเฟิร์มแวร์ด้านในได้ แต่ต่อมาไม่นานบริษัท Espressif ก็ได้ออกไอซีเวอร์ชั่นใหม่มา ซึ่งสามารถอัพเดทเฟิร์มแวร์ได้ และสามารถทำการเขียนเฟิร์มแวร์เองได้ โดยในขณะนั้นการที่จะทำการเขียนเฟิร์มแวร์จะใช้ได้แค่ภาษา C เท่านั้นและใช้ ESP8266 SDK เป็นชุดซอฟต์แวร์พัฒนา ซึ่งด้วยความยากของการใช้งานภาษา C เพียงอย่างเดียวจึงทำให้ไม่ได้รับความนิยมเรื่องการพัฒนาเฟิร์มแวร์เองมากนัก



**รูปที่ 2.11** โมดูล ESP8266 01 โดยบริษัท Ai-Thinker หัวใจหลักคือไอซี ESP8266

หลังจากนั้นมาประมาณ 1 ปี ผู้ผลิตบอร์ด NodeMCU ได้พอร์ตตัว Runtime ภาษา Lua มาลงใน ESP8266 ทำให้ตัวของ ESP8266 สามารถที่จะเขียนโปรแกรมสั่งงานตรง ๆ ได้ง่ายยิ่งขึ้น รวมทั้งมีความเสถียรภาพเพิ่มขึ้น และในขณะนี้เองบอร์ด NodeMCU จะเป็นบอร์ดที่พัฒนาของ ESP8266 สำเร็จรูปเพียงบอร์ดเดียวในตลาด ที่มาพร้อมกับ USB t0 UART ทำให้สามารถอัพโหลดเฟิร์มแวร์เข้า ESP8266 ได้ผ่าน USB โดยตรง นอกจากนี้ผู้พัฒนาบอร์ด NodeMCU ได้ทำการคิดค้นวงจรการเข้าโหมดอัพโหลดโปรแกรมอัตโนมัติ และตั้งชื่อว่า NodeMCU ซึ่งภายหลังบอร์ดพัฒนาทุกรุ่นจะใช้วงจรแบบ NodeMCU ในการเข้าโหมดอัพโหลดโปรแกรมอัตโนมัติ และด้วยเหตุผลที่บอร์ด NodeMCU เป็นบอร์ดพัฒนา ESP8266 บอร์ดแรกในท้องตลาดทำให้ได้รับความนิยมมาก และหลังจากบริษัทในจีนต่าง ๆ ได้ลอกวงจร และลายปริ้นของ

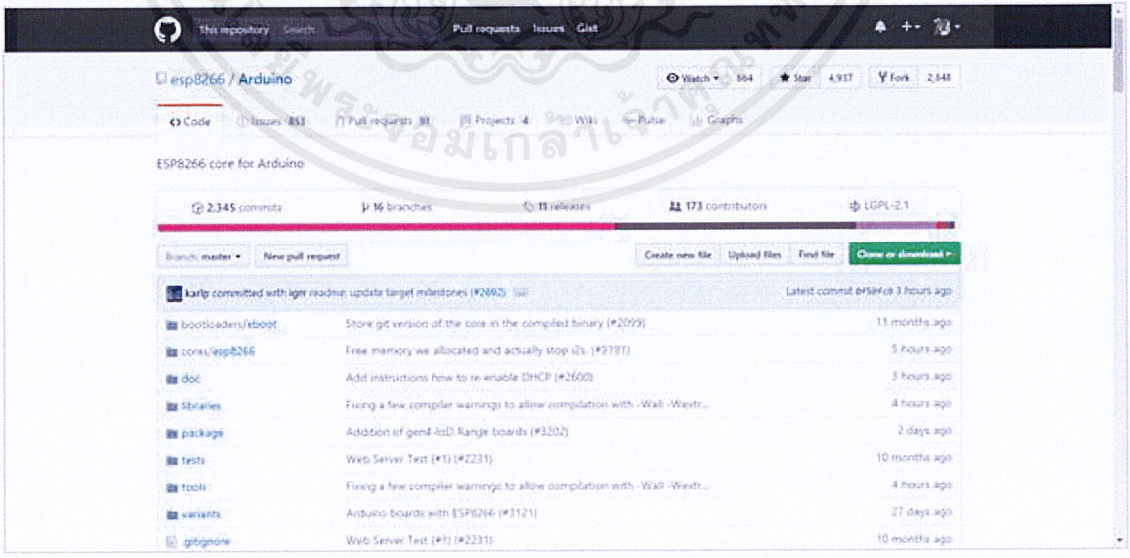
NodeMCU มาทำขายเองในราคาที่ถูกแล้วใช้ชื่อเดิมคือ NodeMCU จึงทำให้บอร์ด NodeMCU ได้รับความนิยมมากจนถึงปัจจุบัน



**รูปที่ 2.12** ด้านซ้ายเป็นบอร์ด NodeMCU 0.9 และด้านขาวเป็นบอร์ด NodeMCU 1.0

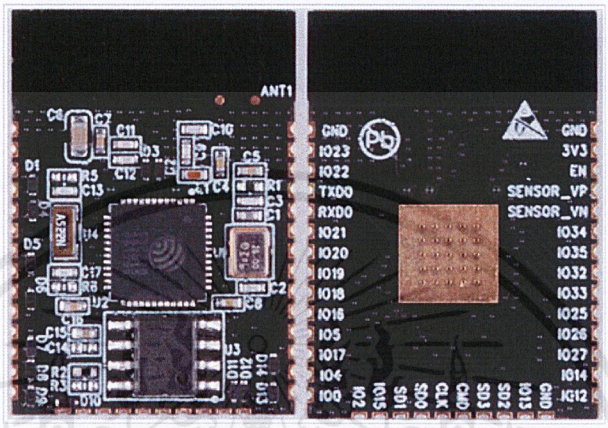
หลังจากตัว Runtime ของภาษา Lua ได้ทำการพอร์ตมาลง ESP8266 ได้ประมาณ 2 – 4 เดือน ทางชุมชนพัฒนา ESP8266 ที่ใช้ชื่อว่า ESP8266 Community Forum (www.esp8266.com) ได้ออกชุดLibrary และคอมไพล์เลอร์สำหรับใช้กับโปรแกรม Arduino IDE มาในชื่อ Arduino core for ESP8266

WIFI chip ทำให้การพัฒนาเฟิร์มแวร์ของ ESP8266 นั้นง่ายขึ้นมาก ๆ โดยใช้การเขียนโปรแกรมแบบ Arduino ดังนั้นคนที่มีพื้นฐานการเขียนโปรแกรมลงบอร์ด Arduino เป็นอยู่แล้วจึงมาเขียนเฟิร์มแวร์ลง ESP8266 โดยใช้โปรแกรม Arduino ได้ไม่ยาก และนอกจากนี้ Library ต่าง ๆ ที่ใช้งานได้กับบอร์ด Arduino ยังสามารถนำมาใช้งานกับ ESP8266 ได้เลยทำให้ ESP8266 ได้รับความนิยมสูงมากมาจนถึงขณะนี้



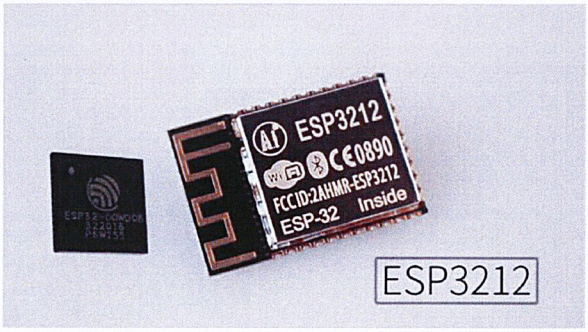

**รูปที่ 2.13** หน้าเว็บหลักของชุดแซฟแวร์ Arduino core for ESP8266 WIFI chip ใน GitHub

ด้วยความสำเร็จของไอซี ESP8266 ทำให้บริษัท Espressif ได้ทำการออกไอซีรุ่นถัดไป ซึ่งในช่วงแรกใช้ชื่อว่า ESP31B เปิดให้ร้านค้าใหญ่ ๆ อย่าง Adafruit SparkFun และผู้สนใจบางส่วนได้ทำการทดสอบ โดยในขณะนั้นได้มีการพัฒนาชุดซอฟแวร์ ESP32_RTOS_SDK ไปพร้อม ๆ กับการพัฒนาไอซี ESP31B ทำให้มีคนนำชุด ESP32_RTOS_SDK ไปพัฒนาลงโปรแกรม Arduino ก่อนไอซีตัวจริงจะออกในชื่อ Arduino core for ESP31B WIFI chip แต่หลังจากนั้นไม่นาน บริษัท Espressif ได้ยกเลิกการใช้ชุดซอฟแวร์พัฒนา ESP32_RTOS_SDK แล้วไปสร้างชุดพัฒนาใหม่ที่มีชื่อว่า ESP-IDF แทน (แต่เมื่อไปทำการเจาะลึกจะพบว่าภายในแทบจะลอก ESP32_RTOS_SDK มาทั้งหมด จากนั้นจึงออกไอซี ESP32 ออกมาเป็นครั้งแรก

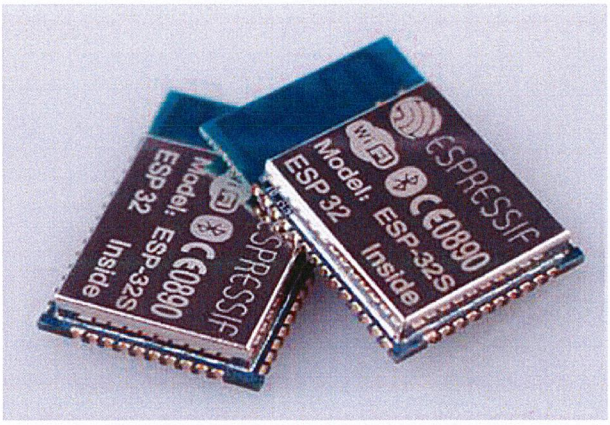



**รูปที่ 2.14** โมดูล ESP31B-WROOM-03 ที่ใช้ชิปไอซี ESP31B

ด้วยในอดีตที่ไอซี ESP8266 ได้ทำไว้ได้ดีมากจึงส่งผลให้ ESP32 ได้รับความสนใจอย่างมากจนผลิตไม่ทันต่อความต้องการ โดยในช่วงแรกบริษัท Espressif ได้ให้ข่าวว่าจะผลิต ESP32 แบบโมดูลออกมาเพียงอย่างเดียวในชื่อ ESP-WROOM-32 หลังจากนั้นไม่นานบริษัท Ai-Thinker ได้ร่วมมือกับ Seeedstudio ผลิตโมดูล ESP3212 ขึ้นมาโดยมีสถานะเป็นพรีออเดอร์ แต่เมื่อถึงกำหนดส่งมอบบริษัท Seeedstudio ได้เลื่อนการส่งมอบออกไปด้วยปัญหาด้านการออกแบบลายวงจรของตัวโมดูลเองทาง Ai-Thinker จึงได้ยกเลิกการผลิต ESP3212 แล้วหันไปผลิต ESP32S แทน โดยลายวงจรเหมือนกับ ESP-WROOM-32 ทุกประการแล้วจึงเริ่มส่งมอบสินค้าได้




**รูปที่ 2.15** โมดูล ESP3212 ที่ Ai-Thinker ได้ทำการร่วมมือกับ Seedstudio ในการผลิต




**รูปที่ 2.16** โมดูล ESP32S ที่ Seedstudio ทำการส่งมอบ

หลังจากสินค้า ESP32S ได้เริ่มส่งมอบทางทีมผู้พัฒนา Arduino core for ESP8266 WIFI chip ได้ถูกบริษัท Espressif ซื้อตัวมาทั้งหมดแล้วจ้างให้พัฒนาชุด Library และคอมไพล์เลอร์สำหรับ Arduino ในชื่อ Arduino core for ESP32 WIFI chip ทำให้การพัฒนาเป็นไปด้วยความรวดเร็วมากขึ้น ภายหลังผู้พัฒนา Arduino core for ESP31B WIFI chip ก็ถูกดึงตัวให้มาร่วมทีมพัฒนา Arduino core for ESP32 WIFI chip ด้วยเช่นเดียวกัน การพัฒนา Arduino core for ESP32 WIFI chip จะทำไปควบคู่กับการพัฒนา ESP-IDF โดยที่ ESP-IDF จะเป็นแกนหลัก เมื่อมีการเพิ่มฟีเจอร์ใหม่ ๆ ให้ ESP-IDF แล้วจึงจะมีการเพิ่มใน Arduino core for ESP32 WIFI chip โดยที่ ESP-IDF รองรับการเขียนโปรแกรมแบบ Arduino เช่นเดียวกัน และจะรองรับทุก Library ที่ใช้ได้สำหรับ Arduino เพียงแต่ ESP-IDF จะไม่มีโปรแกรม Editor โดยเฉพาะเท่านั้นเอง

### 2.4.2 คุณสมบัติของไอซี ESP32

- ใช้สถาปัตยกรรม Tensilica LX6 แบบ 2 แกนสมอง สัญญาณนาฬิกา 240 MHz
- แรมภายในตัว 512 KB
- รองรับการเชื่อมต่อรอมภายนอกสูงสุด 16 MB
- รองรับ WIFI มาตรฐาน 802.11 b/g/n และรองรับการใช้งานในโหมด Station softAP และ WIFI direct
- มีบลูทูธในตัว รองรับการใช้งานโหมด 2.0 และ โหมด 4.0 (BLE)
- ใช้แรงดันไฟฟ้าในการทำงาน 2.6V ถึง 3V
- ทำงานได้ที่อุณหภูมิ -40°C ถึง 125°C

- เซนเซอร์ต่าง ๆ ภายในตัว ESP32
    - วงจรกรองสัญญาณรบกวนภายในวงจรขยายสัญญาณ
    - เซนเซอร์แม่เหล็ก
    - เซนเซอร์สัมผัส (Capacitive touch) รองรับ 10 ช่อง
    - รองรับการเชื่อมต่อคลิสตอล 32.768 kHz สำหรับใช้กับวงจรนับเวลาโดยเฉพาะ
- ขาการใช้งานของ ESP32 ที่รองรับการเชื่อต่อบัสต่าง ๆ
    - GPIO จำนวน 32 ช่อง
    - UART จำนวน 2 ช่อง
    - SPI จำนวน 3 ช่อง
    - I2C จำนวน 2 ช่อง
    - ADC จำนวน 12 ช่อง
    - DAC จำนวน 2 ช่อง
    - I2S จำนวน 2 ช่อง
    - PWM/Timer ทุกช่อง
    - รองรับการเชื่อมต่อกับ SD-Card
- มีฟังก์ชันเกี่ยวกับความปลอดภัยต่าง ๆ
    - รองรับการเข้ารหัสแบบ WIFI แบบ WEP และ WPA/WPA2 PSK/Enterprise
    - มีวงจรเข้ารหัส AES / SHA2 / Elliptical Curve Cryptography / RSA-4096 ในตัว
- การรับ-ส่งข้อมูล
    - เชื่อมต่อแบบ 11n  HT40 จะได้ความเร็วสูงสุด 152 Mbps
    - เชื่อมต่อแบบ HT20 จะได้ความเร็วสูงสุด 72 Mbps
    - เชื่อมต่อแบบ 11g จะได้ความเร็วสูงสุด 54 Mbps
    - เชื่อมต่อแบบ 11b จะได้ความเร็วสูงสุด 11 Mbps
- การเชื่อมต่อผ่านโปรโตคอล UDP จะสามารถรับ-ส่งข้อมูลได้ที่ความเร็ว 135 Mbps
- ในโหมด Sleep จะใช้กระแสไฟฟ้าเพียง 2.5 uA

## 2.5 SIM7020

SIM7020 เป็นโซลูชันโมดูล NB-IoT แบบหลายแบนด์ในรูปแบบ SMT มีความสามารถในการขยายที่แข็งแกร่งพร้อมอินเทอร์เฟซที่หลากหลายรวมถึง UART, GPIO และอื่น ๆ ถูกออกแบบมาสำหรับแอปพลิเคชันที่ต้องการความหน่วงที่ต่ำ และการสื่อสารข้อมูลปริมาณน้อย ในสภาพการแพร่กระจายคลื่นวิทยุที่หลากหลาย เนื่องจากประสิทธิภาพความปลอดภัย และความยืดหยุ่นของโมดูลนี้จึงเหมาะอย่างมากสำหรับแอปพลิเคชัน M2M เช่น การวัดแสง การติดตามสินทรัพย์การตรวจสอบระยะไกล E-health ฯลฯ

### 2.5.1 คุณสมบัติของ SIM7020

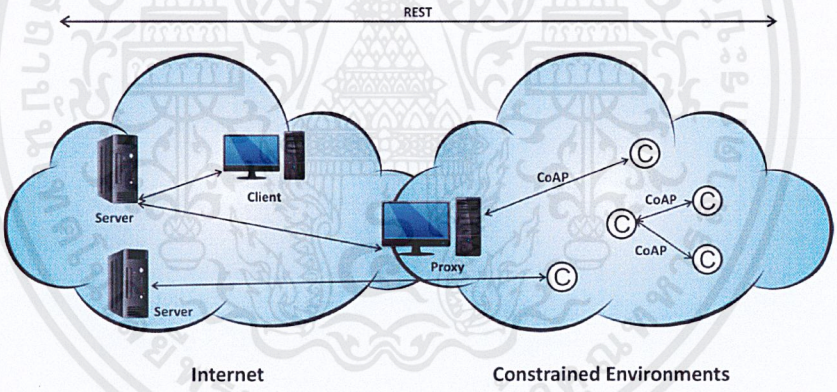
- ควบคุมการทำงานผ่าน AT Command
- คลื่นความถี่ของ SIM7020C: B1/B3/B5/B8
- คลื่นความถี่ของ SIM7020E: B1/B3/B5/B8/B20/B28
- แรงดันไฟฟ้า : 2.1V-3.6V
- อุณหภูมิการทำงาน: -40°C ถึง +85°C
- เฟิร์มแวร์การอัพเดทผ่าน UART
- การสื่อสาร: TCP/UDP, CoAP, MQTT
- การรับ-ส่งข้อมูล
    - ขาส่ง มีความเร็ว 62.5 Kbps
    - ขารับ มีความเร็ง 26.15 Kbps


**รูปที่ 2.17** SIM7020E

## 2.6 โปรโตคอลโคแอป

CoAP (Constrained Application Protocol) เป็นมาตรฐานที่ถูกพัฒนาขึ้นมา โดย IETF ในปี 2014 ซึ่งถูกออกแบบให้มีความคล้ายกับ HTTP ซึ่งเป็น Document transfer protocol แต่มีขนาดเล็กกว่ามาก (มี header แบบคงที่ขนาด 4 byte) เพราะตัดส่วนที่ไม่จำเป็นทิ้ง และรันบน UDP ซึ่งเป็น protocol ที่ไม่มีการสร้างการเชื่อมต่อระหว่างอุปกรณ์ปลายทาง จึงมีความเร็วในการส่งข้อมูลสูงแต่ไม่มีการรับประกันว่าข้อมูลที่ถูกส่งไปนั้นไปถึงปลายทางหรือไม่ และถูกต้องตามลำดับ การส่งข้อมูลซ้ำ และเรียงลำดับข้อมูลจะเกิดขึ้นในระดับแอปพลิเคชัน

โคแอปโปรโตคอลเป็นสถาปัตยกรรมแบบ Client/Server โดย client จะทำการร้องขอทรัพยากรไปยัง server โดยตรง จากนั้น server จะทำการตอบกลับพร้อมกับออพชัน Content-Type เพื่อบอก client ว่าข้อมูลที่จะได้รับนั้น เป็นข้อมูลรูปแบบไหน (เช่น JSON, XML, CBOR เป็นต้น โดย client สามารถ GET, PUT, POST และ DELETE ทรัพยากรบน Server ด้วย URL และ query string คล้ายกับ REST API ที่เราคุ้นเคย ในการสถาปัตยกรรมแบบโคแอปโปรโตคอลจะมีการแลกเปลี่ยนทรัพยากรกันโดยตรง เซนเซอร์โหนดทำหน้าที่เป็นทั้ง Server และ Client ในเวลาเดียวเพราะต้องทำการตอบรับแพ็คเกจที่ถูกส่งมา




**รูปที่ 2.18** สถาปัตยกรรมแบบ CoAP

โคแอปจะมีความคล้ายคลึงกับ HTTP ทำให้การดึงข้อมูลจากเซนเซอร์ไม่ต่างจากการดึงข้อมูลผ่าน Web API โดยจะเปรียบโคแอปได้ว่าเป็น REST API สำหรับ MCU นั่นเอง ทั้งยังเป็นโปรโตคอลที่มีความปลอดภัย เพราะมีการเข้ารหัสแบบ DTLS (เทียบเท่ากับ 3072-bit RSA key) ซึ่งสามารถรันบนอุปกรณ์ขนาดเล็กได้

โคแอปจะถูกออกแบบมาสำหรับการแลกเปลี่ยนข้อมูลแบบ 1 ต่อ 1 เหมาะสำหรับแอปพลิเคชันแบบกระจายศูนย์ ที่มีอุปกรณ์อยู่บนเครือข่ายเดียวกันสามารถติดต่อกันได้โดยตรง

การนำแต่ละโปรโตคอลใดไปใช้ขึ้นอยู่กับสถาปัตยกรรมของระบบ และข้อจำกัดทางด้านทรัพยากรทางด้านเครือข่าย จึงต้องเลือกเครื่องมือให้เหมาะสมกับงานในระบบที่มีความซับซ้อนมาก ๆ อาจจะประยุกต์ใช้พร้อมกันหลาย ๆ โปรโตคอลตามแต่ความเฉพาะเจาะจงของแอปพลิเคชันเพื่อให้บรรลุเป้าหมายในการทำงานได้อย่างมีประสิทธิภาพสูงสุด

## 2.7 รูปแบบ JSON

JSON ย่อมาจาก JavaScript Object Notation คือ รูปแบบของข้อมูลที่ใช้สำหรับแลกเปลี่ยนข้อมูลที่มีขนาดเล็ก ซึ่งสามารถทำความเข้าใจได้ง่าย และสามารถถูกสร้าง และอ่านโดยเครื่องได้ง่าย จะถูกกำหนดภายใต้ภาษา JavaScript (JavaScript Programming Language, Standard ECMA-262 3rd Edition – December 1999.) JSON จะเป็นรูปแบบข้อมูลตัวอักษรที่มีความเป็นอิสระ แต่จะมีหลักการการเขียนที่คุ้นเคยกับนักเขียนโปรแกรมภาษาต่าง ๆ ได้ ไม่ว่าจะเป็น ภาษา C, C++, C#, Java, JavaScript, Perl, Python และอื่น ๆ คุณสมบัติเหล่านี้จะทำให้ JSON เป็นภาษาแลกเปลี่ยนข้อมูลที่สมบูรณ์แบบ

ในการทำงานหลายอย่างกับ JavaScript เราจะพบว่า JSON เข้ามาเกี่ยวข้องด้วย ยกตัวอย่างการทำงานกับ script หลาย ๆ ตัวที่มีการเรียกข้อมูลแบบ AJAX ก็มักจะส่งข้อความตอบกลับมาในรูปแบบ JSON เนื่องจาก รูปแบบ JSON จะสามารถเข้าใจได้ง่าย และยังสามารถรับข้อมูลมาใช้งานได้ง่าย

มาตรฐานของฟอร์แมต JSON คือ RFC 4627 มี Internet media type เป็น application/json และมีนามสกุลของไฟล์เป็น .json

ปัจจุบัน JSON นิยมใช้ในเว็บแอปพลิเคชัน โดยเฉพาะ AJAX โดย JSON เป็นฟอร์แมตทางเลือกในการส่งข้อมูล นอกเหนือไปจาก XML ซึ่งนิยมใช้กันอยู่แต่เดิม สาเหตุที่ JSON เริ่มได้รับความนิยมเป็นเพราะกระชับและเข้าใจง่ายกว่า XML

### 2.7.1 JSON สามารถสร้างได้ 2 รูปแบบ

1. การจัดเก็บในชุดข้อมูลที่มีชื่อข้อมูล และข้อมูลคู่กันในภาษาต่าง ๆ ข้อมูลจะจัดอยู่ในรูปแบบของ Object, record, struct, dictionary, hash table, keyed list หรือ associative array

2. ลำดับของค่าข้อมูลในภาษาโปรแกรมส่วนใหญ่ จะจัดอยู่ในรูปแบบของ array, vector, list หรือ sequence

### 2.7.2 ประเภทของ JSON

1. Object คือ เป็นการเก็บชุดข้อมูลในรูปแบบ Key-Value โดยสามารถสังเกตว่าเป็นอ็อปเจกต์ที่ได้จากการที่ข้อความนั้นอยู่ภายใต้เครื่องหมาย { } ซึ่งข้อมูลที่อยู่ข้างในจะเป็นรูปแบบสตริงทั้งหมด { key1:value1 , key2:value2 } แบ่งแต่ละข้อมูลออกจากกันด้วยการใช้เครื่องหมาย comma (,) และแบ่ง Key กับ Value โดยใช้เครื่องหมาย colon (;) เป็นตัวแบ่ง

2. Array คือ ชุดข้อมูล ซึ่งจะเป็นชนิดใดก็ได้ โดยสามารถสังเกตว่าเป็นอาเรย์ได้จากการที่ข้อความนั้นอยู่ภายใต้เครื่องหมาย square bracket ([ ]) และแบ่งแต่ละข้อมูลออกจากกันด้วยการใช้เครื่องหมายcomma (,)

3. Value คือ String ที่อยู่ในเครื่องหมาย “ ” หรือตัวเลข หรือค่าทางตรรกศาสตร์ (true/false) หรือค่า null หรือ object หรือ array ซึ่งโครงสร้างสามารถวางซ้อนกันได้

4. String คือ ลำดับของตัวอักษรตั้งแต่ 0 ตัวอักษรหรือมากกว่า ซึ่งอยู่ภายใต้เครื่องหมาย “ ” และจะใช้เครื่องหมาย ในการใส่เครื่องหมายกำกับต่าง ๆ ซึ่งจะมีลักษณะคล้ายกับ String ในภาษา C หรือภาษา Java

5. Number คือ ตัวเลข

6. Null คือ ค่าว่าง

ช่องว่าง สามารถที่จะใส่ไว้ระหว่างสัญลักษณ์ต่าง ๆ ได้ ยกเว้นรายละเอียดซึ่งเข้ารหัสที่สมบูรณ์ในการบรรยายภาษาต่าง ๆ

### 2.7.3 โครงสร้างของ JSON

JSON นั้นใช้ลักษณะภาษาของ JavaScript แต่จะไม่ถูกมองว่าเป็นภาษาโปรแกรม กลับถูกมองว่าเป็นภาษาในการแลกเปลี่ยนข้อมูลมากกว่า ซึ่งในปัจจุบันมี Library ของภาษาโปรแกรมอื่น ๆ ที่ใช้ประมวลผลข้อมูลในรูปแบบ JSON มากมาย

#### 2.7.3.1 ตัวอย่างของ JSON

```
[
    {“ชื่อจริง” : “ชื่อ” , “ชื่อนามสกุล” : “นามสกุล”},
    {“ชื่อจริง_1” : “ชื่อ_1” , “ชื่อนามสกุล_1” : “นามสกุล_1”}
]
```

JSON นั้นยังสามารถจัดเก็บข้อมูลที่เป็น ลักษณะของ Master – Detail ได้อีกด้วย ตัวอย่างการจัดเก็บข้อมูล

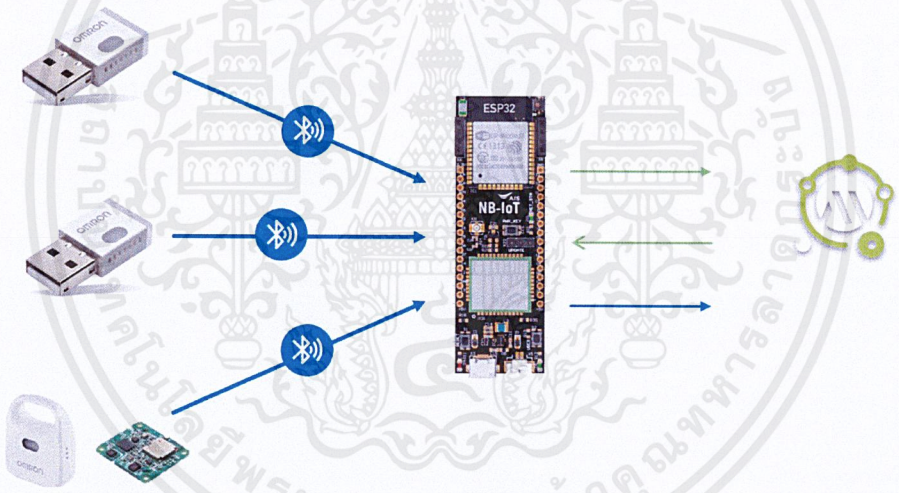
```
[
    {“ชื่อจริง” : “ชื่อ” ,
     “ชื่อนามสกุล” : “นามสกุล”,
     “ที่อยู่” : [
                {
                 “ที่อยู่_1” : “ที่อยู่”,
                 “จังหวัด” : “กรุงเทพ”,
                 “ประเทศ” : “ไทย”
                }
```

# บทที่ 3

# วิธีการดำเนินงาน

ในการดำเนินงานในโครงงานการพัฒนาอุปกรณ์ และการแก้ไขปัญหาของ IoT มีการนำองค์ประกอบความรู้ทางด้านฮาร์ดแวร์ และซอฟต์แวร์มาบูรณาการร่วมกัน เพื่อที่จะทำให้ได้ผลลัพธ์ตามต้องการ โดยโครงงานพัฒนาอุปกรณ์ และการแก้ไขปัญหา IoT เป็นการรับข้อมูลมลภาวะทางอากาศ ที่ได้มาจากเซนเซอร์ของทางบริษัท Omron แล้วนำข้อมูลที่ได้รับมา มาทำการคำนวณเพิ่มเติมเพื่อให้ได้ข้อมูลตามที่ต้องการ จากนั้นจะนำข้อมูลทั้งหมดที่ได้มาทำการรวบรวมและนำมาแสดงผลบน IoT Platform เพื่อให้สามารถติดตามข้อมูลต่าง ๆ ได้อย่างสะดวกสบาย

## 3.1 การออกแบบ




**รูปที่ 3.1** ภาพรวมของระบบ

จากรูปที่ 3.1 จะเป็นการแสดงภาพรวมของระบบซึ่งจะประกอบด้วย 1. เซนเซอร์ของทางบริษัท Omron 2. DEVIO NB-Devkit I (ESP32+SIM7020E) 3. IoT Platform โดยจะแบ่งขั้นตอนการทำงานออกได้ดังนี้

### 3.1.1 การรับข้อมูลเข้าสู่ระบบ

เริ่มจากตัวเซนเซอร์จะทำการรับข้อมูลทางกายภาพเข้ามาแล้วจะส่งผ่านข้อมูลที่ทำการรับมาด้วยการสื่อสารแบบไร้สายด้วย Bluetooth Low Energy เข้าสู่ DEVIO NB-Devkit I โดยตัวเซนเซอร์จะทำหน้าที่เป็นตัวกระจายข้อมูลออกมา และ DEVIO NB-Devkit I จะทำหน้าที่ในการดักจับข้อมูลที่ถูกส่งออก

มา เมื่อได้รับข้อมูลมาแล้ว DEVIO NB-Devkit I จะทำการนำข้อมูลที่ได้มาคำนวณเพื่อให้ได้ข้อมูลที่ต้องกรมากยิ่งขึ้น

### 3.1.2 การส่งข้อมูล

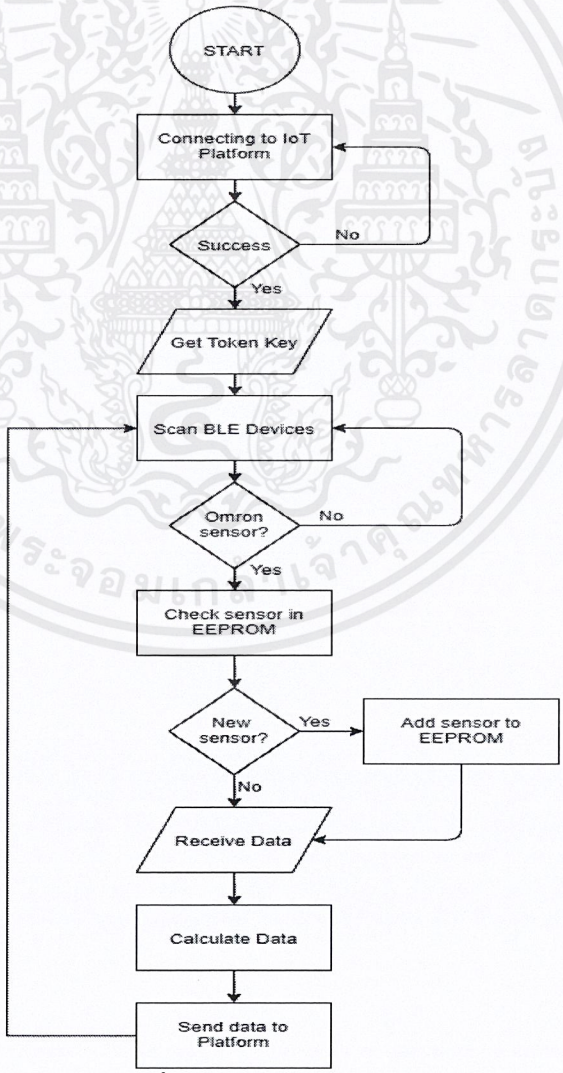
ตัว DEVIO NB-Devkit I จะทำการใช้ Library ของทาง AIS เพื่อทำการส่งข้อมูลไปยัง IoT Platform

### 3.1.3 การแสดงผลข้อมูล

ผู้ใช้งานจะสามารถติดตามข้อมูลได้จากหน้าของ Dashboard บน IoT Platform

### 3.1.4 การทำงาน

แผนภาพการทำงานของโครงงานการพัฒนาอุปกรณ์ และการแก้ไขปัญหา IoT ซึ่งจะอธิบายรายละเอียดการทำงานในแต่ละขั้นตอนดังนี้




**รูปที่ 3.2** Flowchart

จากรูปที่ 3.2

ขั้นตอนที่ 1. SIM7020E จะทำการเชื่อมต่อกับ IoT Platform ผ่าน Library ของทาง AIS ที่ใช้งานบน Arduino IDE ซึ่งจะมีขั้นตอนหลักด้วยกัน 5 ขั้นตอน ดังนี้

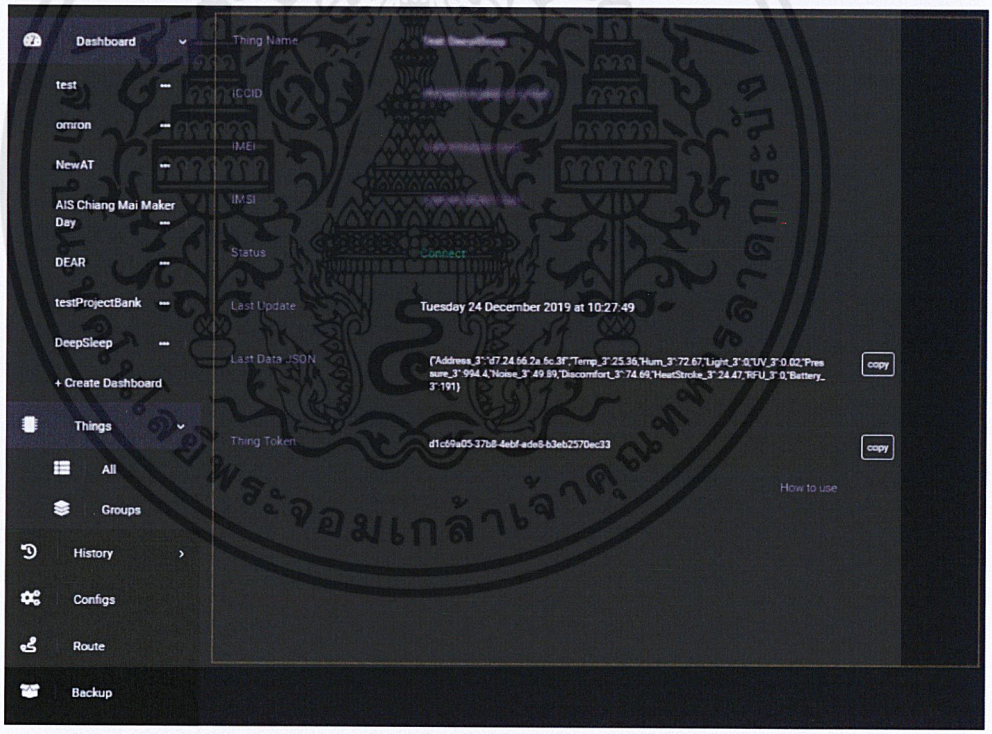
1. ตรวจสอบว่าพอร์ตถูกเปิดให้ใช้งาน

2. ทำการรีเซ็ทตัวโมดูล

3. ตรวจสอบสถานะของระบบเน็ตเวิร์ค

4. ทำการเชื่อมต่อกับ IoT Platform โดยการ register เข้าแพลตฟอร์มเพื่อตรวจสอบว่ามีการจดทะเบียนอุปกรณ์กับแพลตฟอร์มหรือยัง

5. เมื่อ register กับแพลตฟอร์มสำเร็จทางแพลตฟอร์มจะทำการส่ง Token key ซึ่ง Token key เปรียบเสมือนบัตรประชาชน เพื่อทำการส่งข้อมูลเข้าหา Thing ซึ่ง Thing จะทำหน้าที่เปรียบเสมือนกล่องเก็บข้อความที่รอรับข้อมูล




**รูปที่ 3.3** หน้าแสดงรายละเอียดของ Thing

ขั้นตอนที่ 2 ระบบจะทำการสแกนหาอุปกรณ์ที่ส่งข้อมูลในรูปแบบ Advertise ผ่าน Bluetooth Low Energy

```
********** Start BLE Scan **********
[D][BLEScan.cpp:153] setActiveScan(): set Active Scan
[D][FreeRTOS.cpp:189] take(): Semaphore taking: name: ScanEnd (0x3ffdf258), owner: <N/A> for start
[D][FreeRTOS.cpp:198] take(): Semaphore taken:  name: ScanEnd (0x3ffdf258), owner: start
[D][BLEAdvertisedDevice.cpp:435] setRSSI(): - setRSSI(): rssi: -61
[D][BLEAdvertisedDevice.cpp:246] parseAdvertisement(): Type: 0xff (), length: 29, data: 060001092002dd6fc749d46752ea87cf4c2alcd867755042f883c2037c
[D][BLEAdvertisedDevice.cpp:411] setManufacturerData(): - manufacturer data: 060001092002dd6fc749d46752ea87cf4c2alcd867755042f883c2037c
BLE Advertised Device found: Name: , Address: la:f2:25:87:82:04, manufacturer data: 060001092002dd6fc749d46752ea87cf4c2alcd867755042f883c2037c
RSSI : -61
[D][BLEAdvertisedDevice.cpp:435] setRSSI(): - setRSSI(): rssi: -66
[D][BLEAdvertisedDevice.cpp:246] parseAdvertisement(): Type: 0xff (), length: 29, data: 060001092002c3fd01f39ebae912156a7efe28fe90281db5f79be44ed7
[D][BLEAdvertisedDevice.cpp:411] setManufacturerData(): - manufacturer data: 060001092002c3fd01f39ebae912156a7efe28fe90281db5f79be44ed7
BLE Advertised Device found: Name: , Address: 55:75:47:6d:15:cf, manufacturer data: 060001092002c3fd01f39ebae912156a7efe28fe90281db5f79be44ed7
RSSI : -66
[D][BLEScan.cpp:98] handleGAPEvent(): Ignoring la:f2:25:87:82:04, already seen it.
[D][BLEScan.cpp:98] handleGAPEvent(): Ignoring 55:75:47:6d:15:cf, already seen it.
[D][BLEAdvertisedDevice.cpp:435] setRSSI(): - setRSSI(): rssi: -88
[D][BLEAdvertisedDevice.cpp:246] parseAdvertisement(): Type: 0xff (), length: 29, data: 060001092002f93f3b5e0425b6bec28a5f24dlad7f594c932c01b56771
[D][BLEAdvertisedDevice.cpp:411] setManufacturerData(): - manufacturer data: 060001092002f93f3b5e0425b6bec28a5f24dlad7f594c932c01b56771
BLE Advertised Device found: Name: , Address: 26:fd:e5:b9:81:01, manufacturer data: 060001092002f93f3b5e0425b6bec28a5f24dlad7f594c932c01b56771
RSSI : -88
```

**รูปที่ 3.4** แสดงกระบวนการสแกนหาอุปกรณ์ที่ส่งข้อมูลแบบ BLE

ขั้นตอนที่ 3 ถ้าอุปกรณ์ที่เจอไม่ใช่อุปกรณ์ของทางบริษัท Omron จะทำการสแกนหาใหม่จนกว่าจะเจออุปกรณ์ที่เป็นของทางบริษัท Omron

ขั้นตอนที่ 4 ถ้าเจออุปกรณ์ของทาง Omron จะทำการตรวจสอบว่าอุปกรณ์ที่เจอมีการเก็บ address ที่มีอยู่ใน EEPROM หรือไม่ ถ้ายังไม่มีให้ทำการเพิ่มเข้าไปใน EEPROM

```
BLE Advertised Device found: Name: EP, Address: d7:24:66:2a:6c:3f, manufacturer data: d50242a809011c78010300cf261b13c11c48090000bf
RSSI : -83
 ***** Device OMRON Found *****
Receive Data : d7:24:66:2a:6c:3f
Loop check eeprom
Address in Array 2 = d7:24:66:2a:6c:3f
Address 3 = d7:24:66:2a:6c:3f
[I][test_single_ble.ino:346] loop(): free heap ammount: 129588
[I][test_single_ble.ino:347] loop(): free heap minimum ammount: 124704

Address 1 : c8:cd:92:2a:6f:9b
Address 2 : dc:dl:30:96:82:33
Address 3 : d7:24:66:2a:6c:3f
Address 4 :
Address 5 :
Address 6 :
Address 7 :
Address 8 :
Address 9 :
Address 10 :
```

**รูปที่ 3.5** การตรวจสอบว่าอุปกรณ์มี หรือไม่มีใน EEPROM

ขั้นตอนที่ 5 รับข้อมูลต่าง ๆ ที่ได้มาจาก Advertise packet ของเซนเซอร์

```
----- Calculator Ready -----
d7:24:66:2a:6c:3f
d50242a8090110c78010300cf2610b13c110c48090000bf
 - BAG or EP TYPE
Temp : 24.72
Hum : 71.69
Light : 376
UV : 0.03
Pressure : 993.50
Sound Noise : 48.91
Discomfort : 73.61
Heat stroke : 23.76
RFU : 0.00
Battery Voltage : 191
```

**รูปที่ 3.6** แสดงข้อมูลต่าง ๆ ที่ได้รับมาจากทางเซนเซอร์ของบริษัท Omron

ขั้นตอนที่ 6 ส่งข้อมูลที่ได้ไปยัง IoT Platform โดยโมดูล SIM7020E จะทำการนำข้อมูลที่ได้จาก ESP32 จัดให้อยู่ในรูปแบบของ JSON จากนั้นจะทำการส่งขึ้นไปยัง IoT Platform ด้วยวิธีการ CoAP Protocol ข้อมูลที่ถูกส่งจะเข้าไปสู่ Things เพื่อนำไปใช้แสดงค่าในหน้าของ Dashboard


**รูปที่ 3.7** แสดงข้อมูลที่ถูกส่งขึ้นมาภายใน Things

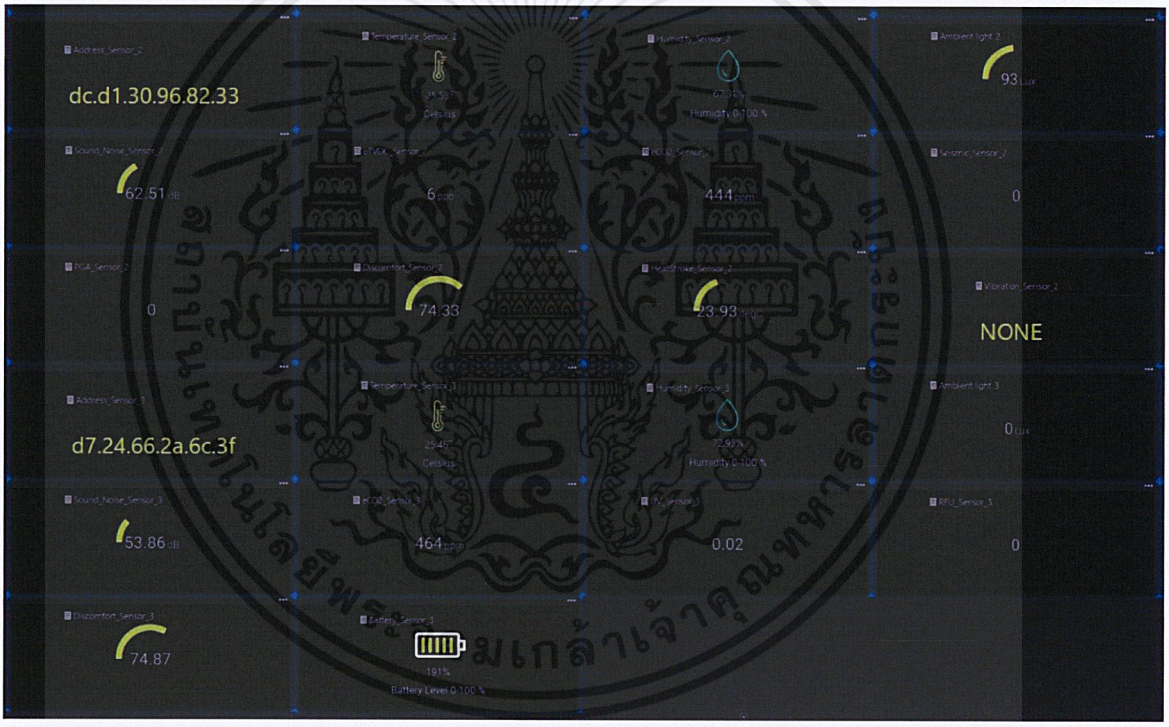
# บทที่ 4

# ผลการดำเนินงาน

ผลของการไปปฏิบัติสหกิจศึกษา ณ บริษัท แอดวานซ์ อินโฟร์ เซอร์วิส จำกัด (มหาชน) ผู้จัดทำได้รับมอบหมายให้พัฒนาอุปกรณ์ และแก้ไขปัญหา IoT โดยได้พัฒนาในส่วนของ ESP32 ให้สามารถนำข้อมูลที่ได้รับมาจากเซนเซอร์ ส่งไปยัง IoT Platform ผ่าน SIM7020E

ในบทนี้จะเป็นการแสดงให้เห็นถึงส่วนแสดงผลบน Magellan ซึ่งเป็น IoT Platform ที่ผู้จัดทำได้กล่าวถึง

## 4.1 ส่วนแสดงผลรวมของ Magellan

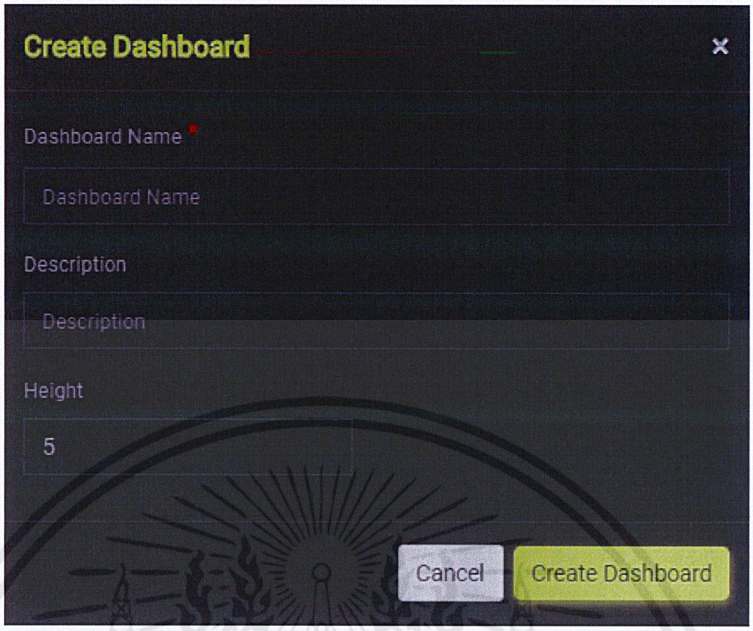



**รูปที่ 4.1** ตัวอย่างหน้าแสดงผลรวม (Dashboard)

จากภาพที่ 4.1 จะเป็นตัวอย่างหน้าแสดงผลรวม โดยเริ่มต้นจะเป็นหน้าว่างเปล่า ผู้ใช้จะจำเป็นที่จะต้องมีข้อมูลมาก่อน แล้วจะถึงสร้างวิทเจ็ท หรือ ชุดคำสั่งโปรแกรมคอมพิวเตอร์ขนาดเล็กสำหรับการแสดงผลข้อมูล เพื่อแสดงผลข้อมูลที่ได้รับมาตามความต้องการของผู้ใช้ โดยที่ผู้ใช้สามารถเลือกข้อมูลของแต่ละ Things เพื่อแสดงผลบน dashboard เดียวกันได้

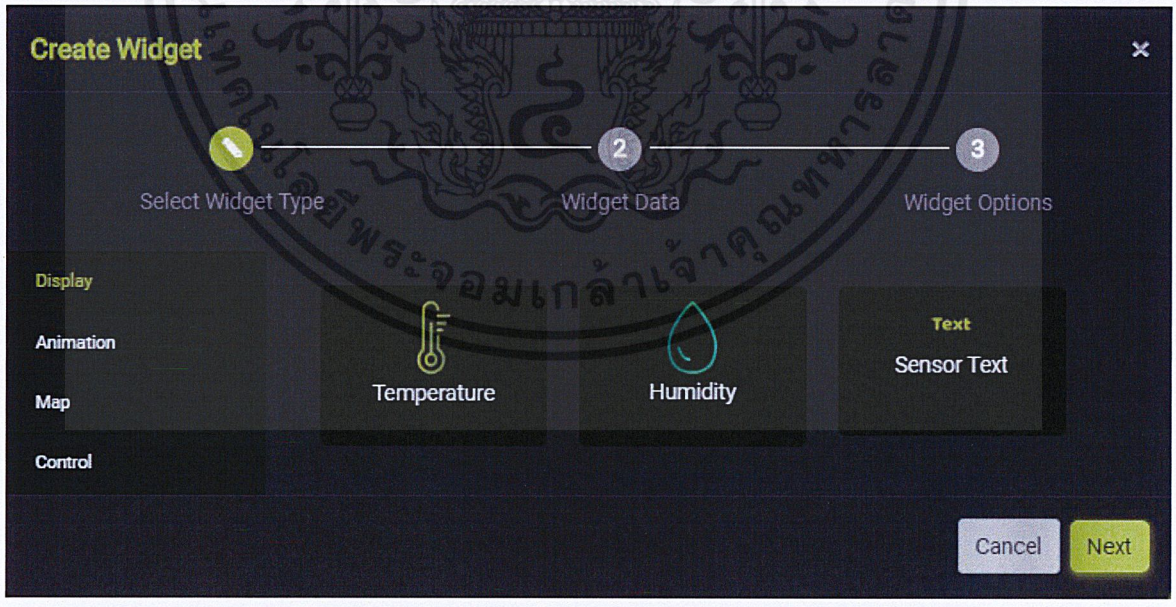
## 4.2 ขั้นตอนการสร้าง Dashboard

ขั้นตอนที่ 1 สร้าง Dashboard

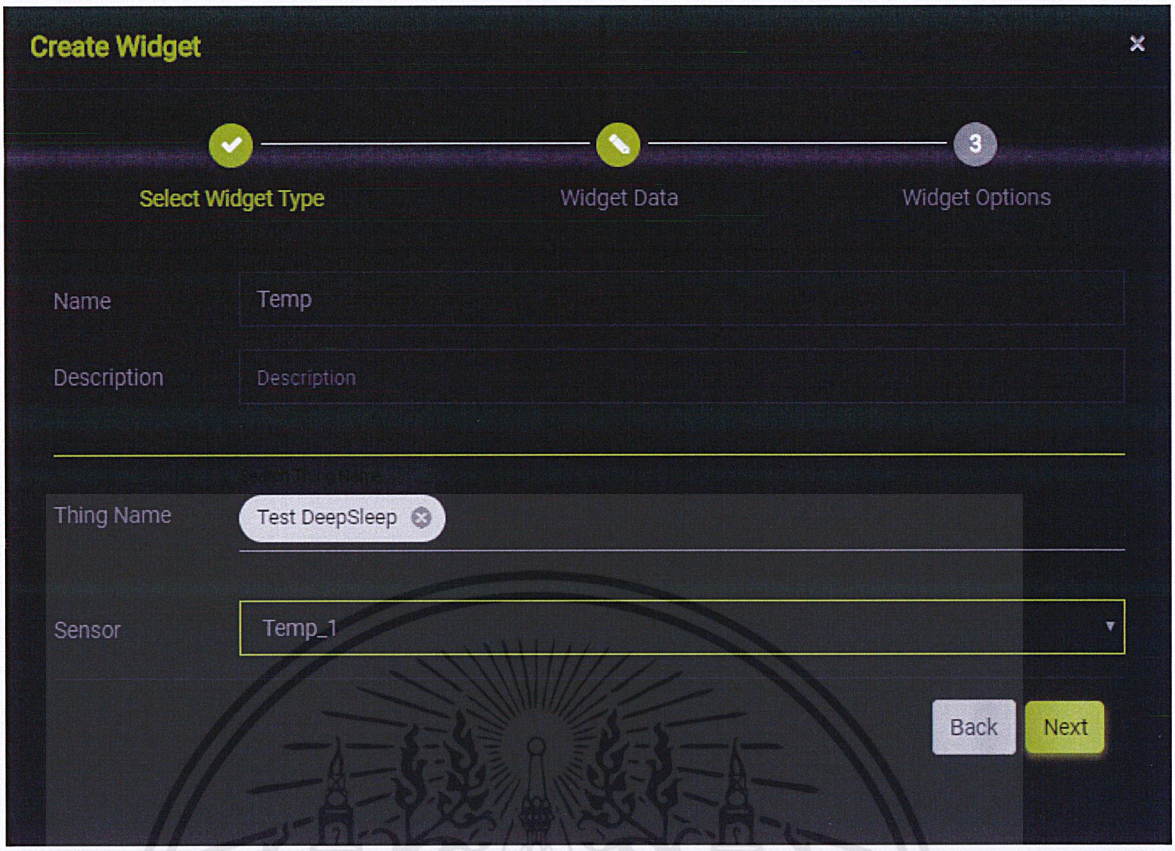



**รูปที่ 4.2** การสร้าง dashboard

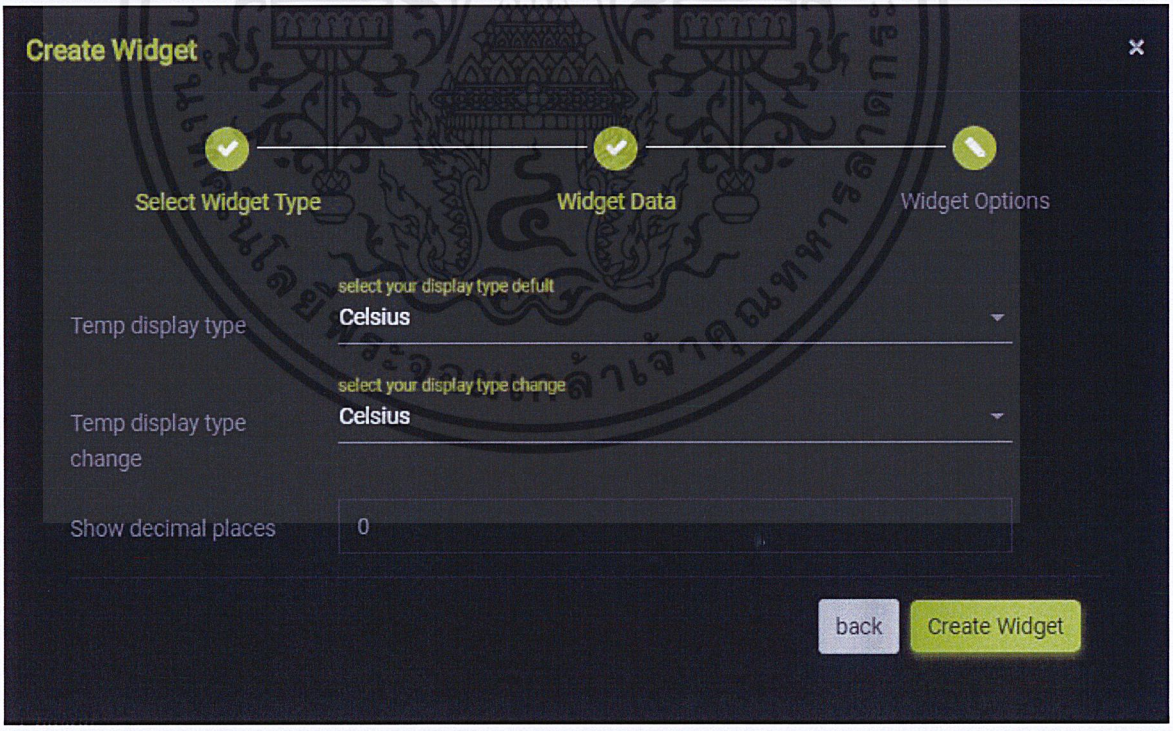
ขั้นตอนที่ 2 เมื่อทำการสร้าง Dashboard เรียบร้อย ก็จะมาสร้างวิทเจ็ทซึ่งจะมีวิทเจ็ทให้เลือกในรูปแบบต่าง ๆ



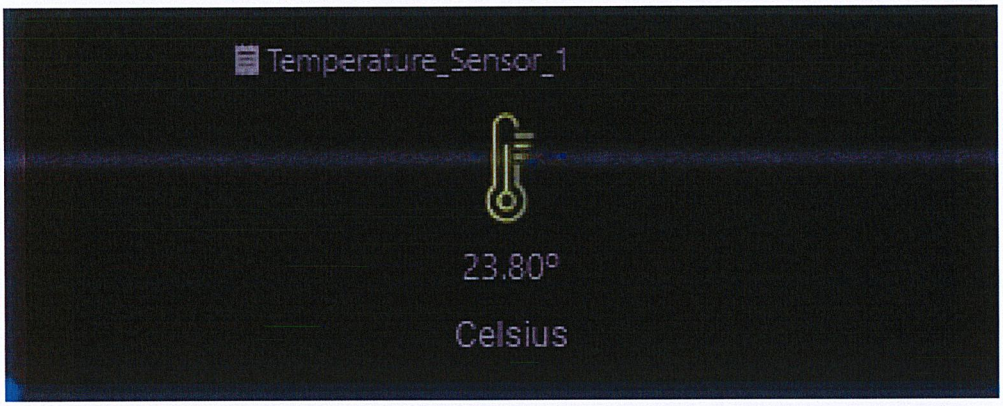

**รูปที่ 4.3** การสร้างวิทเจ็ท เลือกรูปแบบการแสดง




**รูปที่ 4.4** การใส่ตั้งชื่อ และดึงข้อมูลให้วิทเจ็ท




**รูปที่ 4.5** การตั้งค่าเพิ่มเติมให้วิทเจ็ท

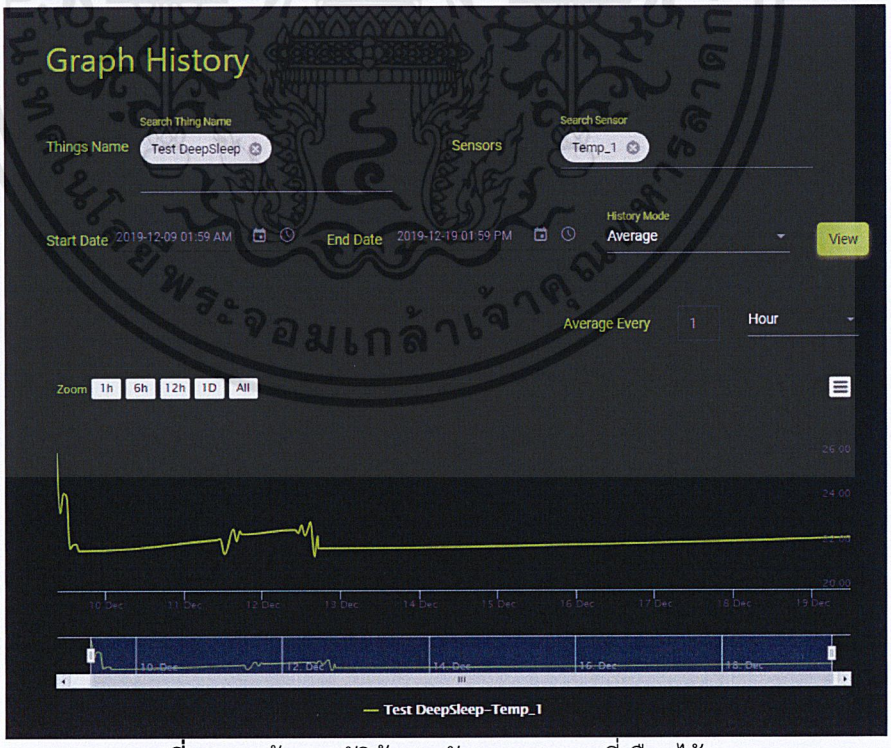



**รูปที่ 4.6** แสดงวิทเจ็ทในหน้า dashboard

ในส่วนของการแสดงผลบน dashboard ใน Magellan Platform ผู้ใช้สามารถเลือกข้อมูลจาก Thing ที่ต่างกัน มาแสดงผลบน dashboard เดียวกันได้

## 4.3 ประวัติย้อนหลังของข้อมูล

ใน Magellan Platform ผู้ใช้สามารถดูประวัติการรับข้อมูลของ Thing ได้ ซึ่งจะสามารถดูในรูปแบบกราฟ ในช่วงเวลาที่ต้องการได้




**รูปที่ 4.7** หน้าประวัติย้อนหลังของ Thing ที่เลือกไว้

# บทที่ 5

# สรุปผลการดำเนินงาน และข้อเสนอแนะ

ตลอดในระยะเวลาการดำเนินงานการวิเคราะห์รูปแบบการทำงานไปจนถึงการปรับปรุงกระบวนการต่าง ๆ จนกระทั่งสิ้นสุดโครงการสหกิจศึกษา สามารถสรุปผลการดำเนินงาน ปัญหา และอุปสรรค และข้อเสนอแนะแนวทางในอนาคตได้ดังต่อไปนี้

## 5.1 สรุปผลการดำเนินงาน

จากการดำเนินงานพัฒนาอุปกรณ์ และแก้ไขปัญหา IoT ส่งผลให้ผู้ใช้สามารถรับรู้ข้อมูลที่ได้รับมาจากตัวเซนเซอร์ผ่าน Magellan Platform หรือสามารถนำข้อมูลเหล่านั้นไปแสดงผลในรูปแบบอื่น ๆ ได้ ทำให้ผู้ที่ได้รับรู้ถึงข้อมูลเหล่านี้จะตระหนักได้ถึงมลภาวะที่อาจจะเกิดขึ้นโดยที่ไม่ได้ทันสังเกต

ซึ่งหลังจากได้พัฒนาโครงการจนจบ อาจจะนำโครงงานนี้ไปใช้ในสภานที่ที่มีความเสี่ยงในด้านของมลภาวะสภาพแวดล้อม หรือสถานที่ต้องการตรวจสอบความสะอาด และปลอดภัย เช่น โรงพยาบาล หรือบ้านพักที่มีผู้สูงอายุ ผู้ใช้สามารถทำการสังเกตข้อมูลจากที่ไหนก็ได้ โดยจากข้อมูลที่ได้รับยังสามารถที่จะตรวจสอบการไหวของแผ่นดินได้

## 5.2 ปัญหาและอุปสรรค

1. การแสดงผลข้อมูลบนหน้า dashboard ของ Magellan Platform ยังมีข้อผิดพลาดซึ่งส่งผลให้ทำการแสดงผลข้อมูลไม่ได้ตามเงื่อนไข

2. ในช่วงแรกของการดำเนินงานพบปัญหาในการใช้งาน Library BLE scan ของ ESP32 ที่ใช้งานบน Arduino IDE ซึ่งทำให้การทำงานเกิดข้อผิดพลาด

3. บางครั้งการสื่อสารกับพี่เลี้ยงผิดพลาดทำให้เข้าใจของเขตของการทำงานผิดพลาด

4. ESP32 เกิดการรีบูตตัวเองเนื่องจากหน่วยความจำภายในตัวถูกใช้งานจนหมด และในบางครั้งเกิดอาการค้างทำให้ไม่สามารถทำงานต่อได้

5. ต้องใช้ภาษาคอมพิวเตอร์หลายภาษา ทำให้ต้องศึกษาด้วยตัวเองเพิ่มเติมจึงส่งผลให้การทำงานเกิดความล่าช้า

6. ไม่ได้วางแผนการทำงานในภาพรวม ทำให้การคิดขั้นตอนหรือวิธีแก้ไขปัญหาต่าง ๆ อาจจะส่งผลให้การทำงานไม่ตรงตามผลลัพธ์ตามที่ต้องการได้

## 5.3 ข้อเสนอแนะและแนวทางในอนาคต

1. ควรสื่อสารกับพี่เลี้ยงให้ชัดเจนก่อนลงมือทำ และทำการจดเกี่ยวกับขอบเขตงาน หรือความต้องการของงาน

2. วางแผนก่อนการทำงานในภาพรวม และไล่การทำงานมาเป็นขั้นตอน เพื่อให้สามารถวางแผนการทำงาน หรือการแก้ไขปัญหาได้อย่างถูกต้อง และไม่ให้ผิดขอบเขตของงาน

3. พัฒนาระบบงานให้มีความเสถียรเพิ่มมากขึ้น และเพิ่มความ security ให้กับการใช้งาน

# บรรณานุกรม


[1] “ว่าด้วยเรื่อง Bluetooth Low Energy ตอน 1” [ออนไลน์]
เข้าถึงได้จาก : http://raspberrypi-thailand.blogspot.com/2018/01/bluetooth-low-energy-1.html (วันที่สืบค้น 20 พฤศจิกายน 2562)

[2] “Bluetooth Low Energy Scanning and Advertising” [ออนไลน์]
เข้าถึงได้จาก :
http://dev.ti.com/tirex/content/simplelink_academy_cc2640r2sdk_1_12_01_16/modules/ble_scan_adv_basic/ble_scan_adv_basic.html (วันที่สืบค้น 20 พฤศจิกายน 2562)

[3] “BLE Advertising Primer” [ออนไลน์]
เข้าถึงได้จาก : https://www.argenox.com/a-ble-advertising-primer/ (วันที่สืบค้น 20 พฤศจิกายน 2562)

[4] “Bluetooth Low Energy Discovery Process” [ออนไลน์]
เข้าถึงได้จาก : https://microchipdeveloper.com/wireless:ble-link-layer-discovery (วันที่สืบค้น 20 พฤศจิกายน 2562)

[5] “Bluetooth vs BLE -difference between Bluetooth and BLE(Bluetooth Low Energy)” [ออนไลน์]
เข้าถึงได้จาก : https://www.rfwireless-world.com/Terminology/Bluetooth-vs-BLE.html (วันที่สืบค้น 20 พฤศจิกายน 2562)

[6] “Arduino IDE อาดุยอิโน่ ไอดีอี คืออะไร” [ออนไลน์]
เข้าถึงได้จาก : https://bit.ly/37EWRgu (วันที่สืบค้น 20 พฤศจิกายน 2562)

[7] “Arduino อาดูโน่ หรืออาดูยอีโน่ คืออะไร” [ออนไลน์]
เข้าถึงได้จาก : https://bit.ly/35oW2qq (วันที่สืบค้น 20 พฤศจิกายน 2562)

[8] “Arduino ตอนที่ 5.0 โครงสร้างภาษา C Arduino เบื้องต้น” [ออนไลน์]
เข้าถึงได้จาก : https://bit.ly/2QL99gb (วันที่สืบค้น 20 พฤศจิกายน 2562)

# บรรณานุกรม (ต่อ)

[9] “Arduino ตอนที่ 5.5 โครงสร้างภาษา C Arduino เบื้องต้น” [ออนไลน์]
เข้าถึงได้จาก : https://bit.ly/39DEXfN (วันที่สืบค้น 20 พฤศจิกายน 2562)

[10] “NB-IoT คืออะไร? แล้วมันจะมาเปลี่ยนโลกการเกษตร และ โลกธุรกิจอย่างไร” [ออนไลน์]
เข้าถึงได้จาก : https://bit.ly/2ST2NOu (วันที่สืบค้น 20 พฤศจิกายน 2562)

[11] “ESP32 เบื้องต้น :: บทที่ 1 แนะนำ ESP32” [ออนไลน์]
เข้าถึงได้จาก : https://www.ioxhop.com/b/62 (วันที่สืบค้น 20 พฤศจิกายน 2562)

[12] “ESP32 #1: รู้จัก ไมโครคอนโทรลเลอร์ ESP32” [ออนไลน์]
เข้าถึงได้จาก : https://bit.ly/2tu72FQ (วันที่สืบค้น 20 พฤศจิกายน 2562)

[13] “SIM7020 CAT-M1/NB-IoT” [ออนไลน์]
เข้าถึงได้จาก : https://simcom.ee/modules/iot/sim7020/ (วันที่สืบค้น 20 พฤศจิกายน 2562)

[14] “โปรโตคอล CoAP” [ออนไลน์]
เข้าถึงได้จาก : http://www.enconlab.com/chiller/index.php/knowledge/11-c-chiller-system/17-chiller-7 (วันที่สืบค้น 20 พฤศจิกายน 2562)

[15] “ทำความรู้จักกับ JSON คืออะไร” [ออนไลน์]
เข้าถึงได้จาก : http://www.boxsingle.com/?page=Blog.ShowBlogDetail&blogID=13 (วันที่สืบค้น 20 พฤศจิกายน 2562)

# ประวัติผู้เขียน

**ชื่อ สกุล** นายจิรพัฒน์ ศิรสีห์

**วัน เดือน ปีเกิด** 15 กุมภาพันธ์ พ.ศ. 2541

**สถานที่อยู่ปัจจุบัน** 56 ตรอกไมตรี เขตป้อปราบศัตรูพ่าย แขวงป้อมปราบฯ จังหวัดกรุงเทพมหานคร 10100

**ประวัติการศึกษา**

| | | |
|---|---|---|
| ปีการศึกษา | 2559-ปัจจุบัน | สาขาวิชาวิศวกรรมสารสนเทศ<br>ภาควิชาวิศวกรรมคอมพิวเตอร์<br>สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง |
| ปีการศึกษา | 2553-2558 | แผนการเรียนคณิตศาสตร์-วิทยาศาสตร์<br>โรงเรียนกุหลาบวิทยา |
| ปีการศึกษา | 2547-2552 | โรงเรียนกุหลาบวิทยา |