

# เริ่มต้นเรียนรู้และพัฒนาอุปกรณ์

# Internet of Things (IoT)

# กับ NodeMCU

---

Getting start the Internet of Things (IoT)  
with NodeMCU

เรียนโดย<sup>๑</sup>  
ธีราฐ จิตรพรหมา<sup>๒</sup>  
ชัยวัฒน์ ลิ้มพรจิตรวิไล<sup>๓</sup>

(C) Innovative Experiment Co.,Ltd.



## Getting start the Internet of Things (IoT) with NodeMCU เริ่มต้นเรียนรู้และพัฒนาอุปกรณ์ Internet of Things (IoT) กับ NodeMCU

ส่วนผลิตสิทธ์ตาม พ.ร.บ. ลิขสิทธ์ พ.ศ. 2521

ห้ามการลอกเลียนไม่ว่าส่วนหนึ่งส่วนใดของหนังสือเล่มนี้ นอกจะจะได้รับอนุญาต

### โครงการใช้หนังสือเล่มนี้

- นักเรียน นิสิต นักศึกษา และบุคคลทั่วไปที่มีความสนใจในการนำไปประยุกต์ใช้ในการพัฒนาระบบและอุปกรณ์เพื่อเชื่อมต่อกับเครือข่ายอินเทอร์เน็ต
- สถาบันการศึกษา โรงเรียน วิทยาลัย มหาวิทยาลัย ที่มีการเปิดการเรียนการสอนวิชาอิเล็กทรอนิกส์ หรือภาควิชาวิศวกรรมอิเล็กทรอนิกส์และคอมพิวเตอร์
- คณาจารย์ที่มีความต้องการศึกษา และเตรียมการเรียนการสอนวิชาไม่ครอบคลุมถึงระบบเครือข่ายข้อมูลผ่านอินเทอร์เน็ต ที่ต้องบูรณาการความรู้ทางอิเล็กทรอนิกส์-ไมโครคอนโทรลเลอร์ การเขียนโปรแกรมคอมพิวเตอร์-ระบบเครือข่ายและสื่อสารข้อมูลในระดับมัธยมศึกษา อาชีวศึกษา และปริญญาตรี

ดำเนินการจัดพิมพ์และจำหน่ายโดย

บริษัท อินโนเวติฟ เอ็กเพอริเมนต์ จำกัด

108 ช.สุขุมวิท 101/2 ถ.สุขุมวิท แขวงบางนา เขตบางนา กรุงเทพฯ 10260

โทรศัพท์ 0-2747-7001-4

โทรสาร 0-2747-7005

รายละเอียดที่ปรากฏในหนังสือเล่มนี้ผ่านการตรวจทานอย่างละเอียดและถ้วนถี่ เพื่อให้มีความสมบูรณ์ และถูกต้องมากที่สุดภายใต้เงื่อนไขและเวลาที่พึงมีก่อนการจัดพิมพ์เผยแพร่ ความเสียหายอันอาจเกิดจาก การนำข้อมูลในหนังสือเล่มนี้ไปใช้ ทางบริษัท อินโนเวติฟ เอ็กเพอริเมนต์ จำกัด มิได้มีภาระในการรับผิดชอบแต่ประการใด ความผิดพลาดคลาดเคลื่อนที่อาจมีและได้รับการจัดพิมพ์เผยแพร่ออกไป นั้น ทางบริษัทฯ จะพยายามชี้แจงและแก้ไขในการจัดพิมพ์ครั้งต่อไป

# คำชี้แจงจากคณ.:ผู้เขียน/เรียบเรียง

---

การนำเสนอข้อมูลเกี่ยวกับข้อมูลทางเทคนิคและเทคโนโลยีในหนังสือเล่มนี้ เกิดจากความต้องการที่จะอธิบายกระบวนการและหลักการทำงาน ของอุปกรณ์ในภาพรวมด้วยถ้อยคำที่ง่ายเพื่อสร้างความเข้าใจแก่ผู้อ่าน ดังนั้นการแปลคำศัพท์ทางเทคนิคหลายๆ คำอาจไม่ตรงตามข้อบัญญัติของราชบัญญัติยสถาน และมีหลายๆ คำที่ยังไม่มีการบัญญัติอย่างเป็นทางการ คณ.ผู้เขียนจึงขออนุญาตบัญญัติศัพท์ขึ้นมาใช้ในการอธิบาย โดยมีข้อจำกัดเพื่อข้างอิงในหนังสือเล่มนี้เท่านั้น

ทั้งนี้สาเหตุหลักของข้อชี้แจงนี้มาจากการรับรวมข้อมูลของอุปกรณ์ในระบบสมองกลฝังตัว และเทคโนโลยีหุ่นยนต์สำหรับการศึกษาเพื่อนำมาเรียบเรียงเป็นภาษาไทยนั้นทำได้ไม่ง่ายนัก ทางคณ.ผู้เขียนต้องทำการรวบรวมและทดลองเพื่อให้แน่ใจว่า ความเข้าใจในกระบวนการทำงานต่างๆ นั้นมีความคลาดเคลื่อนน้อยที่สุด

เมื่อต้องทำการเรียบเรียงอุปกรณ์เป็นภาษาไทย ศัพท์ทางเทคนิคหลายคำมีความหมายที่ทับซ้อนกันมาก การบัญญัติศัพท์จึงเกิดจาก การปฏิบัติจริงร่วมกับความหมายทางภาษาศาสตร์ ดังนั้นหากมีความคลาดเคลื่อนหรือผิดพลาดเกิดขึ้น ทางคณ.ผู้เขียนขออภัยรับและหากได้รับคำอธิบายหรือชี้แนะจากท่านผู้อ่านจะได้ทำการซื้อและปรับปรุงข้อผิดพลาดที่อาจมีเหล่านั้นโดยเร็วที่สุด

ทั้งนี้เพื่อให้การพัฒนาสื่อทางวิชาการ โดยเฉพาะอย่างยิ่งกับความรู้ของเทคโนโลยีสมัยใหม่ สามารถดำเนินไปได้อย่างต่อเนื่อง ภายใต้การมีส่วนร่วมของผู้อ่านทุกภาคส่วน

ในหนังสือเล่มนี้ได้ทำการอ้างอิงถึงซอฟต์แวร์และเว็บไซต์หลายแห่ง ทางผู้จัดทำไม่อาจรับประกัน หรือรับรองการคงอยู่ของสินค้าและบริการใดๆ ที่นำเสนอหรืออ้างอิงในหนังสือเล่มนี้ อย่างไรก็ตาม ในขณะจัดทำหนังสือเล่มนี้ ทางผู้จัดทำได้ทำการทดลองและทดสอบภาษาไทย ครอบเวลาและภาษาปรับปรุงล่าสุดในเวลานั้นๆ หากมีการปรับปรุงอื่นๆ ให้ทำให้การนำเสนอคณาจารย์และผู้อ่านได้ทางผู้จัดทำจะพยายามปรับปรุงเนื้อหาและเผยแพร่ผ่านทางเว็บไซต์และสื่อสารรณรงค์ของบริษัท อินโนเวติฟ เอ็กเพอริเมนต์ จำกัด ที่ [www.inex.co.th](http://www.inex.co.th) และ [www.facebook.com/innovativeexperiment](https://www.facebook.com/innovativeexperiment)

# สารบัญ

---

บทที่ 1 ว่าด้วยเรื่อง Internet of Things อินเทอร์เน็ตสำหรับสิ่งของ.....	7
บทที่ 2 แนะนำอุปกรณ์ IoT Education Kit .....	17
บทที่ 3 การพัฒนาโปรแกรมภาษา C/C++ สำหรับ NodeMCU ด้วย Arduino IDE .....	33
บทที่ 4 NodeMCU กับการติดต่ออุปกรณ์อินพุตเอาต์พุตพื้นฐาน.....	43
บทที่ 5 NodeMCU กับการขับโหลดกระแสไฟฟ้าสูงด้วยโซลิดสเตตปรีเมอร์.....	73
บทที่ 6 ใช้งาน NodeMCU กับตัวตรวจจับ.....	79
บทที่ 7 เชื่อมต่อ NodeMCU กับเครือข่ายไร้สาย WiFi.....	103
บทที่ 8 เชื่อมต่อ NodeMCU กับคลาวด์เซิร์ฟเวอร์.....	109
บทที่ 9 แสดงผลข้อมูลของอุปกรณ์ IoT ด้วย freeboard.....	125
บทที่ 10 อ่านค่าตัวตรวจจับแสดงผลบน freeboard.....	137
บทที่ 11 ควบคุม LED ผ่านเครือข่ายอินเทอร์เน็ต โดยใช้ freeboard และ dweet .....	149

# บทนำ

---

**Internet of Things (IoT)** หรือ อินเทอร์เน็ตสำหรับสิ่งของ เป็นเทคโนโลยีที่บ่งบอกชัดเจน ว่า เป็นการนำอินเทอร์เน็ตมาเป็นส่วนหนึ่งของสิ่งของต่างๆ ทำให้สิ่งของถูกทำให้เกิดความเชื่อมโยง เชื่อมต่อ และเปลี่ยนข้อมูล สิ่งงาน ควบคุม หรือตรวจสอบได้ โดยคนหรือระบบที่คุ้มครองต่อหน้า จะอยู่ที่ใดก็ได้ เพราะการสื่อสารกันจะกระทำผ่านเครือข่ายอินเทอร์เน็ต

ดังนั้นการเรียนรู้เทคโนโลยี IoT จึงเป็นสิ่งที่จำเป็นแล้วสำหรับผู้คนสมัยใหม่ โดยเฉพาะอย่าง ยิ่งบุคลากรที่ต้องทำงานเกี่ยวกับระบบสมองกลฝังตัว เว็บแอปพลิเคชัน ผู้คุ้มครองข้อมูล หรือ กระทั่งผู้ใช้งานทั่วไป วันนี้การควบคุมหรือติดต่อสื่อสารอุปกรณ์มิได้เป็นเพียงการติดต่อแบบซ่อนๆ ตรงๆ อีกต่อไป มันถูกขยายหรือต่อยอดให้เกิดเป็นระบบ ทำให้โครงสร้างที่สามารถเข้าถึงเครือข่ายของ อุปกรณ์ สามารถอ่านค่า ส่งค่า และควบคุมการทำงาน ทุกอย่างเกิดขึ้นจริงแล้ว และจะทวีทั้ง จำนวนอุปกรณ์ ขนาดของระบบ และงบประมาณเข้ามายังเทคโนโลยีนี้

ในอดีต การติดต่อสื่อสารและควบคุมอุปกรณ์ผ่านเครือข่ายพื้นฐานหรือ LAN เป็นเรื่องไม่ง่าย และถ้ายิ่งเป็นการสื่อสารแบบไร้สายผ่าน WiFi ยิ่งเป็นเรื่องที่เมกะเกร porr หรือนักเดิน นักทดลอง ต้อง ใช้ความพยายามสูง บวกกับอุปสรรคใหญ่คือ ราคาของอุปกรณ์ ทำให้การพัฒนาอุปกรณ์ IoT กลาย เป็นเรื่องยากลำบาก ไว้ในวงแคบๆ ของนักพัฒนาด้านไฮเอ็นด์เท่านั้น

จนกระทั่ง ในปี ค.ศ. 2014 ได้เกิดการเปลี่ยนแปลงครั้งสำคัญขึ้น ในแวดวงระบบสมองกลฝังตัว การประภูติของโมดูล Serial WiFi ราคาถูกมากที่ชื่อว่า **ESP8266** ทำให้การพัฒนาอุปกรณ์ IoT เป็นไปได้โดยไม่ต้องมีความรู้ทางด้าน hardware มากนัก ตัว ESP8266 ที่หาซื้อได้ในระดับหลักร้อยบาทหรือไม่เกิน US\$5 สำหรับรุ่นตั้งต้น ก่อให้เกิดการตื่นตัวในการพัฒนาอุปกรณ์เพื่อเชื่อมต่อระบบเครือข่าย WiFi ราคาประหยัดอย่างกว้างขวาง ไปทั่วโลก รวมไปถึงความพยายามในการสร้างเครื่องมือพัฒนา โปรแกรมให้แก่ ESP8266 ให้ง่ายและสะดวกขึ้น นั่นจึงทำให้เกิด **Arduino IDE for ESP8266** ขึ้น

ด้วยความง่ายของโปรแกรมภาษา C/C++ . ในสไต์ล์ Arduino ทำให้ครูฯ ก็สามารถเข้าสู่การ พัฒนาอุปกรณ์เพื่อเชื่อมต่อเครือข่าย WiFi ได้ และต่อยอดไปสู่การติดต่อหรือควบคุมระบบผ่านเครือข่ายอินเทอร์เน็ต ได้ไม่ยาก

ชุดเรียนรู้และพัฒนาอุปกรณ์ IoT เป็นองค์ประกอบที่สำคัญที่สุด ที่สามารถเริ่มต้นเรียนรู้ ทำการทดลองเขียนโปรแกรม ทดสอบการทำงานของชาร์ดแวร์ รู้จักกับการต่อวงจรอิเล็กทรอนิกส์ พื้นฐาน เรียนรู้การเชื่อมต่อระหว่างอุปกรณ์กับเครือข่ายไร้สาย WiFi และกับเครือข่ายที่ใหญ่กว่าอย่าง อินเทอร์เน็ต

หนังสือเล่มนี้เป็นส่วนหนึ่งของชุดเรียนรู้ดังกล่าว โดยคณะผู้จัดทำพยายามอย่างยิ่งในการเรียนเรียงเนื้อหาเพื่อให้เข้าใจถึงแนวทางการพัฒนาอุปกรณ์ IoT อย่างง่ายด้วย NodeMCU-12E หรือ V2 หรือ V1 Development kit กับซอฟต์แวร์ Arduino IDE รุ่นพิเศษที่มีไลบรารีรองรับการทำงานกับ NodeMCU และการเชื่อมต่อกับเครือข่าย WiFi

เนื้อหาในหนังสือจะเริ่มต้นจากการแนะนำให้รู้จักกับเทคโนโลยี IoT เป็นต้น ตามด้วยแนะนำนำอุปกรณ์ที่ใช้ในการเรียนรู้ เครื่องมือทางซอฟต์แวร์ที่ใช้ในการพัฒนาโปรแกรม การทดลองใช้งาน NodeMCU เป็นต้น ตัวอย่างการทดลองเพื่อเชื่อมต่อกับเครือข่าย WiFi โปรโตคอลที่ใช้ในการสื่อสาร เป็นต้น รู้จักกับคลาวด์เซิฟเวอร์ การติดต่อเพื่อรับส่งข้อมูล จนนำไปสู่การสร้างแอปพลิเคชันของ อุปกรณ์ IoT อย่างง่ายขึ้นมาได้

คณะผู้จัดทำเรียบเรียงหนังสือเล่มนี้ในแบบ “เล่าเท่าที่รู้” และพยายามทำให้ผู้อ่านหรือผู้ที่นำไปใช้ในการเริ่มต้นเรียนรู้เข้าใจให้ได้มากที่สุด วันนี้เราต้องเริ่มต้นนับหนึ่งกับการเรียนรู้และพัฒนา อุปกรณ์ IoT แล้ว แม้ว่าในปลายทางจะสามารถเป็นผู้ผลิตอุปกรณ์เพื่อจำหน่ายได้หรือไม่ก็ตาม หากแต่การที่รู้และเข้าใจจะช่วยให้อย่างน้อยเราใช้งานหรือดูแลอุปกรณ์เหล่านี้ได้อย่างมีประสิทธิภาพตาม สมควร

บางที่ IoT อาจไม่ใช่เกินเทอร์เน็ตสำหรับสิ่งของ มันอาจกลายเป็นอินเทอร์เน็ตสำหรับทุก สรรพสิ่ง ทั้งสิ่งมีชีวิตและไม่มีชีวิต กาลเวลาท่านนี้จะเป็นผู้ให้คำตอบ

ชัยวัฒน์ ลิ้มพรจิตรวิໄລ

บรรณาธิการ

## บทที่ 1

# ว่าด้วยเรื่อง Internet of Things (IoT) อินเทอร์เน็ตสำหรับสิ่งของ

---

Internet of Things คำนี้เกิดขึ้นมาตั้งแต่ปี ค.ศ. 1999 โดย Kevin Ashton แห่ง MIT's Media center เขาได้นำเสนอแนวคิดว่า มันคือ การนำสิ่งของต่างๆ ไม่ว่าจะเป็นคอมพิวเตอร์, เครื่องจักร และตัวตรวจจับมาเชื่อมต่อกับเครือข่ายอินเทอร์เน็ต เพื่อรายงานสถานะการทำงาน สถานะข้อมูล และรับรู้คำสั่งควบคุม สิ่งที่น่าประหลาดใจคือ ในช่วงเวลานั้นโลกเพิ่งรู้จักและใช้งานอินเทอร์เน็ตได้ไม่นาน แต่ Kevin มองเห็นอนาคตและพัฒนาการของสรรพสิ่งที่จะต้องเชื่อมโยงถึงกันผ่านเครือข่าย อินเทอร์เน็ต

### 1.1 ก่อนจะมาเป็นชื่อ IoT

แม้ว่าแนวคิดของ IoT ลูกน้ำเสนอตั้งแต่ปี ค.ศ. 1999 แต่ไม่ได้รับการตอบรับมากนัก อาจมาจากสาเหตุที่ว่า ในเวลานั้นอินเทอร์เน็ตเป็นเรื่องกลุ่มคนเฉพาะ ดูยุ่งยาก และต้องการทรัพยากรามาก แต่ก็มีคนนำแนวคิด IoT ไปสานต่อ และมีชื่อเรียกแตกต่างกันไป อาทิ

**Machine-to-machine (M2M)**

**Ubiquitous Computing**

**Embedded Computing**

**Smart Service**

**Industrial Internet**

จนกระทั่งวันนี้ เมื่ออินเทอร์เน็ตเข้าถึงทุกคน ทุกบ้าน ทำให้แนวคิด Internet of Things ได้รับการยอมรับ และเริ่กงานเทคโนโลยีด้วยชื่อเดิมที่ถูกคิดมาตั้งแต่ปี ค.ศ. 1999

## 1.2 ความหมายของ IoT\*

IoT หรือ Internet of Things หมายถึง เทคโนโลยีที่ก่อให้เกิดการเชื่อมโยงกันของสิ่งของ ผู้คน ข้อมูล และการบริการเข้ากับเครือข่ายอินเทอร์เน็ต ปัจจัยสำคัญในการทำให้เกิด IoT ได้คือ การบรรจุ อุปกรณ์สมองกลฝังตัวหรือ embedded system device เข้าไปใน “สิ่งของ” หรือเครื่องมือ เครื่องใช้ต่างๆ มีตัวตรวจจับหรือเซนเซอร์เพื่อตรวจวัดค่าที่สนใจ แล้วส่งมาบังส่วนสมองกล เพื่อส่งต่อมาบังส่วน ประมวลผลกลางและฐานข้อมูลผ่านเครือข่ายอินเทอร์เน็ต ในส่วนหลังนี้มีชื่อเรียกด้วยศัพท์สมัยใหม่ ว่า คลาวด์เซิร์ฟเวอร์ (cloud server)

ด้วยการนำอุปกรณ์สมองกลฝังตัวบรรจุลงใน “สิ่งของ” ต่างๆ ทำให้ “สิ่งของ” เหล่านี้ทำงาน ในแบบอัจฉริยะได้ อุปกรณ์เครื่องใช้ต่างๆ ในบ้าน ในโรงงาน ในที่ทำงาน ในyanพาหนะ ล้วนแล้วแต่ ใช้ระบบสมองกลฝังตัวมากขึ้น ทำให้มันทำงานได้ด้วยตัวเอง และ/หรือรวมเข้าเป็นส่วนหนึ่งของระบบ ใหญ่ เกิดการเชื่อมโยงการทำงานเป็นระบบได้

การทำให้ “สิ่งของ” ทำงานร่วมกันผ่านเครือข่ายอินเทอร์เน็ต จึงทำให้เกิดนิยามของเทคโนโลยี นี้ขึ้น Internet of Things หรือ IoT เป็นการขยายขอบเขตการทำงานของอินเทอร์เน็ต ให้กว้างและลึกลง ไปถึงการเชื่อมต่อเพื่อสื่อสารข้อมูลกับ “สิ่งของ” ทำให้เกิดการรับส่งข้อมูลและตอบสนองในแบบทุกที่ ทุกเวลา และทุกสิ่งของ ได้ในที่สุด

Internet of Thing-IoT เป็นระบบทำงานของสิ่งของอย่างอัตโนมัติ ซึ่งอาจเป็น Person to Things-P2T หรือ Things to Things-T2T เป็นการประยุกต์ที่ใช้งานได้มาก นับเป็นเทคโนโลยีที่มีการ เดินทางต่อเนื่อง การประยุกต์ และการให้บริการบน IoT สูง มีคุณค่าเพิ่มทางเศรษฐกิจ เป็นระบบเปิด ที่พัฒนาต่ออยู่ด้วยกัน นับเป็นเทคโนโลยีร่วมสมัยที่ต้องให้ความสนใจ

การทำงานบนพื้นฐานระบบอัจฉริยะเริ่มจาก Machine to Machine (M2M) เป็นการเชื่อมโยง ระหว่างอุปกรณ์กับอุปกรณ์ ซึ่งเป็นส่วนหนึ่งของ IoT โดยอุปกรณ์ต่างๆ จะต่อเขื่อมกัน ทั้งแบบเชื่อม ต่อตรงหรือผ่านเครือข่าย ทำให้กลไกเป็นส่วนขยายของอินเทอร์เน็ต ดังนั้นการพัฒนาโครงสร้างพื้น ฐานให้รองรับกับ IoT จึงต้องเพิ่มความเร็ว เพิ่มขนาดช่องสัญญาณ เพิ่มขีดความสามารถของเครือข่าย ให้มีความอัจฉริยะ โดยใช้เครือข่ายเป็นฐาน เพื่อใช้ข้อมูลร่วมกัน

## 1.3 ประโยชน์ของ IoT

ในรูปที่ 1-1 เป็นตัวอย่างของการใช้ประโยชน์จาก IoT โดยแสดงให้เห็นถึงความสัมพันธ์ของผู้คน ชุมชน การดำเนินชีวิตสมัยใหม่ผ่านเทคโนโลยี IoT เริ่มจาก นักเรียน นักศึกษา ได้เรียนรู้เกี่ยวกับอุปกรณ์ IoT จนนำไปสู่การสร้างโครงงานเพื่อส่งต่อหรือร้องขอข้อมูลเพื่อนำไปใช้ประโยชน์ ทำให้พวกเขางานเป็น กลุ่มคนในอนาคตที่จะพัฒนาและบำรุงรักษาเทคโนโลยีต่อไป

\* หัวข้อนี้ได้ทำการเรียบเรียงใหม่จากข้อเขียนของ รศ. ยืน ภู่วรรณ



รูปที่ 1-1 ตัวอย่างไดอะแกรมแสดงถึงประโยชน์ของอุปกรณ์และเทคโนโลยี IoT ที่มีต่อชุมชน

ด้านชุมชน ได้ใช้อุปกรณ์ IoT ใน การตรวจสอบสภาพแวดล้อม การใช้พลังงาน ควบคุม สาธารณูปโภคอย่างชาญฉลาด ช่วยให้เกิดความเป็นอยู่ที่ปลอดภัยเป็นปกติสุข

ด้านการดำรงชีวิตในรูปที่ 1-1 เน้นไปที่ เกษตรกร พากเพียได้ใช้ประโยชน์จากอุปกรณ์ IoT ใน การตรวจสอบสภาพดิน น้ำ อากาศ แล้วส่งข้อมูลผ่านระบบคลาวด์เพื่อนำมาประมวลผล จนนำไปสู่ การตัดสินใจแก้ไข หรือปรับปรุงกรรมวิธีในการทำการเกษตร ส่งผลดีต่อปริมาณและคุณภาพของ ผลิตภัณฑ์ โดยมีการประสานความร่วมมือกับนักพัฒนาอุปกรณ์ IoT ที่มีการนำข้อมูลแบ่งปันกับนัก วิชาการเกษตรในพื้นที่อื่นๆ ทั่วโลก เพื่อนำมาปรับปรุงอุปกรณ์ IoT ให้เหมาะสมกับสภาพแวดล้อม ที่ใช้งาน ทำให้ได้ข้อมูลที่ถูกต้อง นำไปสู่การตัดสินใจแก้ไขปัญหาหรือปรับปรุงกระบวนการผลิต ของเกษตรกร ได้อย่างถูกต้องมากที่สุดต่อไป

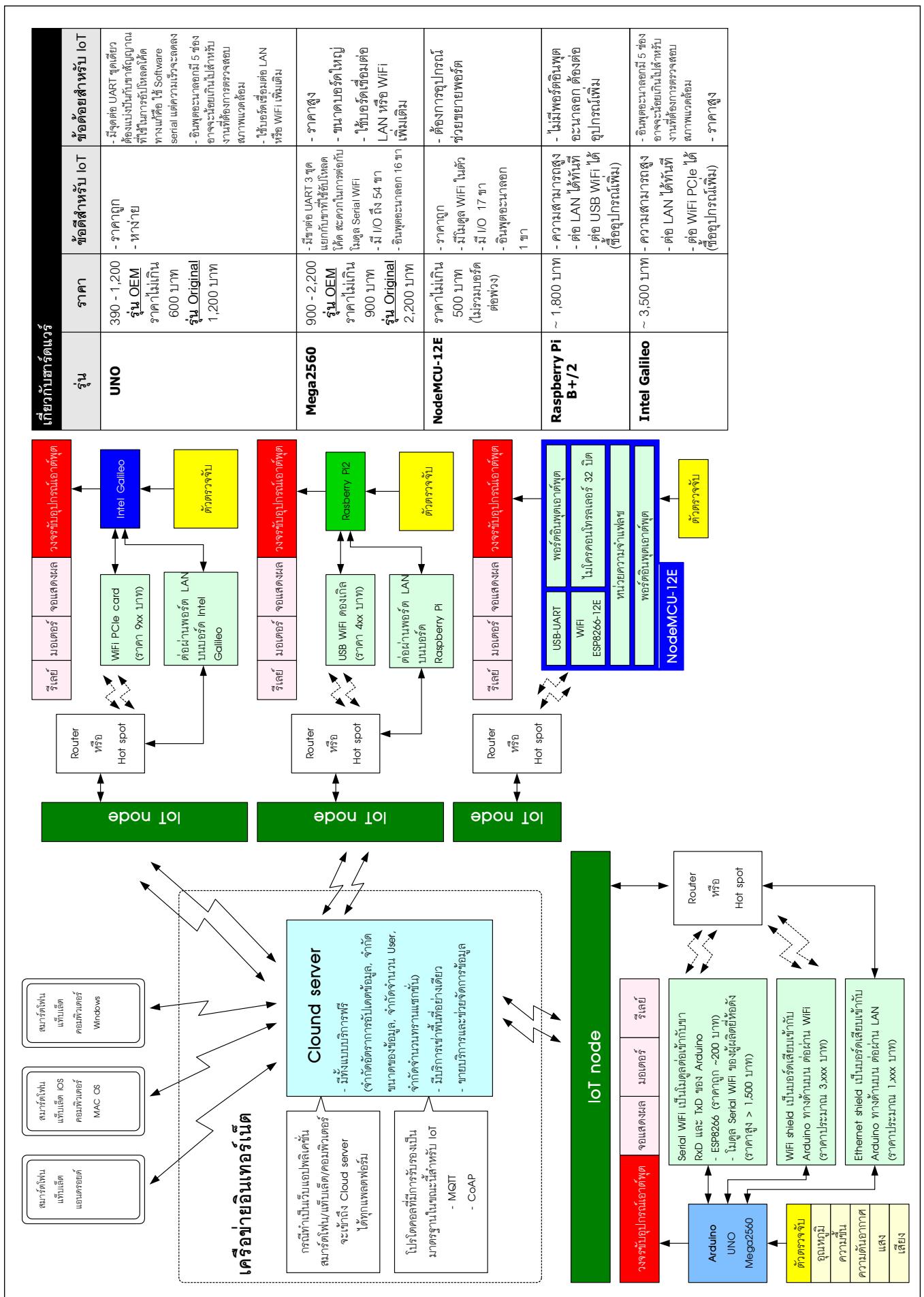
IoT นำมาซึ่งการพัฒนาเมืองอัจฉริยะหรือ Smart City ผู้คนในเมืองนี้ย่อมต้องการบริการต่างๆ ที่สะดวกสบายมากขึ้น อาทิ การเดินทางด้วยยานพาหนะที่หลากหลาย จึงต้องพัฒนา มีการพัฒนา IoV-Internet of Vehicle โดยมีการสร้างสาธารณูปโภคขั้นพื้นฐาน ให้รองรับ ก่อให้เกิดการเชื่อมต่อในลักษณะ V2I หรือ Vehicle to Infrastructure ไม่ว่าจะเป็น อุปกรณ์บอสสัญญาณต่างๆ บอคเต้าແเนนง บอคสภาพผู้การจราจร นอกจากนี้ ยานพาหนะจำเป็นต้องติดต่อสื่อสารกันเอง หรือ V2V-Vehicle to Vehicle ทั้งนี้เพื่อให้เกิดการ เดินทางที่รวดเร็ว สะดวก ปลอดภัย การสร้างเมืองอัจฉริยะจึงเกี่ยวข้องกับเทคโนโลยีหลากหลาย ไม่ว่า จะเป็นการสื่อสาร ไร้สายที่ต่อ กับเครือข่ายอินเทอร์เน็ต, ระบบสมองกลฝังตัว, เทคโนโลยีโครงข่ายตัว ตรวจจับอัจฉริยะแบบไร้สาย, ระบบอาคารหรือบ้านอัตโนมัติ เป็นต้น\*

## 1.4 ส่วนประกอบของ IoT

ระบบหรือเทคโนโลยี IoT จะเกิดขึ้นได้ต้องมีองค์ประกอบครบถ้วน\*

1. สิ่งของ
2. อุปกรณ์ (ตัวควบคุม, ตัวตรวจจับ และอุปกรณ์ขับให้ลดหรืออุปกรณ์เอาต์พุต)
3. ระบบเชื่อมต่ออินเทอร์เน็ต (จะเป็นแบบมีสายหรือไร้สายก็ได้)
4. ข้อมูล
5. ระบบจัดการฐานข้อมูลคลาวด์เซิร์ฟเวอร์ (Cloud server)

\* ย่อหน้านี้ได้ทำการเรียบเรียงใหม่จากข้อเขียนของ รศ. ยืน ภู่วรรณ



รูปที่ 1-2 ไดอะแกรมเบื้องต้นของแนวทางการพัฒนาอุปกรณ์ IoT ในแบบ DIY

## 1.5 แนวทางในการพัฒนาอุปกรณ์ IoT ในแบบ DIY

ในรูปที่ 1-2 แสดงໄດ້ແກຣມເບື້ອງຕົ້ນຂອງแนวทางการพัฒนาອุปกรณ์ IoT ในแบบ DIY (Do it yourself) ໂດຍໃຊ້ສາງແວຣ໌ທີ່ຫາໄດ້ໃນປະເທດໄທຢ ຈາກໃນຮູບແນະນຳ 3 ແບນໍລັກຄື່ອງ

1. ບ່ອຮົດໃນອນຸກຣມ Arduino ທັງ UNO ແລະ Mega2560
2. ບ່ອຮົດຄອມພິວເຕອີ່ Rasperry Pi ຢ່ອ Embedded PC
3. ໂມຄູລ NodeMCU

ໃນແບນແຮກເປັນການໃຊ້ບ່ອຮົດໄນ ໂກຮຄອນໂທຣລເລອຣໃນອນຸກຣມ Arduino ຜຶ້ງເປັນໂວເຟ່ນໜ້ອຮັສ ແພລຕິໂຟຣົມທີ່ໄດ້ຮັບຄວາມນິຍົມສູງ ອຸປະກົນສຳຄັນທີ່ຕ້ອງມີຄື່ອງ ໂມຄູລທີ່ອວງຈະຮົອນບ່ອຮົດຕ່ອຨ່ງທີ່ເຮັກວ່າໜີ້ລົດ (shield) ທີ່ທຳໃຫ້ບ່ອຮົດ Arduino ເຊື່ອມຕ່ອກັບເຄື່ອງຂ່າຍອິນເທຼວຣົນເນີຕ ໄດ້ ໃນການທີ່ເປັນ ໂມຄູລຕົວທີ່ໄດ້ຮັບຄວາມນິຍົມຄື່ອງ ESP-01 ຜຶ້ງໃຊ້ຊີປ WiFi ຄອນໂທຣລເລອຣໃນຕະຮູບ ESP8266 ໂດຍໃຊ້ເພີ່ງ 2 ຂາໃນ ການເຊື່ອມຕ່ອກັບອ່າງ RxD ແລະ TxD ລາກເປັນບ່ອຮົດໜີ້ລົດກີ່ຈະມີທັງອື່ເອຣ້ເນີຕໜີ້ລົດແລະ WiFi ຜີ້ລົດ



(ກ) ບ່ອຮົດ Arduino UNO



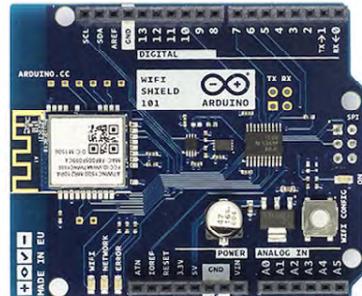
(ຂ) ບ່ອຮົດ Arduino MEGA 2560



(ຄ) ESP-01 ໂມຄູລ WiFi  
ທີ່ໃຊ້ຊີປ ESP8266 ຕິດຕ່ອ  
ແບນອນຸກຣມ UART

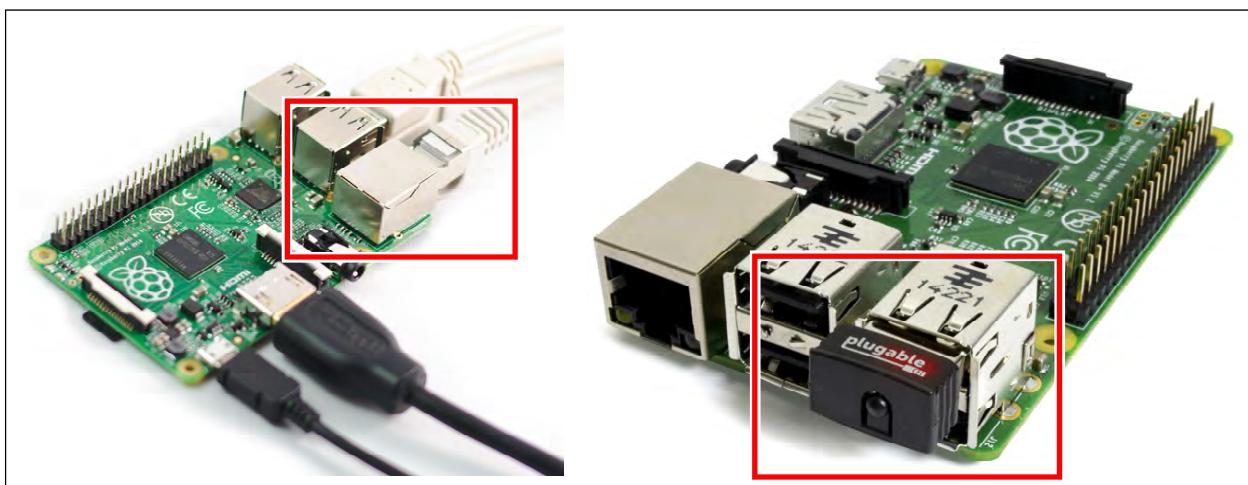


(ງ) ບ່ອຮົດອື່ເອຣ້ເນີຕໜີ້ລົດ



(ຈ) ບ່ອຮົດ WiFi ຜີ້ລົດ

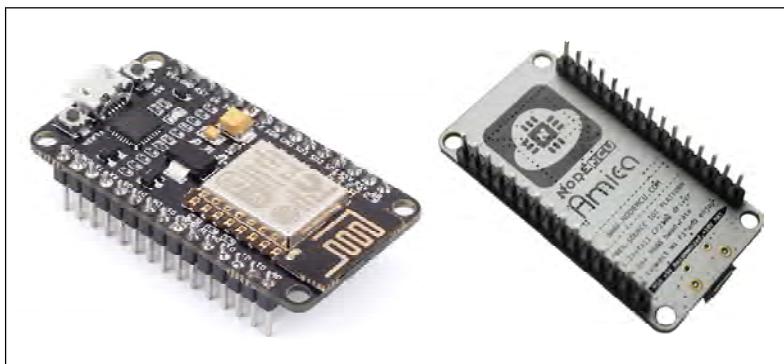
ຮູບທີ່ 1-3 ບ່ອຮົດ Arduino UNO ແລະ ບ່ອຮົດໜີ້ລົດສຳຫັບເຂື່ອມຕ່ອກັບເຄື່ອງຂ່າຍອິນເທຼວຣົນເນີຕ



**รูปที่ 1-4 Raspberry Pi 2 ที่รองรับการเชื่อมต่อ กับเครือข่ายอินเทอร์เน็ตทั้งแบบสายผ่านพอร์ตอีเธอร์เน็ต (ภาคซ้าย) และแบบไร้สายผ่าน USB WiFi dongle (ภาคขวา)**

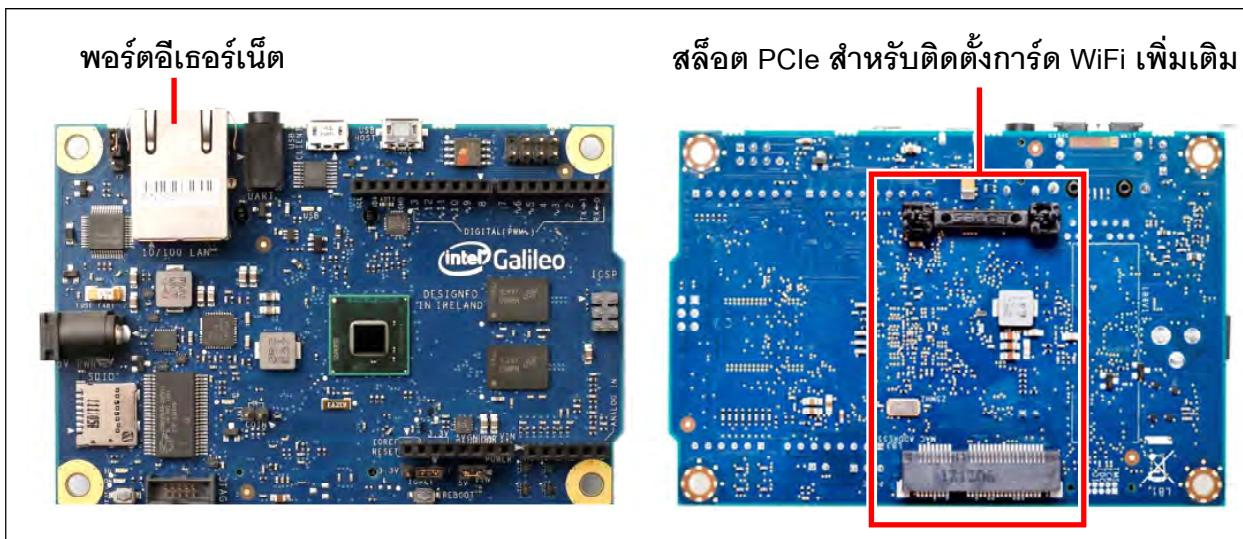
ในแบบที่สอง เป็นการใช้บอร์ดคอมพิวเตอร์ 32 บิตร่วมสมัยอย่าง Raspberry Pi ซึ่งใช้ได้ทั้งรุ่น B, B+ และ 2 ยังรวมไปถึงบอร์ดผู้ผลิตรายอื่น ไม่ว่าจะเป็น Beagle Bone Black, Nano Pi, Banana Pi, Odroid หรืออื่นๆ ที่มีคุณสมบัติเทียบเคียงกัน โดยบอร์ดทั้งหมดที่กล่าวมาจะมีพอร์ตอีเธอร์เน็ตเพื่อเชื่อมต่อ กับเครือข่ายอินเทอร์เน็ต มีพอร์ตอินพุตเอาต์พุตเพื่อเชื่อมต่อ กับตัวตรวจจับและอุปกรณ์เอาต์พุต ภายนอกเพื่อสั่งการและควบคุม ได้ หากต้องการเชื่อมต่อ กับเครือข่ายแบบไร้สาย ก็ต้องจัดหา USB WiFi dongle มาต่อเพิ่มเติม

แบบที่สาม เป็นแบบที่หนังสือเล่มนี้จะใช้อ้างอิงเป็นหลัก นั่นคือการใช้โมดูล NodeMCU-12E หรือ V2 หรือ Development kit 1.0 (บื้นกับการเรียกของผู้ผลิต) โดย NodeMCU-12E นี้ใช้โมดูล WiFi คอนโทรลเลอร์ ESP8266-12E จาก Espressif System ซึ่งมีขาพอร์ตอินพุตเอาต์พุตที่มากพอสำหรับการนำไปใช้งาน ทั้งการติดต่อ กับตัวตรวจจับแบบดิจิตอลและอนาล็อก และการติดต่อ กับอุปกรณ์เอาต์พุต เพื่อขับให้ทำงาน ได้ ที่สำคัญคือ ราคาของ NodeMCU-12E ถูก ทำให้ต้นทุนในการพัฒนา 低廉 สำหรับ IoT ในแบบ DIY ต่ำสุด ด้านการพัฒนาโปรแกรมทำได้ไม่ยากด้วยการใช้เครื่องมือของ Arduino IDE รุ่นที่มีการพัฒนาให้รองรับกับ NodeMCU

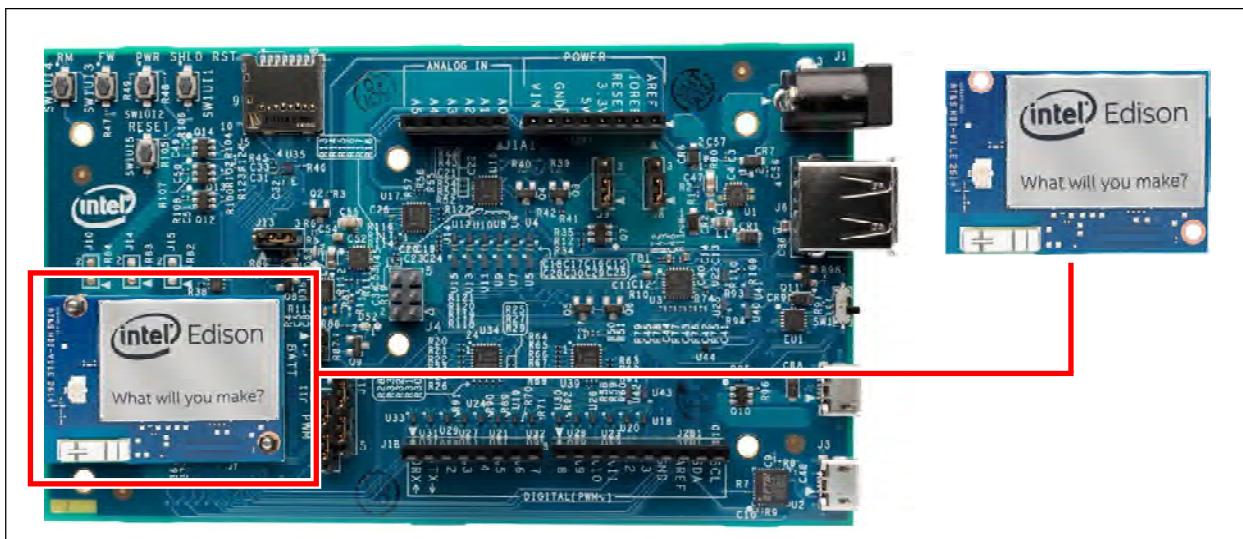


**รูปที่ 1-5 NodeMCU-12E ที่มาพร้อม กับโมดูล WiFi คอนโทรลเลอร์ในตัว ทำให้นำมาพัฒนาเป็นอุปกรณ์ IoT ได้ด้วยต้นทุนที่ต่ำ**

นอกจากนั้น ยังมีบอร์ดอย่าง Intel Galileo หรือ Intel Edison จาก Intel ที่นำมาพัฒนาเป็นอุปกรณ์ IoT ได้ โดย Galileo มีพอร์ตที่ตรงกับ Arduino UNO และมีพอร์ตอีเธอร์เน็ตในตัวสำหรับเชื่อมต่อ กับเครือข่ายอินเทอร์เน็ตได้ หรือถ้าหากต้องการเชื่อมต่อผ่าน WiFi ก็ทำได้โดยใช้การ์ด WiFi ที่ทิจูดเชื่อมต่อแบบ PCIe ส่วน Intel Edison จะมี WiFi ในตัว จึงเชื่อมต่อ กับเครือข่ายอินเทอร์เน็ต ในแบบไร้สาย ได้ โดยไม่ต้องเพิ่มอุปกรณ์ใดๆ แต่เนื่องจาก Intel Edison มีลักษณะเป็นโมดูลคอมพิวเตอร์ขนาดเล็ก การใช้งานในแบบ DIY จึงต้องพึ่งพาบอร์ดอินพุตเอาต์พุตที่มีซ็อกเก็ตสำหรับติดตั้ง โมดูล Intel Edison

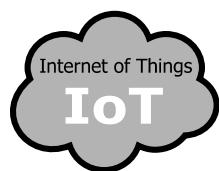


รูปที่ 1-6 หน้าตาของ Intel Galileo จะเห็นว่า ที่บอร์ดด้านบน (ภาพซ้าย) มีพอร์ตอินพุตเอาต์พุตสำหรับต่อ อุปกรณ์ภายนอก เมื่อนอกกับ Arduino UNO และมีพอร์ตอีเธอร์เน็ตสำหรับเชื่อมต่อ กับเครือข่ายอินเทอร์เน็ต อยู่ทางด้านซ้ายบน หากต้องการเชื่อมต่อแบบไร้สายผ่าน WiFi จะต้องติดตั้งการ์ด WiFi แบบ PCIe เข้าที่สล็อต PCIe ที่อยู่ด้านล่างของบอร์ด (ภาพขวา) ใกล้ๆ กลางบอร์ด ถัดมาทางขวาเล็กน้อย



รูปที่ 1-7 Intel Edison ที่ติดตั้งบนบอร์ด Arduino Brakout เพื่อให้ใช้งานได้ง่ายขึ้น เชื่อมต่อ กับเครือข่ายอินเทอร์เน็ตผ่าน WiFi ที่มีมาให้แล้วภายในตัว Intel Edison

ดังนั้น การพัฒนาอุปกรณ์ IoT จึงไม่ได้มีเพียงตัวเลือกเดียว หากแต่ผู้พัฒนาต้องมีความชัดเจนในคุณสมบัติทาง hardware เพื่อเลือกใช้งานให้เหมาะสม มีความเป็นไปได้ที่ในระบบหรือชุดอุปกรณ์ IoT อาจมีการผสมผสานหรือทำงานร่วมกันของ hardware ที่มีความแตกต่างกัน เช่น ส่วนอุปกรณ์ย่อยหรือระบบย่อยใช้ NodeMCU-12E เป็นตัวควบคุม โดยมี Raspberry Pi 2 เป็นเสมือน IoT เซิร์ฟเวอร์ เพื่อติดต่อกับระบบย่อยทั้งหมดผ่าน WiFi ก่อนนำข้อมูลจากระบบย่อยเหล่านั้นเข้าไปยังระบบคลาวด์ เนื่องจาก Raspberry Pi 2 มีความสามารถในการประมวลผลและขนาดของหน่วยความจำที่มากกว่า จึงทำหน้าที่เป็นตัวจัดการข้อมูลในขั้นต้นก่อนส่งเข้าระบบคลาวด์ เพื่อนำข้อมูลไปใช้ประโยชน์ต่อไป





## บทที่ 2

# แนะนำอุปกรณ์ IoT Education kit

---

เพื่อให้การเรียนรู้และพัฒนาอุปกรณ์ IoT เป็นไปได้อย่างสะดวกและเกิดผลสัมฤทธิ์ ทั้งกับผู้เริ่มต้นและผู้สนใจที่พอมีประสบการณ์ จึงขอแนะนำอุปกรณ์ทางชาร์ดแวร์ที่ใช้ในการสร้างงาน เพื่อแสดงให้เห็นถึงการทำงานจริง ซึ่งจะต้องทำงานควบคู่ไปกับการพัฒนาทางซอฟต์แวร์ โดยอุปกรณ์ดังกล่าว ผู้สนใจอาจจัดซื้อแยกเป็นรายการ หรือจะจัดหาแบบเป็นชุดก็ได้ ขึ้นอยู่กับความพร้อมด้านงบประมาณ หรือจะใช้แนวทางในหนังสือ นำไปปรับใช้กับอุปกรณ์ที่มีอยู่เดิมก็ได้ ทั้งนี้เนื่องจากหัวใจของอุปกรณ์ชาร์ดแวร์ในที่นี้คือ โมดูล NodeMCU-12E หรือ NodeMCU V2 หรือ NodeMCU Development Kit 1.0 ขึ้นกับการเรียกของผู้ผลิตและจำหน่าย โดยมีผู้จำหน่ายหลายรายในประเทศไทย หรือจะสั่งซื้อจากร้านค้าออนไลน์ทางอินเทอร์เน็ตก็ได้

## 2.1 IoT Education kit - NodeMCU ชุดเรียนรู้และพัฒนาอุปกรณ์ IoT ด้วย NodeMCU

ชุดเรียนรู้และพัฒนาอุปกรณ์ IoT ด้วย NodeMCU จัดทำโดยบริษัท อินโนเวติฟ เอ็กเพอริเม้นต์ จำกัด ([www.inex.co.th](http://www.inex.co.th)) ประกอบด้วยอุปกรณ์ต่างๆ ดังนี้

1. **NodeMCU-12E** มินิบอร์ดไมโครคอนโทรลเลอร์ 32 บิตที่มี WiFi ในตัว บางครั้งเรียกว่า NodeMCU V2 หรือ Development Kit V1.0 ขึ้นกับผู้ผลิตแต่ละราย โดยมีพื้นฐานมาจากโมดูล WiFi คอนโทรลเลอร์ ESP8266-12E
2. **AX-NodeMCU** บอร์ดอินพุตเอาต์พุตสำหรับ NodeMCU-12E
3. **ZX-LED** บอร์ดขับ LED 8 มม. 3 ชุด ประกอบด้วย LED สีแดง, เหลือง และเขียว
4. **ZX-LED3CS** บอร์ดขับ LED 3 สี RGB
5. **ZX-SPEAKER** บอร์ดขับลำโพงเปียโซ
6. **ZX-SWITCH01** บอร์ดสวิตช์อินพุต 2 ชุด
7. **ZX-DHT11** โมดูลตัวตรวจจับความชื้นสัมพัทธ์และอุณหภูมิ
8. **HC-SR04** โมดูลวัดระยะทางด้วยอัลตร้าโซนิก

9. ZX-BH1750 โมดูลวัดความเข้มแสง

10. ZX-SSR01 บอร์ดโซลิเดเตอร์รีเลย์ 1 ช่อง พร้อมสาย

11. I2C-LCD16x2 โมดูล LCD 16 ตัวอักษร 2 บรรทัดแบบมีไฟส่องหลัง ใช้การติดต่อผ่านบัส I<sup>2</sup>C

12. อະడิโอเพดอร์ไฟตรง +6V 2A

13. สาย microB-USB สำหรับอัปโหลดโปรแกรมและเชื่อมต่อ กับคอมพิวเตอร์ผ่านพอร์ต USB

14. สายเชื่อมต่อและทดลองวงจรรุ่น IDC1MF และ IDC1FF แบบละ 10 เส้น (คละถี่)

15. สายเชื่อมต่อรุ่น JST3AA-8 จำนวน 10 เส้น

16. ไขควงปลายแยกขนาดเล็ก

17. USB แฟลชไดรฟ์บรรจุซอฟต์แวร์ ข้อมูลทางเทคนิคต่างๆ และตัวอย่างโปรแกรม

18. หนังสือเริ่มต้นชุดเรียนรู้และพัฒนาอุปกรณ์ IoT ด้วย NodeMCU (เล่มนี้)

## 2.2 กลุ่มของอุปกรณ์หลัก

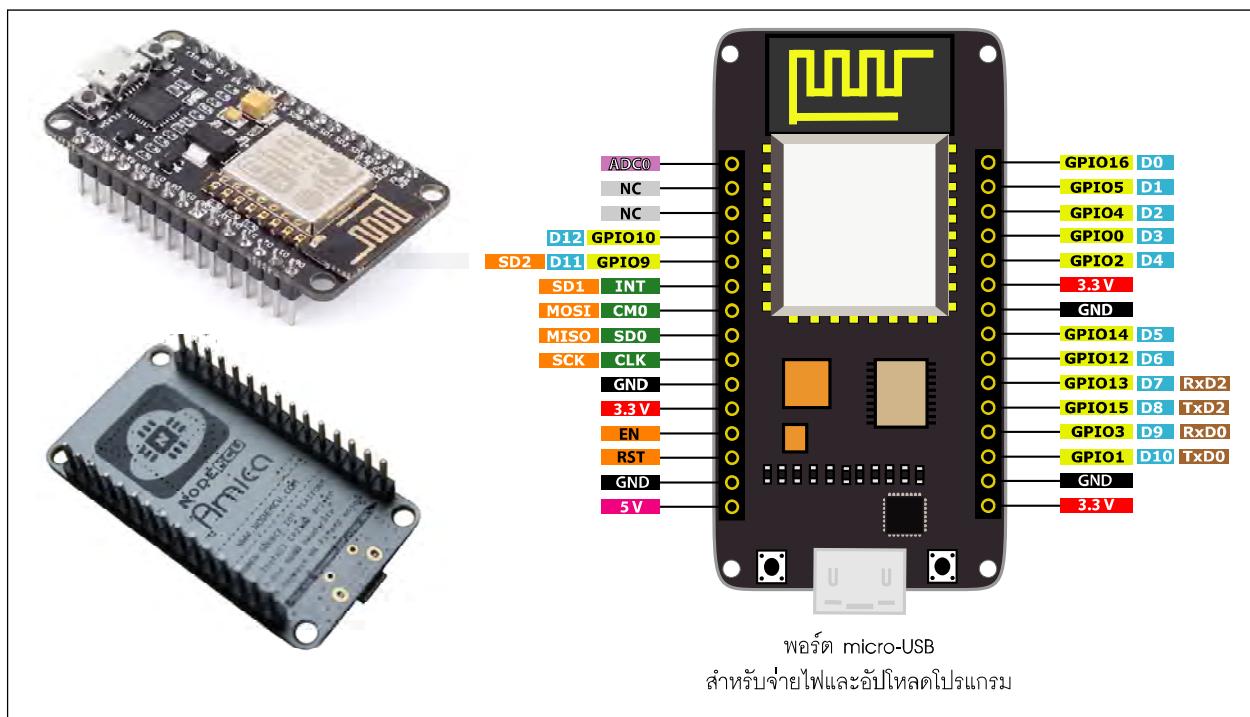
### 2.2.1 NodeMCU-12E โมดูลไมโครคอนโทรลเลอร์ 32 บิตพร้อม WiFi เพื่อการพัฒนาอุปกรณ์ IoT

โมดูล NodeMCU-12E หรือ V2 หรือ Development Kit V1.0 (ซึ่งที่แตกต่างนี้มาจากการเรียกของผู้ผลิต) นี้เป็นการนำโมดูล ESP8266-12E มาต่อร่วมกับชิปแปลงสัญญาณ USB เป็น UART เบอร์ CP2102 ของ Silcon Lab ([โปรดระวังของเลียนแบบจะใช้ชิปเบอร์ CH340](#)) มีสวิตช์เพื่อเข้าสู่โหมดโปรแกรมเฟิร์มแวร์มาพร้อม บรรจุรวมกันอยู่บนแผงวงจรขนาดเล็กที่ออกแบบมาให้ติดตั้งลงบนเบรเดบอร์ดหรือแผงต่อวงจร ได้โดยยังมีรูของเบรเดบอร์ดเหลือให้ต่อสายเพื่อเชื่อมต่อกับอุปกรณ์ภายนอกได้สะดวก ช่วยให้การพัฒนาต้นแบบและการเรียนรู้เกี่ยวกับ IoT ทำได้ง่ายขึ้น

คุณสมบัติทางเทคนิคที่สำคัญ มีดังนี้

- ใช้โมดูล ESP8266-12E ที่ภายในมีไมโครคอนโทรลเลอร์ 32 บิต หน่วยความจำแบบแฟลช ความจุ 4 เมกะไบต์และวงจร WiFi ในตัว

- มีชิป CP2102 สำหรับแปลงสัญญาณพอร์ต USB เป็น UART เพื่อเชื่อมต่อกับคอมพิวเตอร์สำหรับโปรแกรมเฟิร์มแวร์ ([ของเลียนแบบจะใช้ชิปเบอร์ CH340](#))



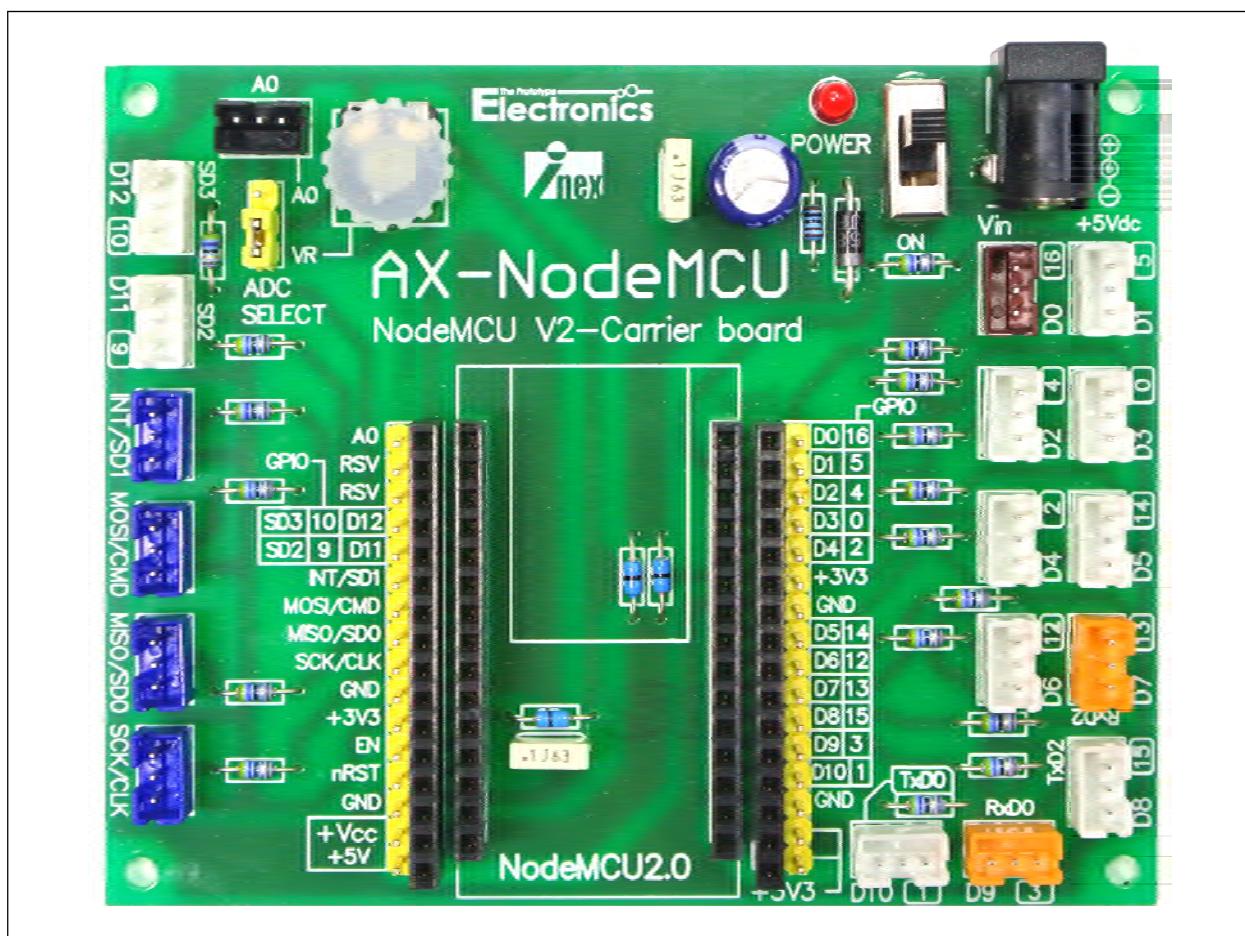
รูปที่ 2-1 หน้าตาของ NodeMCU-12E และการจัดขา

- ใช้ไฟเลี้ยงภายนอก +5V มีวงจรควบคุมแรงดันไฟเลี้ยงสำหรับอุปกรณ์ 3.3V กระแสไฟฟ้าสูงสุด 800mA
  - มีขาพอร์ต SPI สำหรับติดต่อกับ SD การ์ด
  - มีสวิตช์ RESET และ FLASH สำหรับโปรแกรมเฟิร์มแวร์ใหม่
  - มีอินพุตเอาต์พุตดิจิตอล (ลอจิก 3.3V) รวม 16 ขา
    - มีอินพุตอะนาล็อก 1 ช่อง รับแรงดันไฟตรง 0 ถึง +3.3Vdc เข้าสู่วงจรแปลงสัญญาณอะนาล็อก เป็นดิจิตอล ความละเอียด 10 บิต (ที่อินพุตมีวงจรแบ่งแรงดัน เนื่องจากอินพุตอะนาล็อกของ ESP8266-12E รับแรงดันได้เพียง 0 ถึง 1V จึงต้องมีการต่อตัวต้านทานเพื่อช่วยลดแรงดันลงจาก +3.3V ให้เหลือไม่เกิน 1.0V)
      - เสียบลงบนเบรเดบอร์ดเพื่อทำการทดสอบได้ทันที หรือนำໄไปติดตั้งบนแพงวงจรประยุกต์ที่ออกแบบขึ้นเองได้สะดวก

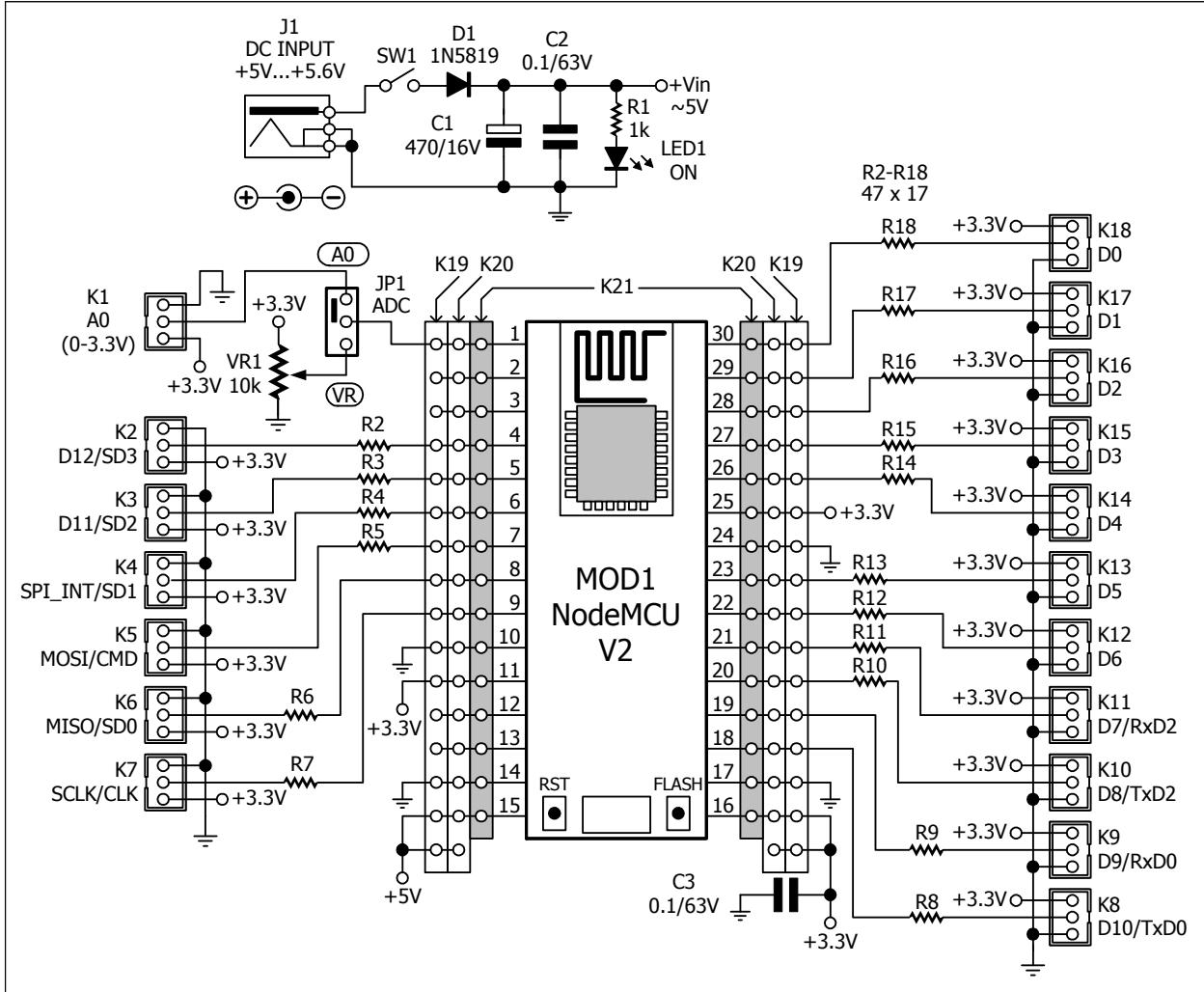
## 2.2.2 AX-NodeMCU บอร์ดอินพุตเอาต์พุตสำหรับทดลองและใช้งาน โมดูล NodeMCU-12E เพื่อการพัฒนาอุปกรณ์ IoT

ในการใช้งาน โมดูล NodeMCU-12E หรือ V2 หรือ Development Kit V1.0 ง่ายที่สุดก็เพียงเสียบโมดูลลงบนเบรเดบอร์ด แล้วต่อสายเข้ากับพอร์ต USB ของคอมพิวเตอร์ ก็จะทำการพัฒนาและอัปโหลดโปรแกรมได้แล้ว ด้านไฟเลี้ยงก็จะจากพอร์ต USB หากต้องการใช้งานแบบโดยล้ำพัง จะต้องจัดหาแหล่งจ่ายไฟภายนอกเพิ่มเติม ซึ่งที่หาได้ง่ายและสะดวกคือ เพาเวอร์แบงก์ (power bank) เนื่องจากให้แรงดัน +5V และใช้สาย Micro-USB เชื่อมต่อ

อย่างไรก็ตาม หากต้องการให้ โมดูล NodeMCU-12E ทำงานกับแพงวงจรตรวจจับและอุปกรณ์อินพุตเอาต์พุตของผู้ผลิตที่มีอยู่อย่างหลากหลาย การจัดการขาพอร์ตใหม่มีจุดต่อที่สะดวกต่อการเชื่อมต่อ ก็น่าจะเป็นทางเลือกที่ดี ในที่นี่แนะนำให้ใช้ AX-NodeMCU บอร์ดทดลองและเรียนรู้สำหรับ NodeMCU-12E



รูปที่ 2-2 บอร์ด AX-NodeMCU สำหรับติดตั้งและใช้งานโมดูล NodeMCU-12E



รูปที่ 2-3 วงจรของ AX-NodeMCU บอร์ดอินพุตเอาต์พุตสำหรับทดลองและใช้งานโมดูล NodeMCU-12E

### 2.2.2.1 คุณสมบัติทางเทคนิคที่สำคัญ

- มีช่องเก็บสำหรับติดตั้งโมดูล NodeMCU-12E หรือ V2 หรือ V1.0 Development kit
- มีจุดต่อพอร์ตอินพุตเอาต์พุตทั้งหมดของโมดูล NodeMCU-12E ในรูปแบบของคอนเนกเตอร์ JST 2.0 มม. ตัวผู้ และ IDC 2.54 มม. ทั้งตัวผู้และตัวเมีย ทำให้ใช้งานกับบอร์ดอินพุตเอาต์พุต และตัวตรวจจับได้ทุกรุ่น ทุกผู้ผลิต รวมถึงการใช้งานกับแพ็คต่อวงจรหรือเบรคบอร์ด
- พิมพ์ชื่อ, หมายเลข และฟังก์ชันการทำงานของพอร์ตต่างๆ ไว้อย่างชัดเจน
- มีตัวด้านหน้าปรับค่าได้ติดตั้งบนบอร์ดสำหรับทดสอบการทำงานของอินพุตอะนาล็อก ซึ่งใช้งานร่วมกับจุดต่ออินพุตอะนาล็อก A0 โดยมีจ้มเปอร์เลือกต่อใช้งาน
- มีจุดต่อไฟเลี้ยงจากภายนอกผ่านผ่านแจ็คอะแดปเตอร์ พร้อมสวิตช์เปิดปิด
- มี LED และสถานะไฟเลี้ยง
- มีโอดป้องกันการจ่ายไฟกลับข้าม และป้องกันแรงดันไฟเลี้ยงย้อนกลับหากต่อแหล่งจ่ายไฟภายนอกพร้อมกับต่อพอร์ต USB หากมีการต่อพอร์ต USB ไฟเลี้ยงโมดูล NodeMCU-12E จะรับจากพอร์ต USB เป็นหลัก

### 2.2.2.2 วงจรและการทำงาน

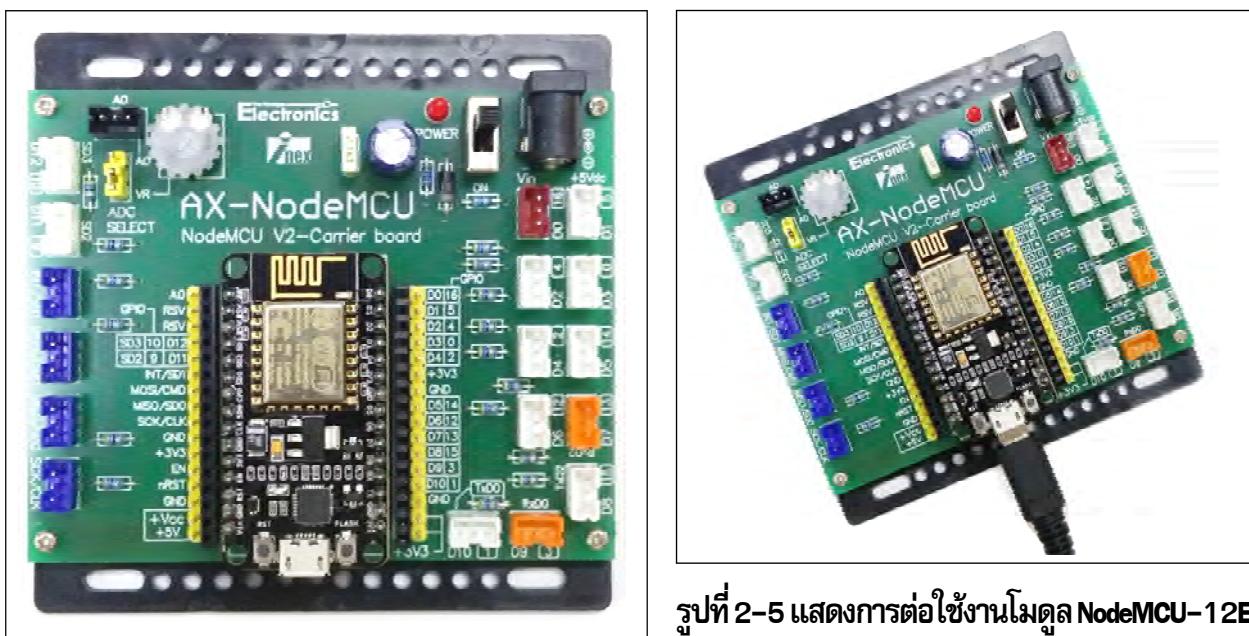
ในรูปที่ 2-3 แสดงวงจรของบอร์ด AX-NodeMCU เริ่มจากส่วนของไฟเลี้ยง มาได้จาก 2 ทางคือ ทาง J1 แจ็કอะแดปเตอร์ และจากพอร์ต USB ที่มีบันตัวโมดูล NodeMCU-12E ในกรณีที่จ่ายไฟผ่านทาง J1 แรงดัน +5V (สูงสุดไม่เกิน +5.6V) จะได้รับการตัดต่อเพื่อเข้าสู่วงจรด้วยสวิตช์ SW1 มีไอดิโอด D1 ต่อไว้เพื่อป้องกันการจ่ายไฟกลับข้าม และใช้ป้องกันไม่ให้แรงดัน +5V จากพอร์ต USB ข้อนกลับเข้าไปที่แหล่งจ่ายไฟภายนอกด้วย C1 และ C2 ช่วยลดสัญญาณรบกวน ส่วนการแสดงสถานะไฟเลี้ยงใช้ LED1

ซึ่งยกเก็ตสำหรับรองรับตัวโมดูล NodeMCU-12E คือ K21 ขาพอร์ตทั้งหมดของโมดูล NodeMCU จะถูกตัดต่อเข้ากับ K19 และ K20 รวมถึง K2 ถึง K18 ซึ่งเป็นคอนเนกเตอร์ JST 2.0 มม. 3 ขา โดยจัดสรรร่วมกับขาไฟเลี้ยง +3.3V และกราวด์ (GND) และมีตัวต้านทาน R2 ถึง R18 ต่ออนุกรมเพื่อจำกัดกระแสไฟฟ้าที่ไหลผ่านขาพอร์ต ลดโอกาสที่ขาพอร์ตจะเสียหายจากการต่อไฟเกินหรือลัดวงจร

ส่วนอินพุตของนาลอก A0 นั้นจะต่อเข้ากับ JP1 เพื่อเลือกใช้งานในแบบต่อกับแรงดันของนาลอกภายนอก หรือต่อกับแรงดันที่ได้จากตัวต้านทานปรับค่าได้ VR1 ที่มีบันแพรวงจร

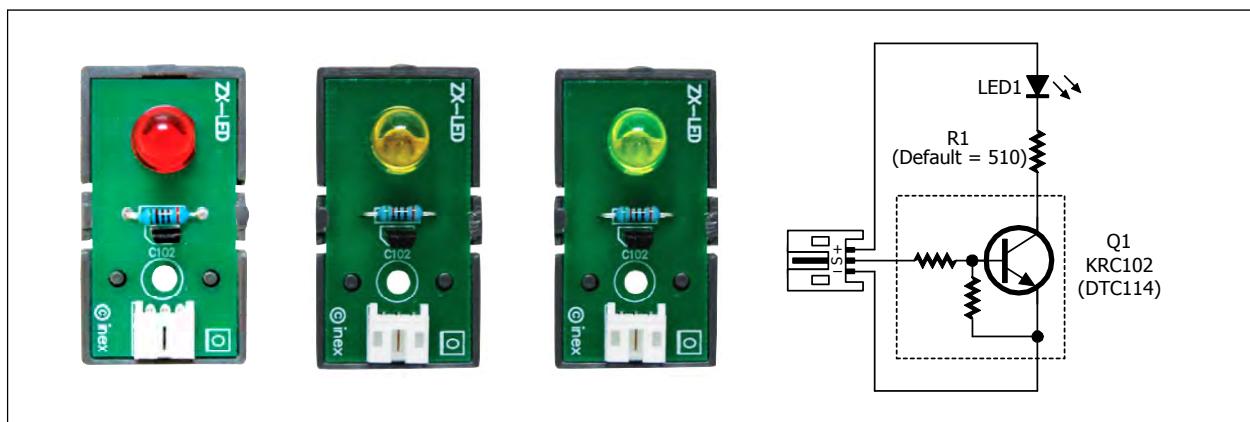
### 2.2.2.3 การนำไปใช้งาน

หากการใช้งานไม่มีอุปกรณ์ที่ต้องการกระแสไฟฟ้าสูง การใช้ไฟเลี้ยงจากพอร์ต USB นับเป็นทางเลือกที่สะดวก โดยติดตั้งโมดูล NodeMCU-12E แล้วต่อสายจากพอร์ต USB ของคอมพิวเตอร์ หากติดตั้งโปรแกรมและไดรเวอร์ไว้แล้ว ก็จะใช้งานทั้ง NodeMCU และบอร์ด AX-NodeMCU ได้ทันที



รูปที่ 2-4 แสดงบอร์ด AX-NodeMCU ที่ได้ติดตั้งโมดูล NodeMCU เรียบร้อย พร้อมใช้งาน

รูปที่ 2-5 แสดงการต่อใช้งานโมดูล NodeMCU-12E และบอร์ด AX-NodeMCU กับคอมพิวเตอร์ผ่านพอร์ต USB



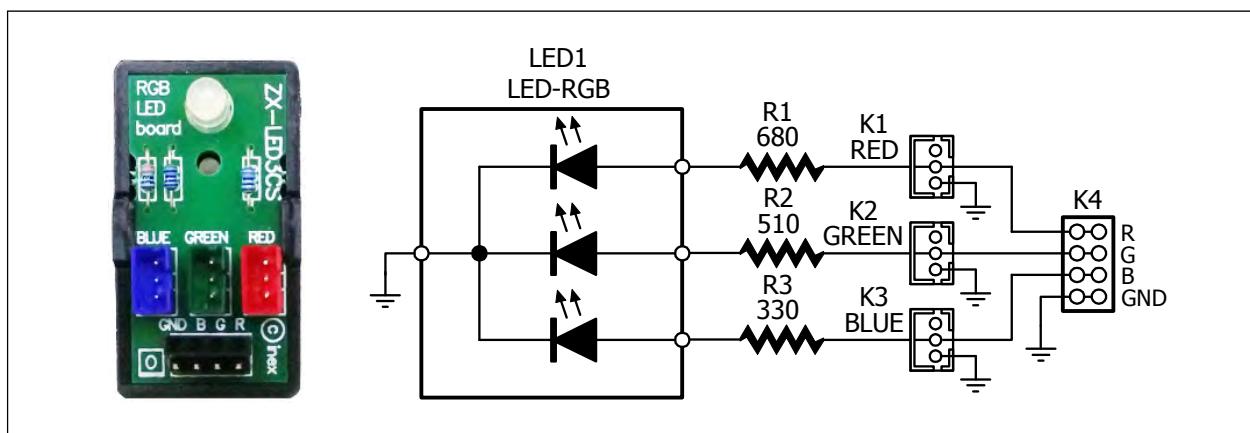
รูปที่ 2-6 รูปร่างและวงจรของ ZX-LED บอร์ดขั้บ LED

### 2.2.3 ZX-LED บอร์ดขั้บ LED

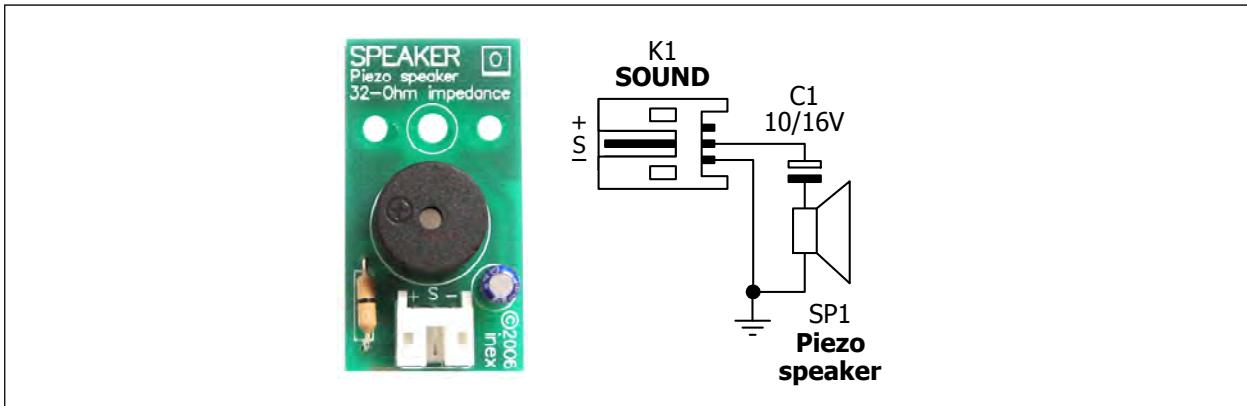
เป็นบอร์ดอุปกรณ์เอาต์พุต ใช้ขั้บ LED 8 มิลลิเมตร ต้องการลอกิก “1” ในการขั้บ LED ให้สว่าง มีวงจรแสดงในรูปที่ 2-6 ในชุด IoT Education kit มีบอร์ด ZX-LED 3 ชุด เป็นสีแดง, เหลือง และเขียว

### 2.2.4 ZX-LED3CS บอร์ดขั้บ LED 3 สี RGB

เป็นมินิบอร์ดอุปกรณ์เอาต์พุตสำหรับขั้บ LED 3 สีแบบ RGB (Red - สีแดง, Green - สีเขียว, Blue - สีน้ำเงิน ขนาด 5 มิลลิเมตร มีวงจรแสดงในรูปที่ 2-7 โดย LED1 ที่ใช้เป็น LED 3 สี RGB แบบแคลโตกิด ร่วม ตัวต้านทาน R1 ถึง R3 มีค่าแตกต่างกัน เพื่อให้ LED แต่ละสีภายใน LED1 ทำงานได้ใกล้เคียงกัน เมื่อได้รับแรงดันเท่าๆ กัน ZX-LED3CS จะถูกขั้บให้แสดงแสงสีต่างๆ ได้จากการป้อนแรงดันแก่ขา แอนด์แทลล์ของ LED 3 สี RGB



รูปที่ 2-7 วงจรของบอร์ด LED 3 สี : ZX-LED3CS



รูปที่ 2-8 วงจรของบอร์ดขับลำโพงเปียโซ ZX-SPEAKER

### 2.2.5 ZX-SPEAKER บอร์ดขับลำโพงเปียโซ

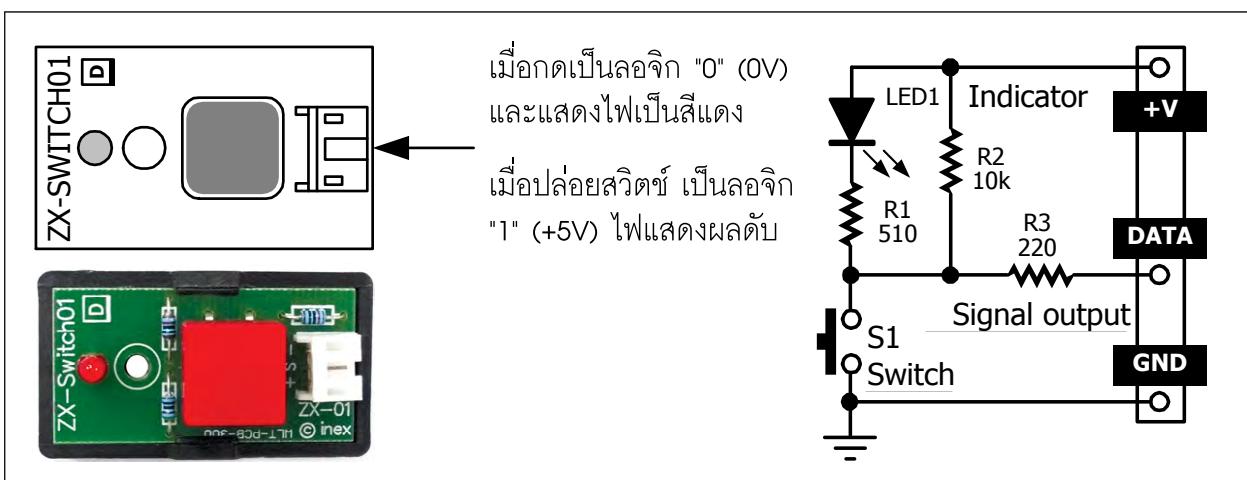
มีวงจรและหน้าตาของบอร์ดแสดงในรูปที่ 2-8 คุณสมบัติทางเทคนิคที่สำคัญมีดังนี้

- ใช้ลำโพงเปียโซ มีอิมพีเดนซ์  $32\Omega$
- มีค่าความถี่เรโซแนนซ์ในย่าน 1 ถึง  $3\text{kHz}$

### 2.2.6 ZX-SWITCH01 บอร์ดสวิตช์อินพุต 1 ช่อง

มีวงจรแสดงในรูปที่ 2-9 ประกอบด้วยสวิตช์พร้อมไฟแสดงผล ต้องการไฟเลี้ยงในย่าน  $+3$  ถึง  $+5\text{V}$  ใช้กระแสไฟฟ้า  $10\text{mA}$  ในการทำงาน เมื่อมีการกดสวิตช์

ให้อาดพุตคือ หากมีการกดสวิตช์ จะส่งลอจิก “0” (ระดับแรงดัน  $0\text{V}$ ) และไฟสีแดง



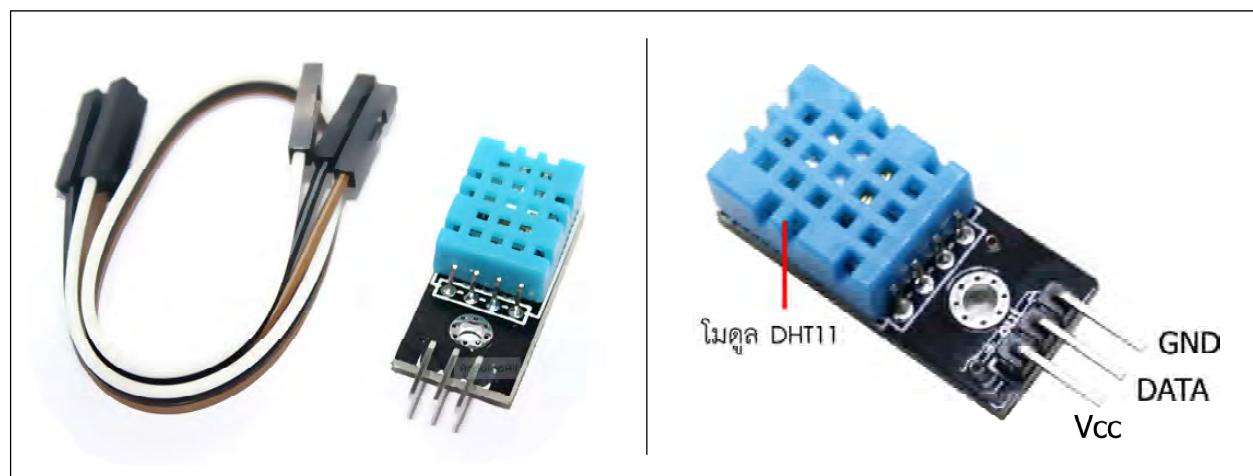
รูปที่ 2-9 รูปร่างและวงจรของบอร์ดสวิตช์อินพุต 1 ช่อง (ZX-SWITCH01)

## 2.2.7 ZX-DHT11 บอร์ดวัดความชื้นสัมพัทธ์และอุณหภูมิ

ZX-DHT11 เป็นแพ็คเกจที่บรรจุโมดูลตรวจจับและวัดความชื้นสัมพัทธ์เบอร์ DHT11 ซึ่งนอกจากจะวัดความชื้นสัมพัทธ์ได้แล้ว ยังให้ค่าของอุณหภูมิของพื้นที่ที่ตรวจวัดความชื้นด้วย การติดต่อเป็นแบบหนึ่งสาย นั่นคือใช้ขาพอร์ตของไมโครคอนโทรลเลอร์ เพียง 1 หนึ่งขาในการทำงาน ในรูปที่ 2-10 แสดงหน้าตาของ ZX-DHT11 และการจัดขา

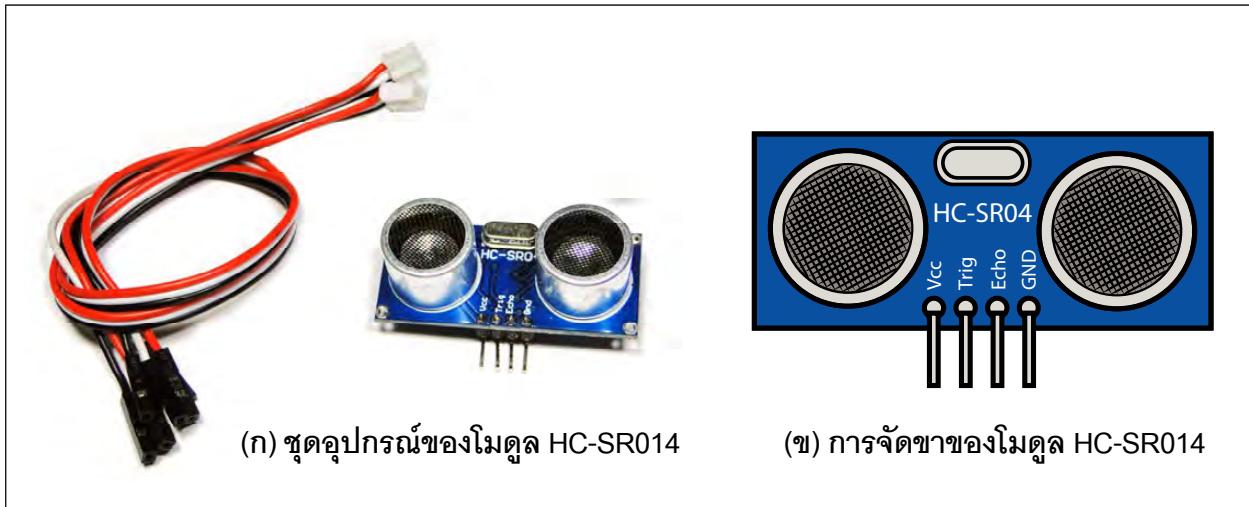
ZX-DHT11 มีคุณสมบัติทางเทคนิคที่ควรทราบเพื่อเป็นข้อมูลประกอบในการใช้งานดังนี้

- ใช้โมดูล DHT11 ติดตั้งบนแผ่นวงจรพิมพ์
- มีตัวต้านทานต่อพูลอปทิ่ม DATA ทำให้เชื่อมต่อกับขาพอร์ตของไมโครคอนโทรลเลอร์ ได้โดยไม่ต้องต่อตัวต้านทานเพิ่ม
  - ใช้ไฟเลี้ยง +3 ถึง +5.5V ต้องการกระแสไฟฟ้า 2.5mA ในขณะทำการวัดค่า และ 0.5mA ในโหมดสตีป
  - วัดความชื้นสัมพัทธ์ได้ 20 ถึง 80%RH มีความผิดพลาด  $\pm 5\%$ RH และมีความละเอียดในการวัด 1 % ขนาดของข้อมูล 8 บิต
  - วัดอุณหภูมิได้ 0 ถึง 50 องศาเซลเซียส มีความผิดพลาด  $\pm 2$  องศาเซลเซียส ความละเอียดในการวัด 1 องศาเซลเซียส ขนาดของข้อมูล 8 บิต
  - อัตราการสุ่มวัด 1 วินาที
  - ความเร็วในการตอบสนองต่อการเปลี่ยนแปลงในการวัด 6 ถึง 30 วินาที
  - ขนาด 12 x 28 มิลลิเมตร



รูปที่ 2-10 หน้าตาและการจัดขาของ ZX-DHT11 บอร์ดวัดความชื้นสัมพัทธ์และอุณหภูมิ

**หมายเหตุ :** มีผู้ผลิตแพ็คเกจที่ใช้โมดูล DHT11 หลายราย อาจมีการจัดขาที่ต่างไปจากนี้ ดังนั้น จึงควรตรวจสอบตำแหน่งขาให้ถูกต้องก่อนเชื่อมต่อเพื่อใช้งาน



รูปที่ 2-11 หน้าตาและการจัดขาของ HC-SR04 โมดูลวัดระยะทางด้วยคลื่นอัลตร้าโซนิก

## 2.2.8 HC-SR04 โมดูลวัดระยะทางด้วยคลื่นอัลตร้าโซนิกรุ่นประยุกต์

HC-SR04 เป็นโมดูลวัดระยะทางที่ใช้หลักการสะท้อนของคลื่นอัลตร้าโซนิก ประกอบด้วยตัวกำเนิดคลื่นอัลตร้าโซนิกทำหน้าที่ส่งคลื่นออกไปสะท้อนกับวัตถุที่อยู่ข้างหน้ากลับมาบันทึกตัวรับสัญญาณ โดยระยะทางที่วัดได้จะสัมพันธ์กับระยะเวลาที่คลื่นอัลตร้าโซนิกเคลื่อนที่ไปบรรจบวัตถุ และสะท้อนกลับมาบันทึกตัวรับ เมื่อรู้ระยะเวลาที่คลื่นอัลตร้าโซนิกสะท้อนกลับมา จึงนำมาคำนวณหาเป็นระยะทางระหว่างโมดูล HC-SR04 กับวัตถุได้

โมดูล HC-SR04 วัดระยะทางได้ถูกต้องในช่วง 2 ถึง 200 ซม. (2 เมตร) มีความละเอียดอยู่ที่ 0.3 ซม. ใช้ไฟเลี้ยง +5V

การเชื่อมต่อ กับไมโครคอนโทรลเลอร์ของโมดูล HC-SR04 ใช้ขาพอร์ต 2 ขา ขาหนึ่งทำหน้าที่เป็นเอาต์พุตส่งสัญญาณมายังขา Trig เพื่อกระตุนให้โมดูล HC-SR04 ทำงาน ส่วนอีกขาหนึ่งทำหน้าที่เป็นอินพุต รับสัญญาณจากขา Echo ของโมดูล HC-SR04 เพื่ออ่านค่าสัญญาณพัลส์ จากนั้นนำไปคำนวณเป็นค่าระยะทางกลับออกมา

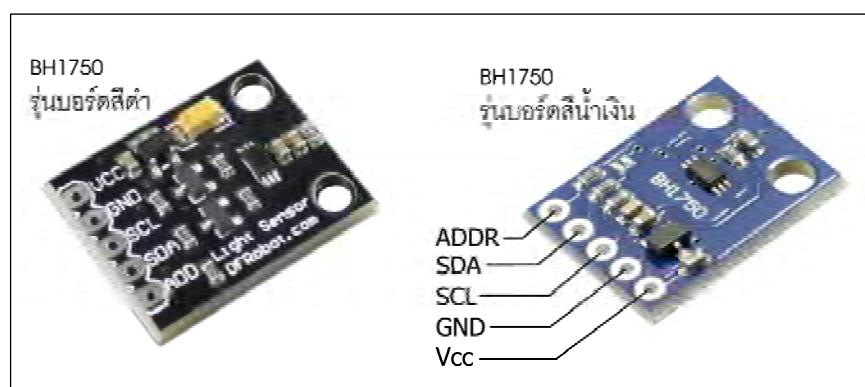
## 2.2.9 ZX-BH1750 บอร์ดวัดความเข้มแสงผ่านบัส I<sup>2</sup>C

เป็นแพ็คเกจที่ติดตั้งตัวตรวจจับแสงเบอร์ BH1750 โดย BH1750 เป็นผลงานของ ROHM Semiconductor ([www.rohm.com](http://www.rohm.com)) ผู้ผลิตอุปกรณ์สารกึ่งตัวนำขั้นนำของโลก BH1750 นับเป็นตัวตรวจจับแสงที่มีประสิทธิภาพสูง ใช้งานง่าย ด้วยการติดต่อผ่านบัส 2 สายหรือ I<sup>2</sup>C ให้ผลการวัดความเข้มแสงเป็นหน่วยลักซ์ (Lux) ทำให้นำข้อมูลที่ได้ไปใช้ประโยชน์ต่อได้ทันที โดยไม่ต้องพึงกระบวนการทางคณิตศาสตร์เพื่อแปลงหน่วย ภายในตัวตรวจจับมีวงจรแปลงสัญญาณอะนาลอกเป็นดิจิตอลความละเอียด 16 บิตทำให้ได้ข้อมูลดิจิตอลของความเข้มแสงที่มีความละเอียดและแม่นยำมาก พอดำรงการนำไปสร้างเครื่องวัดความเข้มแสงหรือลักซ์มิเตอร์ (Luxmeter)

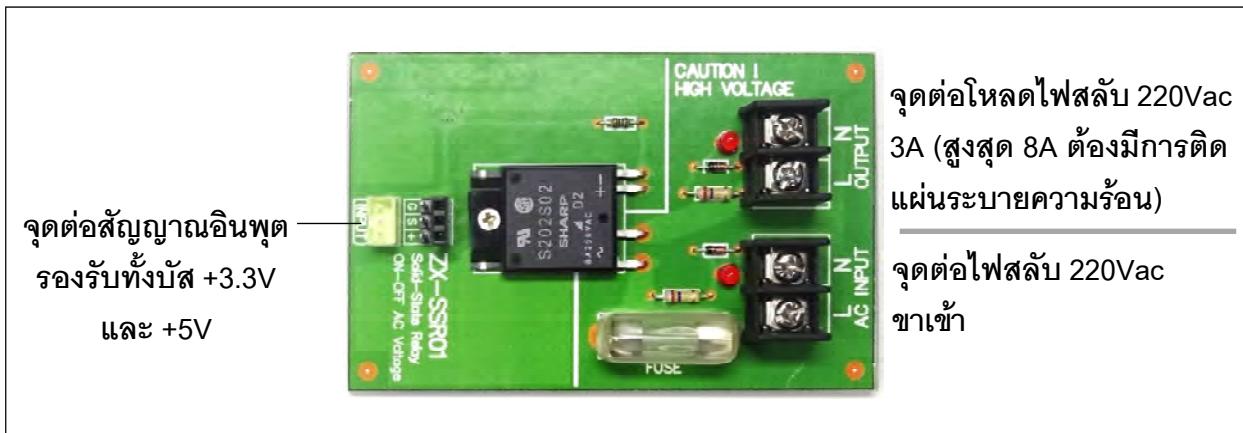
คุณสมบัติทางเทคนิคที่ควรทราบของบอร์ดวัดความเข้มแสง BH1750 มีดังนี้

- ติดตั้งตัวตรวจจับแสงเบอร์ BH1750 บนบอร์ดภายในมีตัวรับแสงเป็นไฟโตไดโอดต่อร่วมกับวงจรขยายสัญญาณ, วงจรแปลงสัญญาณอะนาลอกเป็นดิจิตอล และวงจรเชื่อมต่อระบบบัส I<sup>2</sup>C
- มีตัวต้านทานต่อพูลอัปที่ขาเชื่อมต่อนบัส I<sup>2</sup>C ไว้พร้อม ทำให้มีอนามัยเชื่อมต่อกับไมโครคอนโทรลเลอร์ทำได้ทันที โดยไม่ต้องต่อตัวต้านทานเพิ่ม
- ใช้ไฟเลี้ยง +3 ถึง +5Vdc กินกระแสไฟฟ้าต่ำมาก ประมาณ 200μA เท่านั้น
- ย่านวัดความเข้มแสง 1 ถึง 65,535 ลักซ์ มีค่าความผิดพลาด 20%
- กำหนดแอดเดรสให้กับ BH1750 ได้ 2 รูปแบบผ่านทางขา ADDR
- ทนต่อการรบกวนจากแสงอินฟราเรด
- ขนาด 21 × 16 มม.

รูปที่ 2-12 แสดงหน้าตาและการจัดขาของบอร์ดวัดความเข้มแสงที่ใช้ตัวตรวจจับเบอร์ BH1750 จะเห็นได้ว่า มีการผลิตออกมา 2 แบบ มีการจัดขาสัญญาณสลับกันเล็กน้อย ดังนี้ เมื่อนำมาต่อใช้งานควรตรวจสอบตำแหน่งขาให้ถูกต้องก่อน



รูปที่ 2-12 หน้าตาและการจัดขาของบอร์ดวัดความเข้มแสง BH1750 มีการผลิตออกมากำหนด 2 รูปแบบหลัก



รูปที่ 2-13 รูป่างหน้าตาของ ZX-SSR01 บอร์ดขับโซลิดสเตต里的เลย์

### 2.2.10 ZX-SSR01 บอร์ดขับโซลิดสเตต里的เลย์ 1 ช่อง

เป็นบอร์ดสำหรับเปิดปิดอุปกรณ์ไฟฟ้ากระแสสลับด้วยการควบคุมจากสัญญาณอิจิกจากไมโครคอนโทรลเลอร์หรือวงจรอิเล็กทรอนิกส์ตัวใดก็ได้ โดยอุปกรณ์ที่เป็นหัวใจหลักก็คือ โซลิดสเตต里的เลย์ (Solid State Relay - SSR) เบอร์ S202S02 ของ Sharp Semiconductor

โซลิดสเตต里的เลย์คือรีเลย์ที่ไม่มีการเคลื่อนไหวของกลไก วงจรภายในเป็นอุปกรณ์เชิงมيكอนดัก เตอร์ทั้งหมด ตัดแยกแรงดันไฟต่ำและไฟสูงออกจากกันอย่างเด็ดขาด โดยส่งสัญญาณควบคุมผ่านสายแกน

คุณสมบัติของบอร์ด ZX-SSR01 ที่สำคัญมีดังนี้

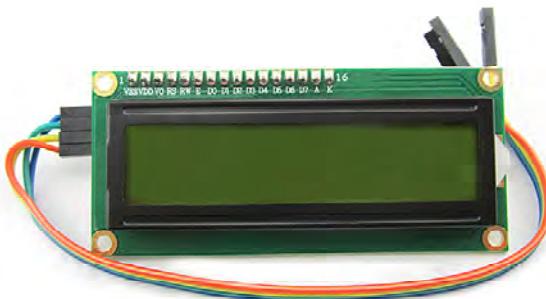
- ควบคุมอุปกรณ์ไฟฟ้า 220V กำลังสูงสุด 600W
- ใช้สัญญาณการเปิด/ปิดคือวัลวอลจิก 0V และ 3.3V ถึง 5V จึงใช้กับไมโครคอนโทรลเลอร์ได้ทั้งแบบบัสแรงดัน +3.3V และ +5V
- มี LED แสดงสถานะการทำงานของโซลิดสเตต里的เลย์ และไฟสลับขาเข้า 220Vac

## 2.2.11 I2C-LCD16x2 โมดูล LCD 16 ตัวอักษร 2 บรรทัดแบบมีไฟส่องหลัง ติดต่อผ่านบัส I<sup>2</sup>C

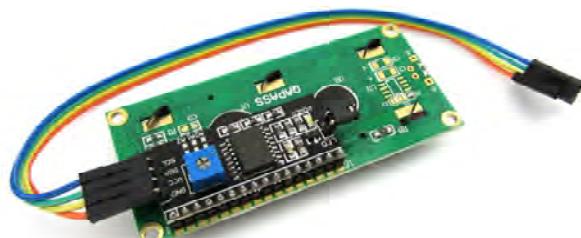
เป็นอุปกรณ์แสดงผลที่ใช้โมดูล LCD 16 ตัวอักษร 2 บรรทัด แบบมีไฟส่องหลัง ใช้แสดงตัวอักษร ตัวเลข สัญลักษณ์ และข้อความ ติดต่อผ่านบัสสองสายในแบบ I<sup>2</sup>C จึงใช้ข้าพอร์ตของไมโครคอนโทรลเลอร์ในการติดต่อเพียง 2 ขาจากปกติท้องใช้อย่างน้อย 6 ขา มีหน้าตาแสดงในรูปที่ 2-14

คุณสมบัติทางเทคนิคที่สำคัญมีดังนี้

- แสดงผลได้ 16 ตัวอักษร 2 บรรทัด
- ต่อ กับพอร์ตของไมโครคอนโทรลเลอร์และโมดูล NodeMCU-12E ได้โดยตรง โดยใช้ข้าพอร์ต 2 ขา ติดต่อในลักษณะบัส I<sup>2</sup>C
- แอดเดรสบัส I<sup>2</sup>C มี 2 ค่า แยกตามเบอร์ของไอซีที่ใช้ในการเชื่อมต่อบัส I<sup>2</sup>C คือ 0x27 สำหรับเบอร์ PCF8574 และ 0x3F สำหรับเบอร์ PCF8574A
- ใช้ชุดคำสั่งควบคุมเหมือนกับโมดูล LCD มาตรฐานที่ใช้ตัวควบคุมเบอร์ HD44780 หรือเทียบเท่า
  - ใช้สายต่อ 4 เส้น รวมไฟเลี้ยง ประกอบด้วย Vcc (+), GND (G), SDA และ SCL
  - ใช้ไฟเลี้ยง +5V



(ก) ภาพด้านหน้าของโมดูล LCD แบบบัส I<sup>2</sup>C



(ข) ภาพด้านหลังของโมดูล LCD แบบบัส I<sup>2</sup>C แสดงให้เห็นถึงบอร์ดเชื่อมต่อบัส I<sup>2</sup>C ที่ใช้ไอซีเบอร์ PCF8574 หรือ PCF8574A รวมถึงตัวต้านทานปรับค่าได้สำหรับปรับความชัดเจนในการแสดงผล

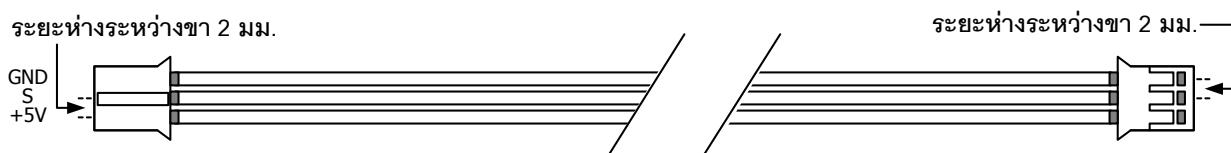
รูปที่ 2-14 รูป่างหน้าตาของ I2C-LCD16x2 โมดูล LCD 16 ตัวอักษร 2 บรรทัดติดต่อผ่านบัส I<sup>2</sup>C

## 2.3 ข้อมูลของสายสัญญาณที่ใช้ในชุดกล่องสมองกล IPST-MicroBOX\_(SE)

### 2.3.1 สาย JST3AA-8 : สายเชื่อมต่อระหว่างบอร์ดแบบหัวต่อ JST



สาย JST3AA-8 ใช้เชื่อมต่อระหว่างบอร์ด AX-NodeMCU กับบอร์ดอุปกรณ์ตรวจจับและบอร์ดแพงวงจรอุปกรณ์ต่างๆ เป็นสายแพ 3 เส้น ยาว 8 นิ้ว ปลายสายทั้งสองด้านติดตั้งคอนเนกเตอร์แบบ JST 3 ขา ตัวเมีย ระยะห่างระหว่างขา 2 มิลลิเมตร มีการจัดขาดังนี้



### 2.3.2 สาย IDC1MF/1FF : สายเชื่อมต่อระหว่างบอร์ดแบบหัวต่อ IDC

เป็นสายสัญญาณสำหรับเชื่อมต่อระหว่างชุดต่อขาดของ NodeMCU ที่ใช้หัวต่อแบบ IDC ทั้งตัวผู้และตัวเมีย ยาว 25 ซม. มีระยะห่างของแต่ละขาคือ 2.54 มิลลิเมตร โดยแบ่งเป็นสายที่ปลายด้านหนึ่งเป็นหัวต่อ IDC ตัวผู้ ปลายอีกด้านหนึ่งเป็นหัวต่อตัวเมีย (IDC1MF) และแบบปลายทั้งสองด้านเป็นหัวต่อตัวเมีย (IDC1FF) ในชุดมีแบบละ 10 เส้นคละสี



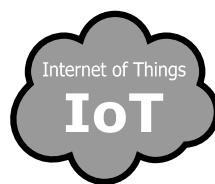
### 2.3.3 สาย microB-USB

เป็นสายสัญญาณสำหรับเชื่อมต่อระหว่างพอร์ต USB ของคอมพิวเตอร์กับโ้มดูล NodeMCU-12E ยาว 1.5 เมตร โดยประมาณ



### 2.3.4 อะแดปเตอร์ไฟตรง 5V 2A

ในชุด IoT Education kit มีอะแดปเตอร์ไฟตรง 5V 2A มาพร้อมใช้งาน ต่อกับบอร์ด AX-NODEMCU ได้ทันที





## บทที่ 3

# การพัฒนาโปรแกรมภาษา C/C++ สำหรับ NodeMCU ด้วย Arduino IDE

---

การพัฒนาโปรแกรมเพื่อใช้งาน NodeMCU ทำได้ด้วยโปรแกรมภาษา Lua และ C/C++ สำหรับในที่นี่เลือกใช้โปรแกรมภาษา C/C++ โดยใช้เครื่องมือพัฒนาที่ได้รับความนิยมสูงนั่นคือ Arduino IDE โดยมีนักพัฒนาอิสระชื่อ **Christian Klippe** ได้เริ่มต้นพัฒนาเครื่องมือที่ชื่อว่า *Esptool* โดยใช้ข้อมูลจาก *Espressif SDK* ที่พัฒนาโดย *Espressif* ผู้ผลิตโมดูล ESP8266 จากนั้น **Ivan Grokhotkov** ชาวสหราชอาณาจักร เผด็จศึกษาต่อในมหาวิทยาลัยการเพิ่มคอมไฟเลอร์สำหรับ NodeMCU ลงใน Arduino IDE ก่อให้เกิดเป็น Arduino IDE รุ่นพิเศษ มีข้อมูลและโปรแกรมให้ดาวน์โหลดที่ <https://github.com/esp8266/arduino> โดยมีขั้นตอนตามปกติคือ ติดตั้ง Arduino IDE เวอร์ชัน 1.6.xx (แนะนำเวอร์ชัน 1.6.4.xxx) ก่อน จากนั้นจึงดาวน์โหลดไฟล์สำหรับ ESP8266 ซึ่งก็คือ อุปกรณ์หลักของ NodeMCU ซึ่งในขั้นตอนนี้ต้องทำการเชื่อมต่อกับเว็บไซต์ของผู้พัฒนาและดาวน์โหลดโปรแกรมลงมายังเครื่องของ Arduin IDE โดยปกติจะใช้เวลากันพอสมควร นอกจากนี้ในขั้นตอนการดาวน์โหลดไฟล์และไฟล์ที่ต้องมีการแก้ไขไฟล์ภายในเดือนน้อย จึงจะใช้งาน Arduino IDE ในการพัฒนาโปรแกรมให้แก่ NodeMCU หรือโมดูล ESP8266 ทุกรุ่นได้

### License and credits

- Arduino IDE is developed and maintained by the Arduino team. The IDE is licensed under GPL.
- ESP8266 core includes an xtensa gcc toolchain, which is also under GPL.
- Esptool written by Christian Klippe is licensed under GPLv2, currently maintained by Ivan Grokhotkov: <https://github.com/igrr/esptool-ck>.
- Espressif SDK included in this build is under Espressif MIT License.
- ESP8266 core files are licensed under LGPL.
- SPI Flash File System (SPIFFS) written by Peter Andersson is used in this project. It is distributed under MIT license



**Ivan Grokhotkov** ผู้พัฒนา  
Arduino IDE สำหรับโมดูล  
**ESP8266** (ภาพจาก  
<https://github.com/igrr>)

### 3.1 Arduino IDE 1.6.5R2 for ESP8266/NodeMCU

อย่างไรก็ตาม เพื่อให้เกิดความสะดวกมากขึ้น วิศวกรของบริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด หรือ INEX ได้ทำการสร้างไฟล์ติดตั้งซอฟต์แวร์ Arduino IDE for ESP8266/NodeMCU ขึ้นมาใหม่ เป็นไฟล์ *Arduino1.6.5r5\_Setup151116.exe* (เลขเวอร์ชันอาจเปลี่ยนแปลงได้)

โดยตัดขั้นตอนการผนวกไฟล์และแก้ไขไฟล์องค์ประกอบหลังจากการติดตั้ง ทำให้การติดตั้งโปรแกรมง่าย เหมือนกับการติดตั้งโปรแกรมประยุกต์ทั่วไป นั่นคือ ดับเบิลคลิกไฟล์ติดตั้ง คลิกปุ่ม เพื่อตอบรับการติดตั้งโปรแกรม รอจนกว่าทั้งการติดตั้งเสร็จสมบูรณ์ ก็จะใช้งานได้ทันที

Arduino IDE for ESP8266/NodeMCU ที่ INEX จัดทำขึ้น ดาวน์โหลดได้โดยไม่มีค่าใช้จ่ายที่ [www.inex.co.th](http://www.inex.co.th) หรือติดตั้งจาก USB แฟลชไดรฟ์ที่มากับชุด IoT Education Kit - NodeMCU

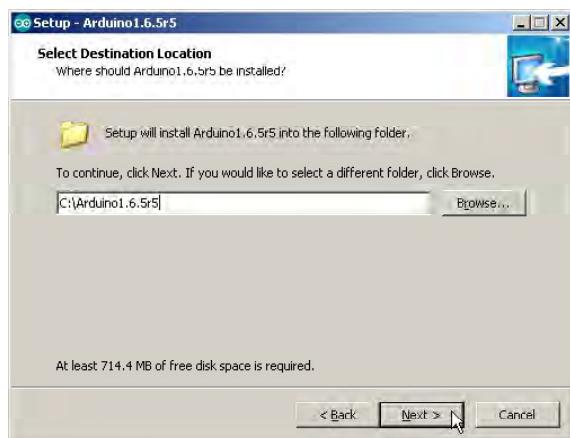
### 3.2 ติดตั้งโปรแกรมและไดรเวอร์

(1) ดาวน์โหลดไฟล์ติดตั้ง *Arduino1.6.5r5\_Setup151116.exe*

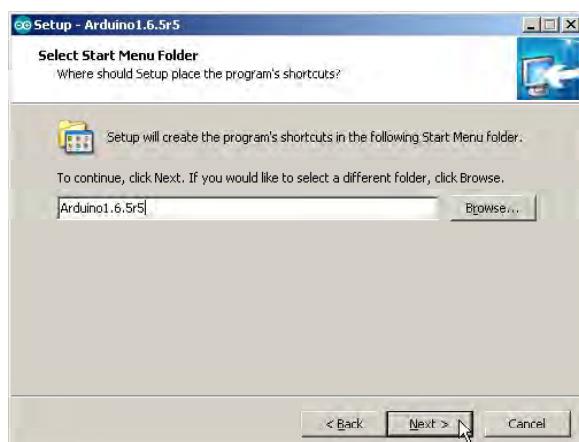
(2) ดับเบิลคลิกเพื่อสั่งให้ไฟล์ติดตั้งทำงาน จะปรากฏข้อความต้อนรับการติดตั้งโปรแกรม คลิกปุ่ม Next เพื่อไปยังขั้นตอนต่อไป



(3) เลือกไฟล์เดอร์ปลายทางที่ต้องการจัดเก็บไฟล์ที่เกี่ยวข้องของโปรแกรม คลิกปุ่ม Next



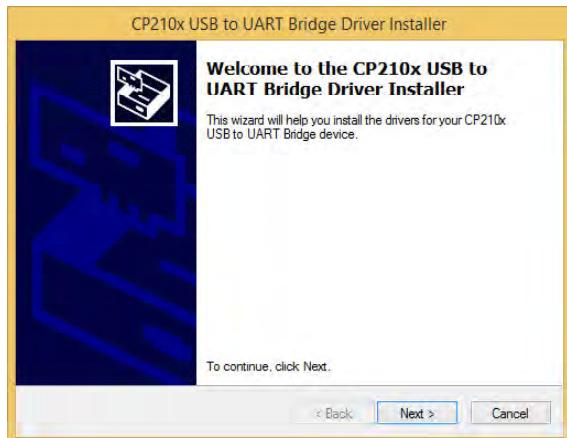
(4) เลือกไฟล์เดอร์ที่ต้องการใน Start Menu คลิกปุ่ม Next เพื่อไปยังขั้นตอนต่อไป



(5) คลิกปุ่ม Install เพื่อเริ่มต้นการติดตั้งโปรแกรม



(6) ขั้นตอนต่อมา เป็นการติดตั้ง ไดรเวอร์ของอุปกรณ์ที่เชื่อมต่อผ่านพอร์ต USB คลิกปุ่ม **Next** เพื่อเข้าสู่การติดตั้ง ไดรเวอร์



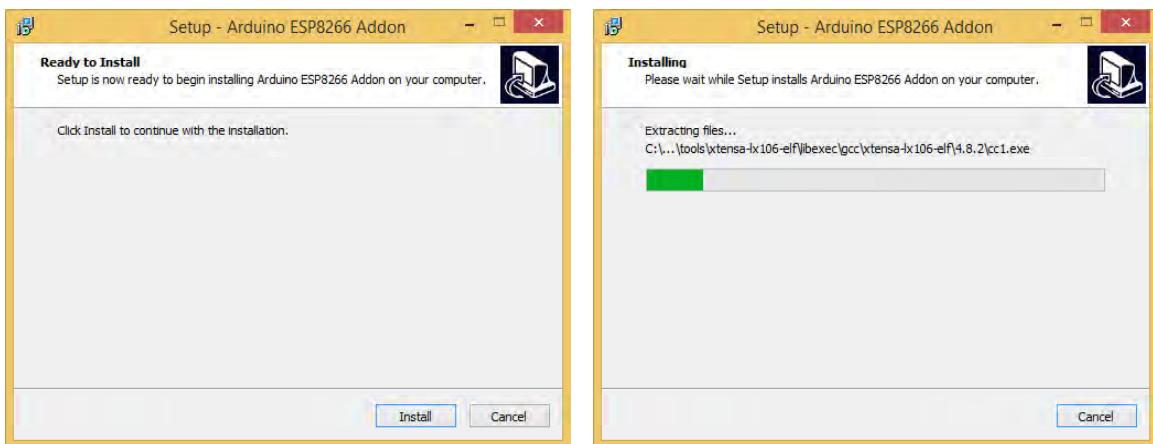
(7) คลิกเลือก **I accept this agreement** เพื่อยอมรับข้อตกลงค้านลิขสิทธิ์ จากนั้นคลิกปุ่ม **Next**



(8) คลิกปุ่ม **Finish** เพื่อสิ้นสุดการติดตั้ง ไดรเวอร์



(9) จากนั้นจะเข้าสู่การติดตั้ง Tool chain สำหรับ ESP8266/NodeMCU ให้กับ Arduino IDE 1.6.5 ให้คลิกปุ่ม **Install** เพื่อทำการติดตั้ง

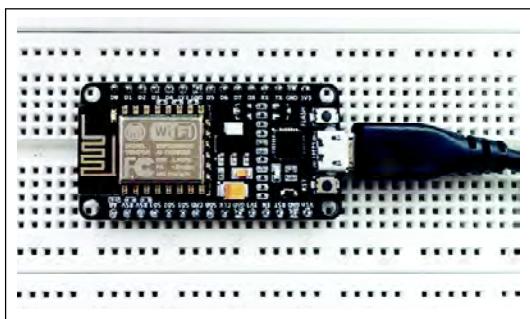


(10) รอนการทั้งการติดตั้งเสร็จสมบูรณ์ จะได้ ArduinoIDE ที่พร้อมสำหรับการพัฒนาโปรแกรมให้กับโมดูล ESP8266 และ NodeMCU

### 3.3 ทดสอบโปรแกรมเบื้องต้น

(1) เชื่อมต่อโมดูล NodeMCU กับพอร์ต USB โดย

(A) หากใช้โมดูล NodeMCU-12E กับเบรเดอร์ด ให้เสียบโมดูล NodeMCU-12E ลงบนเบรเดอร์ด ดังรูปที่ 3-1 จากนั้นต่อสาย microB-USB เข้ากับโมดูล NodeMCU-12E และพอร์ต USB ของคอมพิวเตอร์ รอสักครู่เพื่อให้การเชื่อมต่อสมบูรณ์



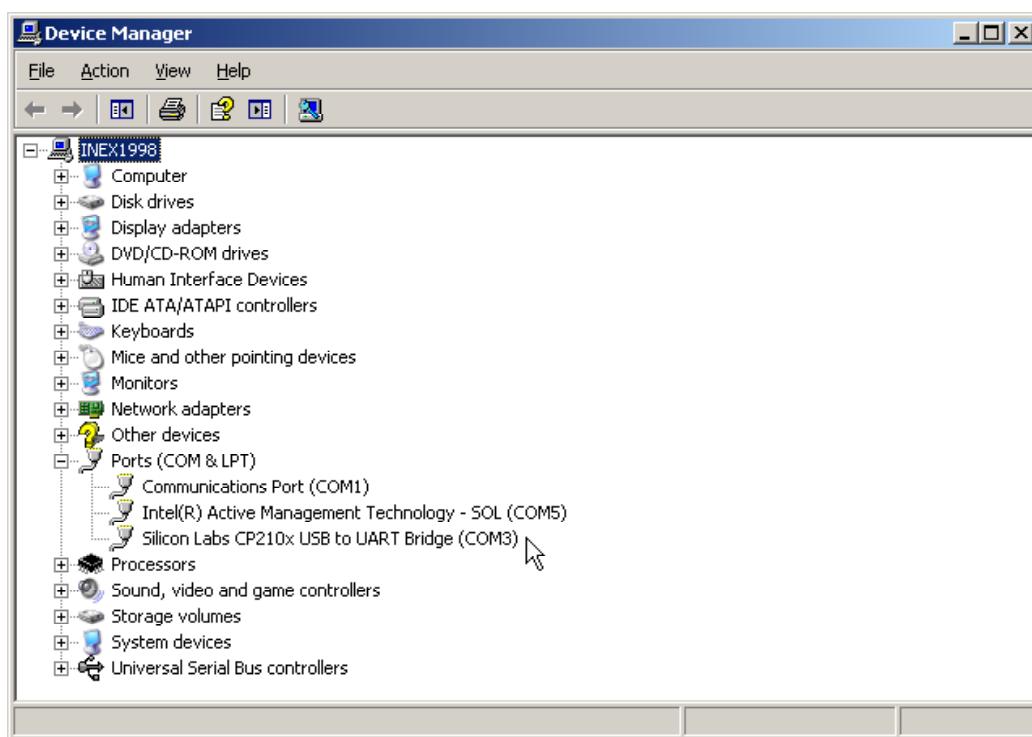
รูปที่ 3-1 ทดสอบการทำงานเบื้องต้นของ NodeMCU-12E บนแผงต่อวงจรหรือเบรเดอร์ด



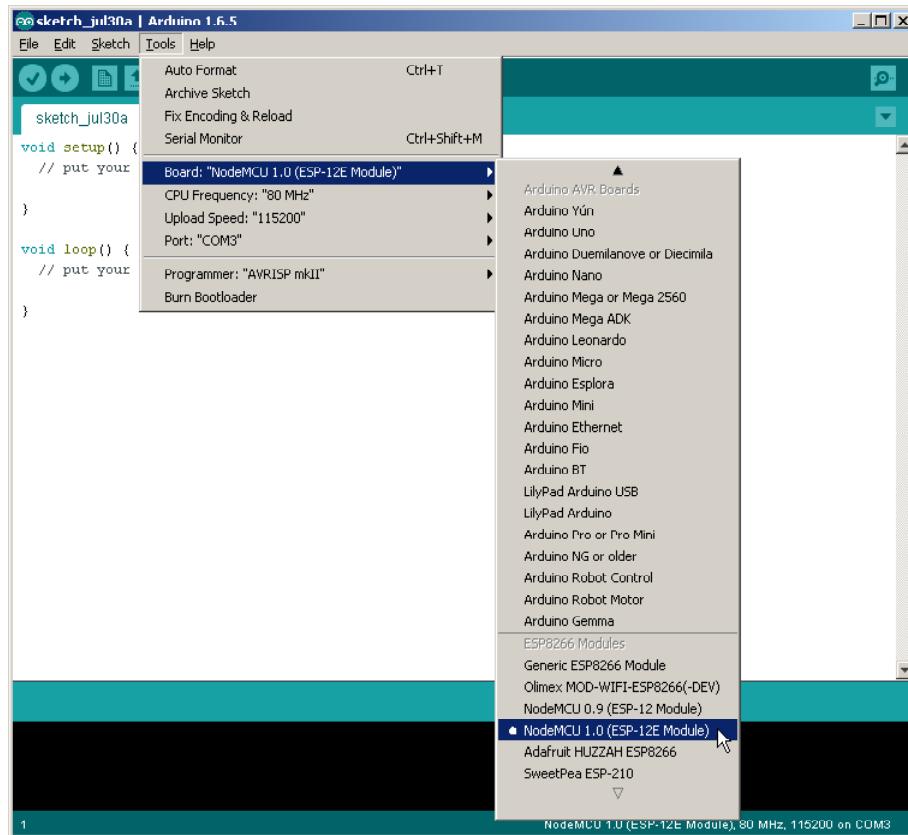
รูปที่ 3-2 การใช้งาน NodeMCU-12E กับบอร์ด AX-NodeMCU เพื่อการทดลอง เรียนรู้ และใช้งานจริง

(B) หากใช้โมดูล NodeMCU-12E กับบอร์ด AX-NodeMCU ให้ทำการติดตั้งโมดูล NodeMCU-12E บนซ็อกเก็ตของบอร์ด AX-NodeMCU ดังรูปที่ 3-2 จากนั้นต่อสาย microB-USB เชื่อมต่อโมดูล NodeMCU-12E และพอร์ต USB โดยไม่ต้องจ่ายไฟเลี้ยงเข้าที่แจ็કอะแดปเตอร์บนบอร์ด AX-NodeMCU

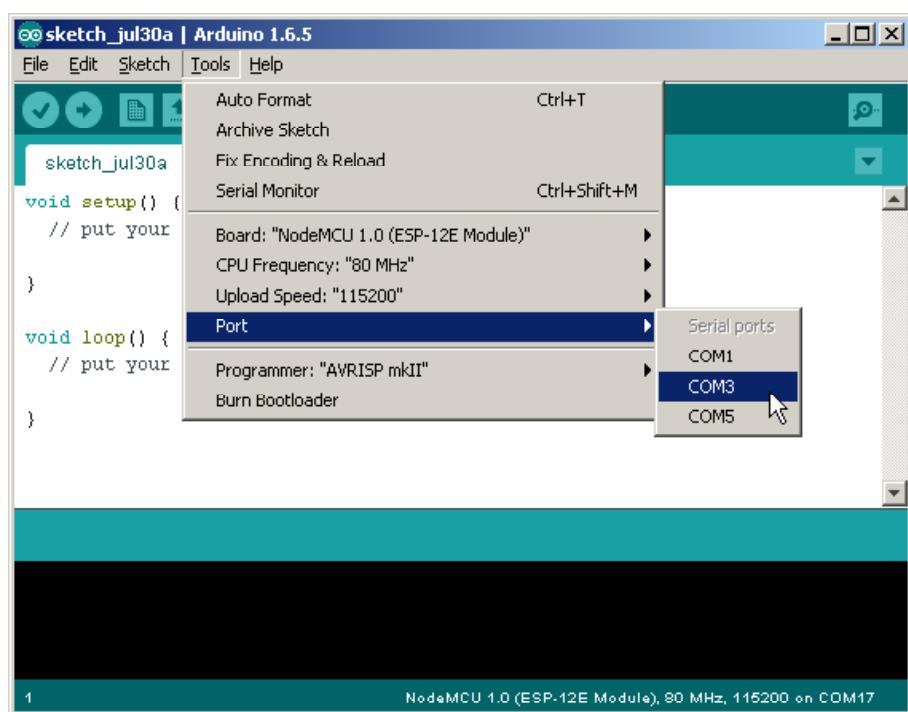
(2) ตรวจสอบพอร์ตเชื่อมต่อที่เกิดขึ้นจากไดรเวอร์ของ NodeMCU ได้ที่ **Control panel > System > Hardware > Device Manager > Port** ลังกอกหัวข้อ **Silicon Labs CP210x USB to UART Bridge (COMxx)** ในที่นี่คือ COM3



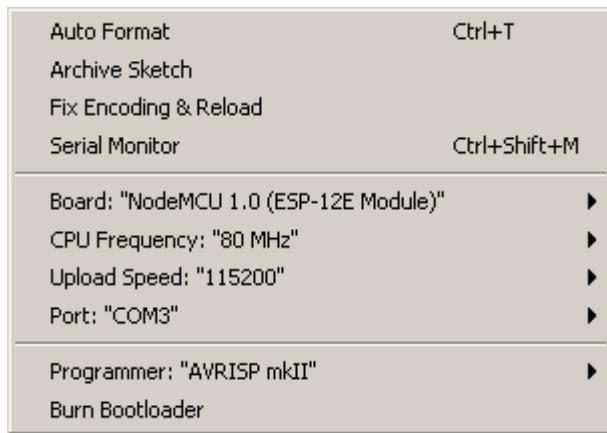
(3) เปิดซอฟต์แวร์ ArduinoIDE 1.6.5R2 แล้วเลือกชาร์ดแวร์โดยไปที่เมนู Tools > Board > NodeMCU 1.0 (ESP-12E Module)



(4) เลือกพอร์ตเชื่อมต่อ โดยไปที่เมนู Tools > Port > COM 3



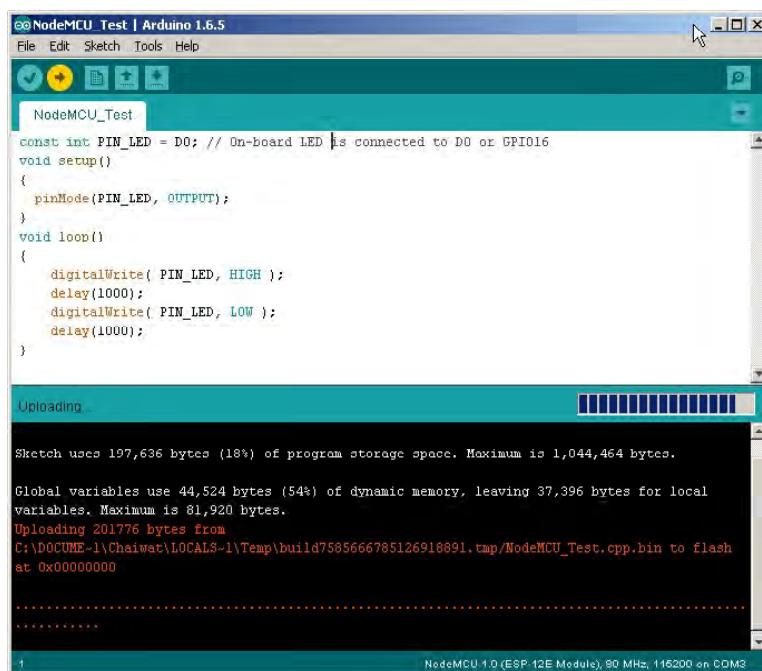
(5) จะได้ข้อมูลการเขื่อมต่อในภาพรวม ดังนี้



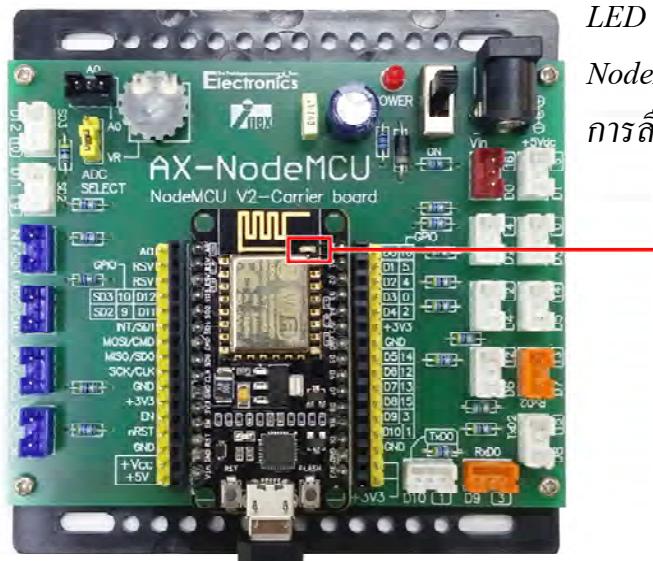
(6) เขียนโปรแกรมต่อไปนี้

```
const int PIN_LED = D0;
void setup()
{
    pinMode(PIN_LED, OUTPUT);
}
void loop()
{
    digitalWrite( PIN_LED, HIGH );
    delay(1000);
    digitalWrite( PIN_LED, LOW );
    delay(1000);
}
```

(7) อัปโหลดโค้ด โดยคลิกที่ปุ่ม Upload หรือเลือกเมนู Sketch > Upload หรือกดคีย์ Ctrl ตามด้วย U

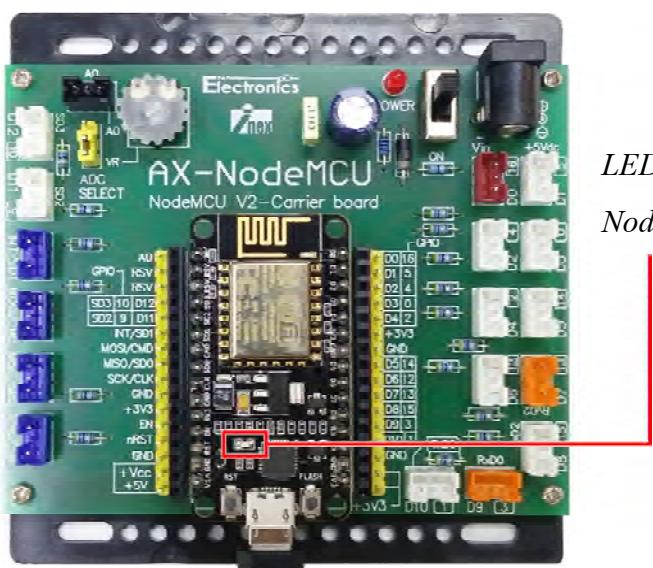


(8) ซอฟต์แวร์จะทำการคอมไพล์โค้ด (compiling) เมื่อเสร็จแล้วจะแสดงผลการคอมไпал์ขนาดไฟล์ พื้นที่เหลือของหน่วยความจำ ตามด้วยการอัปโหลดโค้ด และแสดงสถานะการอัปโหลดด้วยจุดไฟปลาสติก .... ที่หน้าต่างสถานะ พร้อมกันนี้ LED แสดงสถานะการอัปโหลดบนบอร์ด NodeMCU-12E (ซึ่งต่อ กับขาพอร์ต D4 หรือ GPIO2) จะติดกะพริบตามจังหวะการถ่ายทอดข้อมูล การอัปโหลดโค้ด จะใช้เวลาประมาณ 30 วินาที เมื่ออัปโหลดโค้ดไปยังบอร์ด ได้สำเร็จ จะแจ้งด้วยข้อความ **Done uploading** ที่ข่องแสดงสถานะ



LED สีนำเงินตำแหน่ง D4 บน บอร์ด NodeMCU กะพริบถี่ๆ ตามจังหวะ การถือสารข้อมูล

(9) เมื่ออัปโหลดโค้ดเสร็จ NodeMCU จะเริ่มทำงานทันที



LED ที่ตำแหน่ง D0 บน บอร์ด NodeMCU กะพริบทุกๆ วินาที



## บทที่ 4

# NodeMCU กับการติดต่ออุปกรณ์อินพุต เอาต์พุตพื้นฐาน

ก่อนที่จะเข้าสู่ขั้นตอนการพัฒนาอุปกรณ์ IoT ด้วย NodeMCU ขั้นตอนหนึ่งที่สำคัญและผู้พัฒนาควรให้ความสนใจคือ การพัฒนาโปรแกรมเพื่อใช้งานโมดูล NodeMCU-12E ในการติดต่อกับอุปกรณ์อินพุตเอาต์พุตภายนอกพื้นฐาน เช่น LED, สวิตช์ ไปจนถึงอุปกรณ์แสดงผลที่มีความซับซ้อนขึ้นอย่างโมดูล LCD รีจักรับอุปกรณ์ขับไฟLED และโซลิเดเตอร์LED ติดต่อและอ่านค่าจากตัวตรวจจับทั้งแบบอะนาล็อกและแบบดิจิตอลที่ต้องใช้การสื่อสารข้อมูลอนุกรมผ่านระบบบัส

ทั้งนี้เพื่อให้แน่ใจว่า การทำงานของชาร์ดแวร์ทั้งในการรับค่าจากตัวตรวจจับและอุปกรณ์อินพุตต่างๆ กับการส่งสัญญาณหรือข้อมูลไปยังอุปกรณ์เอาต์พุตเป็นไปอย่างถูกต้อง เมื่อเข้าสู่กระบวนการเชื่อมต่อกับเครือข่ายอินเทอร์เน็ตแล้ว การพัฒนาต่อจากนั้นจะถูกไฟกัสไปยังกระบวนการทางซอฟต์แวร์เป็นหลัก ไม่ว่าจะเป็นการนำข้อมูลไปแสดงผลในรูปของกราฟิกแบบต่างๆ การประมวลผลและจัดเก็บฐานข้อมูล การคุ้มครองข้อมูลด้านความปลอดภัยของข้อมูล



รูปที่ 4-1 อุปกรณ์ทางชาร์ดแวร์ทั้งหมดที่ใช้ในบทนี้ อย่างไรก็ตาม ผู้พัฒนาสามารถเปลี่ยนอุปกรณ์จากมินิบอร์ดสำเร็จรูปเป็นอุปกรณ์อิเล็กทรอนิกส์พื้นฐาน โดยดูจากการที่ใช้ในการทดลอง แล้วทำการต่อวงจรขึ้นใหม่บนแผงต่อวงจรหรือเบรดบอร์ดก็ได้ (สำหรับผู้จัดซื้อชุด IoT Education Kit จะมีอุปกรณ์ทั้งหมดนี้ครบถ้วน)

หากการทำงานทางชาร์ดแวร์ไม่มีความแน่นอนหรือไม่เสถียร ข้อมูลที่ได้รับหรือส่งต่อจากการเชื่อมต่อกับอุปกรณ์ชาร์ดแวร์จะไม่เกิดประโภชน์ ส่งผลให้อุปกรณ์ IoT ตัวนั้นๆ ล้มเหลวในการนำไปใช้งานอย่างสิ้นเชิง นั่นหมายความว่า ในการพัฒนาอุปกรณ์ IoT จะต้องมีความสมบูรณ์พร้อมทั้งชาร์ดแวร์ ซอฟต์แวร์ การจัดการค้านเครือข่าย และการบริหารพื้นที่หน่วยความจำ จึงจะทำให้อุปกรณ์ IoT ตัวนั้นๆ หรือระบบนั้นๆ ทำงานได้อย่างมีประสิทธิภาพ

ในบทนี้จึงให้ความสำคัญต่อการนำเสนอด้วยการติดต่อกับอุปกรณ์ภายนอกของโมดูล NodeMCU ซึ่งเป็นอุปกรณ์หลักในการพัฒนาอุปกรณ์ IoT สำหรับผู้เริ่มต้นและบรรดาเมกเกอร์ที่ต้องการก้าวเข้าสู่โลกของ IoT โดยแบ่งเนื้อหาออกเป็น 4 ส่วน ดังนี้

1. แนะนำขั้นตอนในการพัฒนาโปรแกรม
2. ตัวอย่างการติดต่อกับอุปกรณ์เอาต์พุตพื้นฐาน เช่น LED และ LED 3 สี RGB
3. ตัวอย่างการติดต่อกับอุปกรณ์อินพุตพื้นฐาน เช่น สวิตช์กดติดปล่อยดับ
4. การติดต่อกับโมดูลแสดงผล LCD ผ่านบัส 2 สายหรือ I<sup>2</sup>C

## 4.1 อุปกรณ์ชาร์ดแวร์

ประกอบด้วย

1. โมดูล NodeMCU-12E และบอร์ด AX-NodeMCU เป็นส่วนประมวลผลและความคุณการทำงาน - อาจใช้โมดูล NodeMCU-12E ร่วมกับแพงต์อองจรหรือเบรเดบอร์ดขนาด 390 หรือ 800 ชุดเป็นทางเลือกแทนได้

2. ZX-LED มินิบอร์ดขับ LED (3 ตัว) - อาจใช้ LED ปกติต่อร่วมกับตัวต้านทานจำกัดกระแสไฟฟ้า (ค่า  $510\Omega$ ) เป็นทางเลือกแทนได้

3. ZX-LED3CS มินิบอร์ดขับ LED 3 สี RGB - อาจใช้ LED 3 สี RGB แบบแคนโปล์ร่วมต่อกับตัวต้านทานจำกัดกระแสไฟฟ้า 3 ตัว (ค่า  $330\Omega$ ,  $510\Omega$  และ  $680\Omega$ ) เป็นทางเลือกแทนได้

4. ZX-SPEAKER มินิบอร์ดลำโพงเปียโซ - อาจใช้ลำโพงเปียโซต่อร่วมกับตัวเก็บประจุ  $10\mu F$   $16V$  ชนิดอิเล็กโทรไลต์ เป็นทางเลือกแทนได้

5. ZX-SWITCH01 มินิบอร์ดสวิตช์อินพุต - อาจใช้สวิตช์กดติดปล่อยดับที่เสียบลงบนเบรเดบอร์ด ได้ต่อร่วมตัวต้านทานสำหรับพูลอัป (ค่า  $10k\Omega$ ) เป็นทางเลือกแทนได้

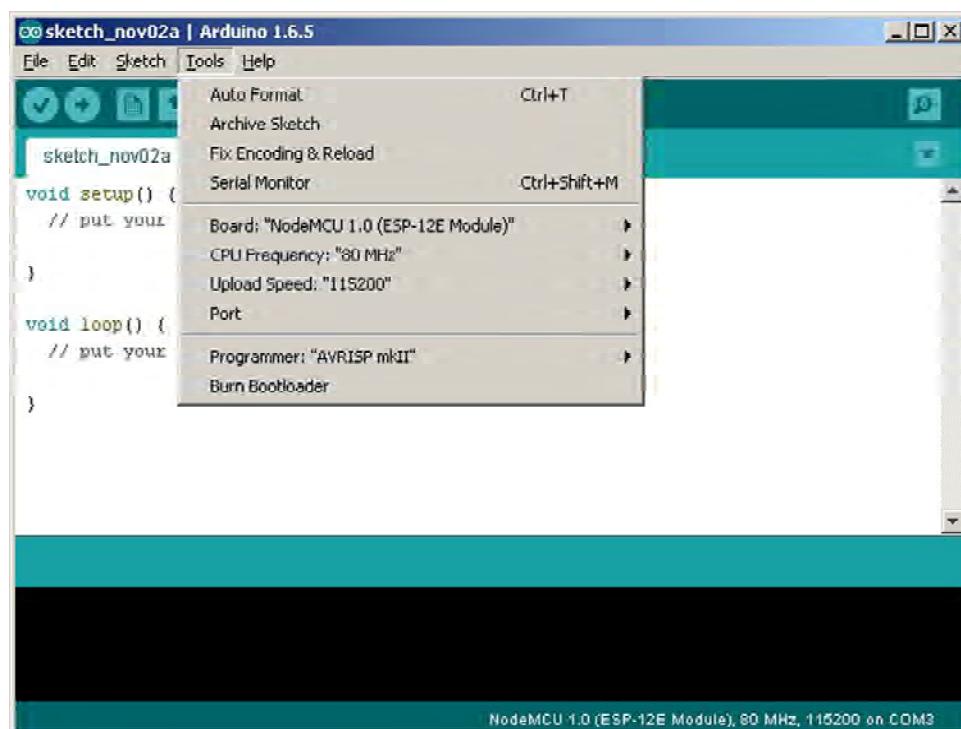
6. โมดูล LCD 16 ตัวอักษร 2 บรรทัดแบบใช้การเชื่อมต่อผ่านบัส I<sup>2</sup>C (โมดูล I2C-LCD16x2)

## 4.2 ซอฟต์แวร์ที่ใช้ในการพัฒนาโปรแกรม

การพัฒนาโปรแกรมให้แก่ NodeMCU-12E ในที่นี้เลือกใช้ภาษา C/C++ แบบโอเพ่นซอร์ส (open source) ที่นำมาใช้ได้โดยไม่ต้องเสียค่าใช้จ่าย นั่นคือ **Arduino IDE** เวอร์ชัน 1.6.5r2

Arduino IDE เวอร์ชันนี้เป็นรุ่นพิเศษที่ INEX จัดทำขึ้น เป็นซอฟต์แวร์ Arduino IDE for ESP8266/NodeMCU โดยนำ Arduino IDE เวอร์ชัน 1.6.5 มาพนวกเข้ากับคอมไพล์เตอร์และไลบรารีของ ESP8266 ได้เป็นชุดซอฟต์แวร์สำหรับพัฒนาโปรแกรมมีความสมบูรณ์พร้อม ทั้งด้านไลบรารีที่บรรจุฟังก์ชันสำหรับติดต่อกับฮาร์ดแวร์ได้หลากหลาย รวมถึงการติดต่อกับเครือข่ายอินเทอร์เน็ตผ่าน WiFi สนับสนุนการพัฒนาโปรแกรมด้วยหน้าต่างการทำงานเพียงหน้าต่างเดียว ตั้งแต่เขียนโปรแกรม คอมไพล์ จนถึงการอัปโหลดโค้ด ทำให้ง่ายต่อการทำความเข้าใจและใช้งาน ผู้ใช้งานอาจไม่จำเป็นต้องมีความรู้ด้านสารคดแวร์มากนักสามารถเขียนโปรแกรมควบคุมอุปกรณ์สารคดแวร์ต่างๆ ได้ทั้งแบบโดยตรงและผ่านเครือข่าย WiFi นอกจากนี้ยังมีนักพัฒนาจากทั่วโลกร่วมพัฒนาไฟล์ไลบรารีเพิ่มเติม ทำให้ Arduino เวอร์ชันนี้มีความสามารถเพิ่มมากขึ้น

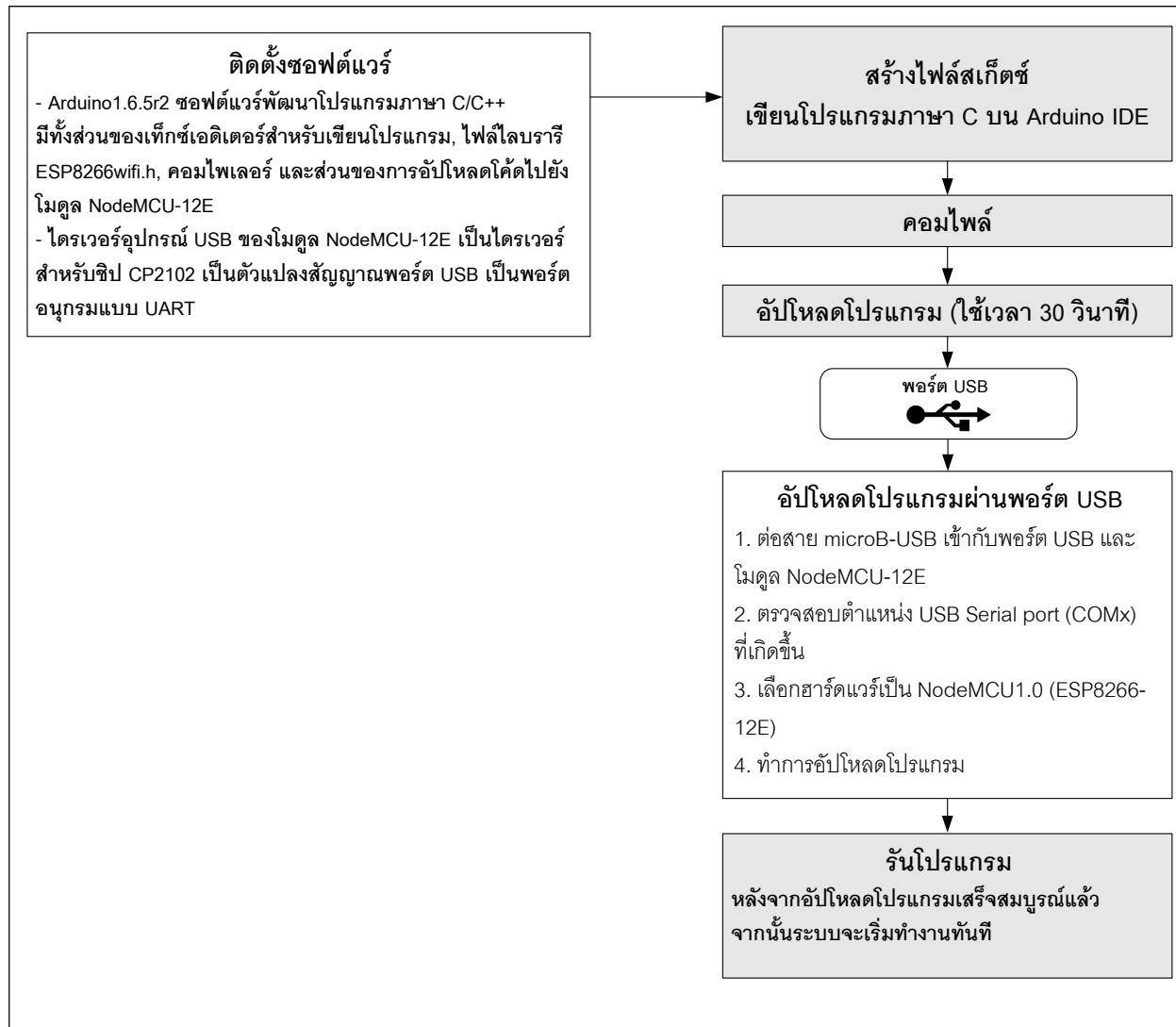
Arduino IDE for ESP8266/NodeMCU ที่ INEX จัดทำขึ้น ดาวน์โหลดได้โดยไม่มีค่าใช้จ่ายที่ [www.inex.co.th](http://www.inex.co.th) หรือ [http://www.mediafire.com/download/rvo7q6j131t4pc6/Arduino1.6.5r2\\_Setup150707.exe](http://www.mediafire.com/download/rvo7q6j131t4pc6/Arduino1.6.5r2_Setup150707.exe) หรือติดตั้งจาก USB แฟลชไ/dr/Firmwareที่มา กับชุด IoT Education Kit - NodeMCU



รูปที่ 4-2 หน้าต่างของซอฟต์แวร์ Arduino IDE เวอร์ชัน 1.6.5r2 ที่ทำขึ้นเป็นพิเศษ มีการพนวกโมดูล NodeMCU ไว้ในรายการสำหรับเลือกอุปกรณ์สารคดแวร์ที่ใช้ในการพัฒนาโปรแกรม

### 4.3 ขั้นตอนการพัฒนาโปรแกรมภาษา C/C++ เพื่อใช้งาน NodeMCU-12E

ขั้นตอนการพัฒนาโปรแกรมสำหรับโมดูล WiFi คอนโทรลเลอร์ NodeMCU-12E หรือ V2 หรือ Development kit 1.0 แสดงเป็นแผนภาพดังรูปที่ 4-3



รูปที่ 4-3 แสดงผังงานของการพัฒนาโปรแกรมเพื่อควบคุมการทำงานของโมดูล NodeMCU-12E ด้วยภาษา C/C++ โดยใช้ Arduino IDE 1.6.5r2 เป็นรุ่นที่ปรับปรุงให้ใช้งานกับโมดูล NodeMCU ได้สะดวกและง่ายขึ้น

## 4.4 ข้อควรทราบในการเขียนโปรแกรมภาษา C/C++ ด้วย Arduino IDE สำหรับ NodeMCU-12E

ในการเขียนโปรแกรมภาษา C/C++ สำหรับโมดูล NodeMCU-12E โดยใช้ภาษาของ Arduino (Arduino programming language) ผู้พัฒนาโปรแกรมควร มีความเข้าใจในส่วนประกอบของโปรแกรม เลยก่อน ซึ่งแบ่งได้เป็น 2 ส่วนหลักคือ

1. โครงสร้างภาษา (structure) ตัวแปรและค่าคงที่
2. ฟังก์ชัน (function)

ภาษาของ Arduino จะอ้างอิงตามภาษา C/C++ จึงอาจกล่าวได้ว่า การเขียนโปรแกรมสำหรับ Arduino ก็คือการเขียน โปรแกรมภาษา C โดยเรียกใช้ฟังก์ชันและไลบรารีที่ทาง Arduino ได้เตรียมไว้ให้แล้ว ซึ่งสะดวก และทำให้ผู้ที่ไม่มีความรู้ด้านไมโครคอนโทรลเลอร์ย่างลึกซึ้ง สามารถเขียนโปรแกรมสั่งงานได้

โปรแกรมของ Arduino แบ่งได้เป็นสองส่วนคือ

```
void setup()      และ  
void loop()
```

โดยฟังก์ชัน `setup()` เมื่อโปรแกรมทำงานจะทำการสั่งของฟังก์ชันนี้เพียงครั้งเดียว จึงเหมาะสมที่จะใช้ในการกำหนดค่าเริ่มต้นของการทำงาน โดยปกติใช้กำหนดโหมดการทำงานของขาต่างๆ

ส่วนฟังก์ชัน `loop()` เป็นส่วนทำงาน โปรแกรมจะทำการสั่งในฟังก์ชันนี้ต่อเนื่องกันตลอดเวลา เช่น อ่านค่าอินพุต ประมวลผล ลั่งงานเอาต์พุต ฯลฯ สำหรับการกำหนดค่าเริ่มต้น เช่น ตัวแปร จะต้องเขียนที่ส่วนหัวของโปรแกรม ก่อนถึงตัวฟังก์ชัน นอกจากนั้น ยังต้องคำนึงถึงตัวพิมพ์เล็ก-ใหญ่ ของตัวแปรและชื่อฟังก์ชันให้ถูกต้องด้วย

#### 4.4.1 ส่วนของฟังก์ชัน `setup()`

ฟังก์ชันนี้จะเขียนที่ส่วนต้นของโปรแกรม ทำงานเมื่อโปรแกรมเริ่มต้นเพียงครั้งเดียว ใช้เพื่อกำหนดค่าของตัวแปร ให้มีการทำงานของขาต่างๆ เริ่มต้นเรียกใช้ไลบรารี ฯลฯ

##### ตัวอย่างที่ 4-1

```
int LEDpin = D0;
void setup()
{
    pinMode(LEDpin, OUTPUT);
}
void loop()
{
    digitalWrite(LEDpin, HIGH);
    delay(500);
    digitalWrite(LEDpin, LOW);
    delay(500);
}
```

จากตัวอย่างนี้ ฟังก์ชัน `setup()` ใช้กำหนดให้มีการทำงานของขาพอร์ตที่อ้างอิงด้วยชื่อ `LEDpin` ซึ่งก็คือ ขาพอร์ต `D0` ของโมดูล NodeMCU-12E ให้เป็นขาพอร์ตเอาต์พุตดิจิตอล

#### 4.4.2 ส่วนของฟังก์ชัน `loop()`

หลังจากที่เขียนฟังก์ชัน `setup()` ที่กำหนดค่าเริ่มต้นของโปรแกรมแล้ว ส่วนถัดมาคือฟังก์ชัน `loop()` ซึ่งมีการทำงานตรงตามชื่อคือ จะทำงานตามฟังก์ชันนี้วนต่อเนื่องตลอดเวลา ภายในฟังก์ชันนี้จะมีโปรแกรมของผู้ใช้ เพื่อรับค่าจากพอร์ต ประมาณ แล้วส่งเอาต์พุตออกขาต่างๆ เพื่อควบคุมการทำงานของบอร์ด

##### ตัวอย่างที่ 4-2

```
int LEDpin = D0;
void setup()
{
    pinMode(LEDpin, OUTPUT);
}
void loop()
{
    digitalWrite(LEDpin, HIGH);
    delay(500);
    digitalWrite(LEDpin, LOW);
    delay(500);
}
```

จากตัวอย่างนี้ ฟังก์ชัน `loop()` เป็นการสั่งให้ขาพอร์ตที่อ้างอิงด้วยชื่อ `LEDpin` ซึ่งก็คือ ขาพอร์ต `D0` ของโมดูล NodeMCU-12E มีสถานะเป็น “1” (`HIGH`) และ “0” (`LOW`) สลับกันทุกๆ 0.5 วินาที และวนทำงานเช่นนี้ไปตลอด

## 4.5 ตัวอย่างการติดต่ออุปกรณ์อินพุตเอาต์พุตพื้นฐานของโมดูล NodeMCU-12E

ขั้นตอนการพัฒนาโปรแกรมในแต่ละตัวอย่างในหัวข้อนี้จะเนื่องกัน นั่นคือ เปิดซอฟต์แวร์ Arduino IDE 1.6.5r2 ที่ปรับปรุงมาเป็นพิเศษสำหรับพัฒนาโปรแกรมให้กับโมดูล NodeMCU-12E ทำการเขียนโปรแกรม คอมไฟล์ และอัปโหลดลงบนโมดูล NodeMCU-12E จากนั้นต่อวงจรเพื่อทดสอบการทำงาน

การต่อวงจรสำหรับเชื่อมต่อ กับ อุปกรณ์อินพุตเอาต์พุต ให้แก่ โมดูล NodeMCU-12E ในหัวข้อนี้ มีด้วยกัน 2 ลักษณะคือ แนะนำว่าจะที่ผู้ทดลองสามารถต่อวงจรบนเบรเดอร์ได้ แม้ว่าไม่ได้จัดซื้อชุด IoT Education Kit - NodeMCU กับอีกลักษณะคือ การต่อวงจรโดยใช้อุปกรณ์ที่จัดเตรียมมาพร้อมในชุด IoT Education Kit

สิ่งสำคัญที่ต้องเน้นย้ำคือ

1. การจ่ายไฟให้แก่ โมดูล NodeMCU-12E ทำได้ 2 ทาง นั่นคือ จ่ายไฟผ่านทางพอร์ต USB และจ่ายไฟ +5V เข้าที่ขา +Vcc ของ โมดูล NodeMCU-12E **แต่ต้องไม่ทำพร้อมกัน**
2. ในการเขียนโปรแกรม ต้องเลือกชาร์ดแวร์หรือ Tool > Board > NodeMCU1.0 (ESP-12E module) และเลือกช่องการเชื่อมต่อในเมนู Tool > Port ให้ถูกต้อง ก่อนทำการคอมไฟล์และอัปโหลด โค้ด โปรแกรม

## 4.5.1 ใช้งานพอร์ตเอาต์พุตขั้บ LED

### 4.5.1.1 คุณสมบัติของขาพอร์ตเอาต์พุต

สำหรับขาพอร์ตทุกขาของ NodeMCU-12E กำหนดให้เป็นเอาต์พุตดิจิตอลได้ทั้งหมด เมื่อทำการกำหนดเป็นเอาต์พุตดิจิตอลแล้ว ขาพอร์ตจะมีสถานะเป็นอิมพีเดนซ์ต่ำ ทำให้จ่ายกระแสไฟฟ้าให้กับวงจรภายนอกได้สูงสุด 15mA ซึ่งเพียงพอสำหรับขั้บให้ LED สว่าง (ต้องต่อตัวทานทานอนุกรมเพื่อจำกัดกระแสไฟฟ้าให้ LED ด้วย) หรือใช้กับอุปกรณ์เอาต์พุตดิจิตอลอื่นๆ ได้แต่ไม่เพียงพอสำหรับขั้บเรลย์ โซลินอยด์ หรือมอเตอร์ โดยตรง

### 4.5.1.2 การกำหนดโหมดของขาพอร์ต

ในรูปที่ 4-4 แสดงขั้นตอนการใช้งานขาพอร์ตของโมดูล NodeMCU-12E เป็นพอร์ตเอาต์พุตดิจิตอล โดยก่อนใช้งานต้องกำหนดโหมดการทำงานเสียก่อน ด้วยฟังก์ชัน `pinMode()` มีรูปแบบดังนี้

`pinMode(pin, mode);`

เมื่อ `pin` คือ หมายเลขขาพอร์ตที่ต้องการ

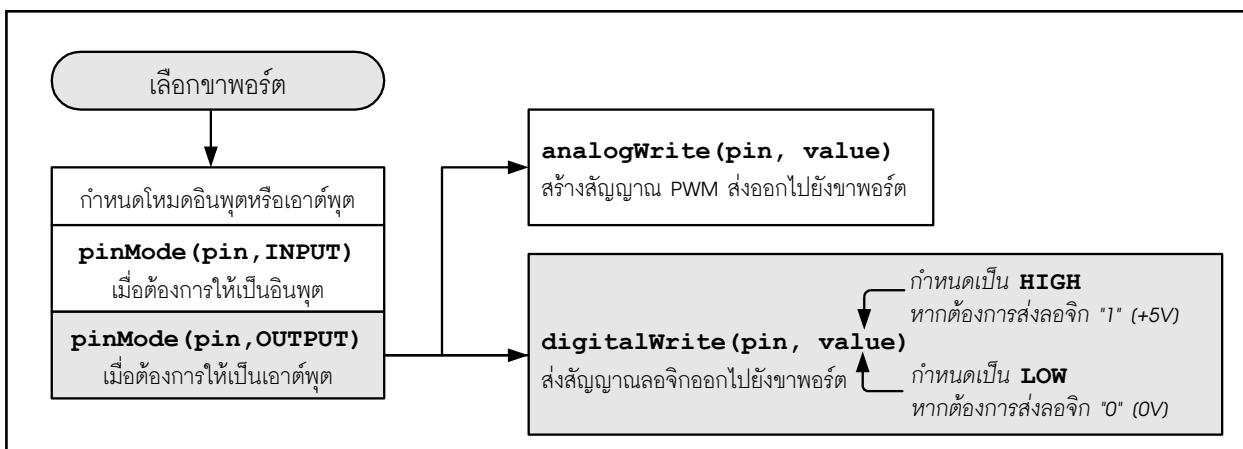
`Mode` คือ โหมดการทำงานเป็นอินพุต (`INPUT`) หรือ เอาต์พุต (`OUTPUT`)

หลังจากกำหนดให้เป็นพอร์ตเอาต์พุตแล้ว เมื่อต้องการเปลี่ยนค่าไปยังขาพอร์ตนั้นๆ ให้เรียกใช้ฟังก์ชัน `digitalWrite()` โดยมีรูปแบบดังนี้

`digitalWrite(pin, value);`

เมื่อ `pin` คือ หมายเลขขาพอร์ตที่ต้องการ

`value` สถานะโลจิกที่ต้องการ (`HIGH` หรือ `LOW`)



รูปที่ 4-4 ไดอะแกรมแสดงกลไกการกำหนดให้ขาพอร์ตของ NodeMCU-12E เป็นพอร์ตเอาต์พุตดิจิตอล (บล็อกสีเทาแทนการทำงานหรือเงื่อนไขที่ต้องการ) ใน การเขียนโปรแกรมด้วย Arduino IDE

#### 4.5.1.3 ตัวอย่างโปรแกรมขับ LED 3 ดวง

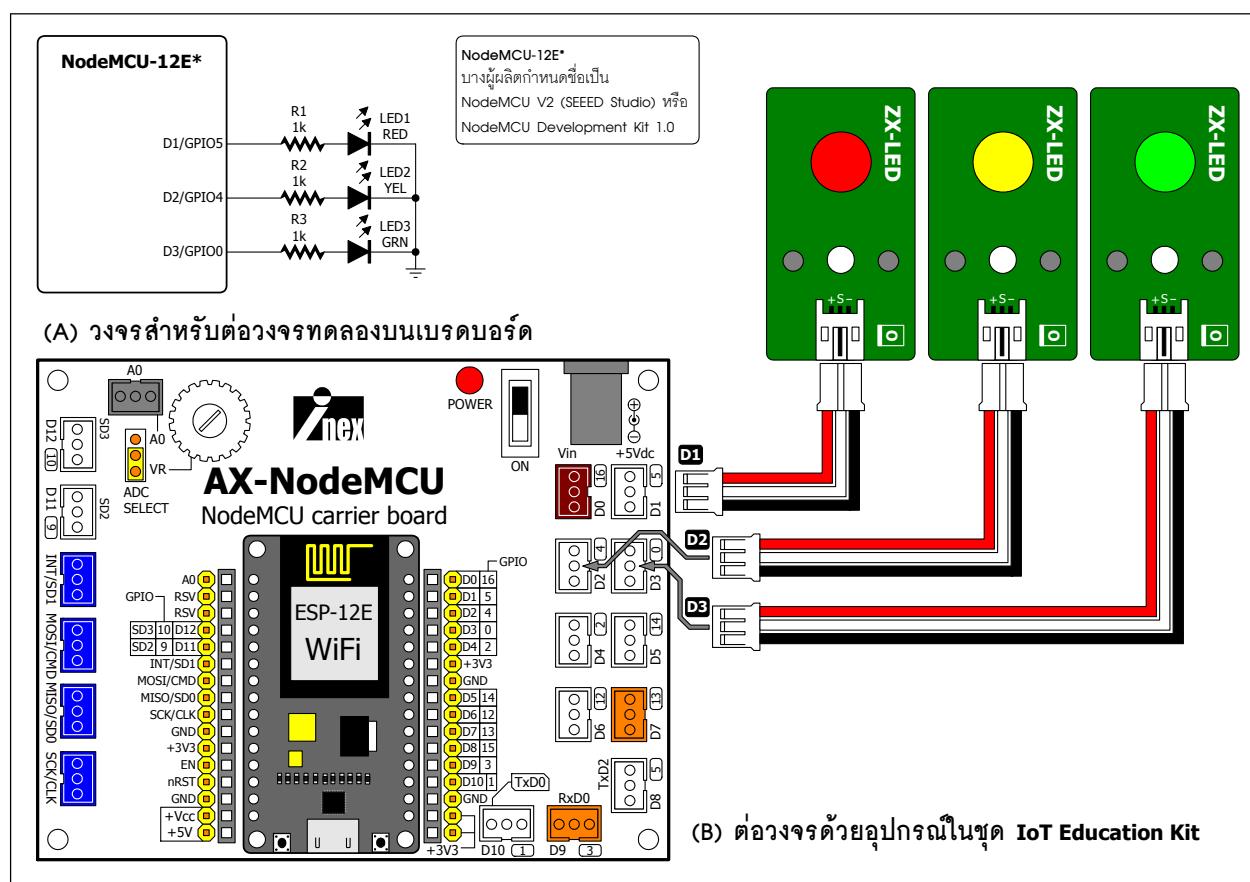
ในตัวอย่างนี้เป็นการสั่งให้พอร์ตทำงานเป็นเอาต์พุต จากนั้นสั่งให้ขาพอร์ตเอาต์พุตนั้นๆ เป็นลอจิก “0” หรือ “1” ตามที่ต้องการ โดยต่อ กับ LED สั่งให้ LED ติดดับต่อเนื่องกัน

เริ่มต้นด้วยการต่อวงจร ดังรูปที่ 4-5 โดยในรูปที่ 4-5 (A) เป็นวงจรทดลองที่แนะนำให้ต่อลงบนเบรดบอร์ดหรือแพงต์วงจร ส่วนรูปที่ 4-5 (B) เป็นการต่อทดลองโดยใช้อุปกรณ์ที่มีในชุด IoT Education Kit

โดย LED ทำงานที่ล็อกิก “1” คือเมื่อสั่งให้ขาพอร์ตเป็น HIGH จะทำให้ LED ติด เมื่อสั่งให้ขาเป็น LOW หรือ “0” LED จะดับ เมื่อต่ออุปกรณ์แล้ว ให้เขียนโปรแกรมที่ 4-1 ทดลองคอมไฟล์ อัปโหลดลง NodeMCU-12E ศึกษาผลการทำงาน

เมื่ออัปโหลดโค้ดและรันโปรแกรม

LED3 ที่ต่อ กับขาพอร์ต D3 ติดก่อน ในจังหวะแรก จากนั้นอีก 1 วินาที LED2 ที่ต่อ กับขาพอร์ต D2 จะติดพร้อมกับ LED3 ถัดไปอีก 1 วินาที LED ทั้ง 3 ดวงจะติดหมุน และดับลงพร้อมกันในอีก 1 วินาที จากนั้นวนกลับไปทำงานให้ LED3 ติดเพียงดวงเดียวอีกรอบ



รูปที่ 4-5 วงจรสำหรับทดลองใช้งานขาพอร์ตเอาต์พุตของ NodeMCU-12E ในการขับ LED 3 ดวง

```

#define ledPin1 D1           // Define pin name
#define ledPin2 D2
#define ledPin3 D3
void setup()
{
    pinMode(ledPin1,OUTPUT); //Set D1 pin as output
    pinMode(ledPin2,OUTPUT); //Set D2 pin as output
    pinMode(ledPin3,OUTPUT); //Set D3 pin as output
}

void loop()
{
    digitalWrite(ledPin1, 0); //Set display pattern off-off-on
    digitalWrite(ledPin2, 0);
    digitalWrite(ledPin3, 1);
    delay(1000);           //Delay 1 second
    digitalWrite(ledPin1, 0); //Set display pattern off-on-on
    digitalWrite(ledPin2, 1);
    digitalWrite(ledPin3, 1);
    delay(1000);           //Delay 1 second
    digitalWrite(ledPin1, 1); //Set display pattern on-on-on
    digitalWrite(ledPin2, 1);
    digitalWrite(ledPin3, 1);
    delay(1000);           //Delay 1 second
    digitalWrite(ledPin1, 0); //Turn off all
    digitalWrite(ledPin2, 0);
    digitalWrite(ledPin3, 0);
    delay(1000);           //Delay 1 second
}

```

### คำอธิบายโปรแกรมเพิ่มเติม

ในโปรแกรมนี้มีการกำหนดชื่อของขาพอร์ตทั้ง 3 ขาที่ใช้ในตอนต้นของโปรแกรมด้วยคำสั่ง `#define` ยกตัวอย่าง ในโปรแกรมกำหนดค่าคงที่ให้ข้อมูล `ledPin1` มีค่าเท่ากับ `D1` หลังจากบรรทัดนี้ในโปรแกรม เมื่อพิมพ์ข้อความ `ledPin1` จะหมายถึง ขาพอร์ต `D1` เป็นต้น

ส่วนของฟังก์ชัน `setup()` ใช้กำหนดการทำงานของขาพอร์ตทั้งสามขาให้เป็นเอกสารพูดด้วยคำสั่ง `pinMode`

ส่วนของฟังก์ชัน `loop()` เป็นโปรแกรมกำหนดให้ LED ทำงานตามรูปแบบที่กำหนดซึ่งมีด้วยกัน 3 แบบ แต่ละแบบจะคืนด้วยการหน่วงเวลา 1 วินาทีด้วยคำสั่ง `delay(1000)` ; ขั้นเป็นการเรียกใช้ฟังก์ชัน `delay()` โดยส่งค่า 1000 ให้กับฟังก์ชัน เพื่อให้เกิดการหน่วงเวลา 1000 มิลลิวินาที (ms) หรือ 1 วินาที เปลี่ยนค่าในวงล้อเป็นค่าอื่นๆ ได้ หากค่าอื่นมากการทำงานของแต่ละรูปแบบจะนานขึ้น

**โปรแกรมที่ 4-1 ไฟล์ NodeMCU\_LED3.ino โปรแกรมสำหรับใช้งานขาพอร์ตเอกสารพูดของ NodeMCU-12E เพื่อขับ LED 3 ดวง**

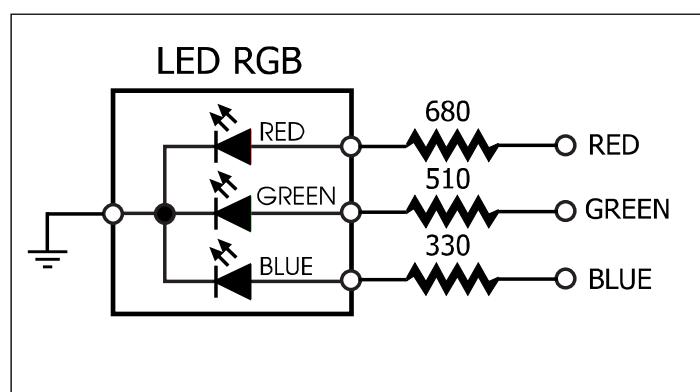
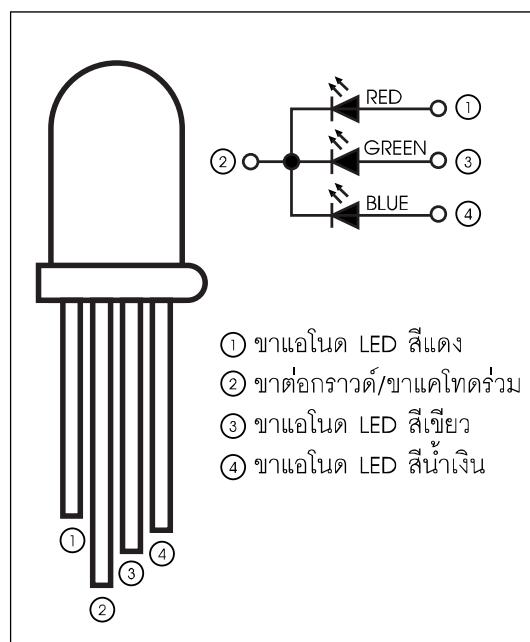
## 4.5.2 ใช้งานพอร์ตเอาต์พุตขับ LED 3 สี RGB

### 4.5.2.1 ความรู้เบื้องต้นเกี่ยวกับ LED 3 สี RGB

เมื่อมีการผลิต LED สีน้ำเงินขึ้นสำเร็จ ทำให้มี LED ขึ้นแสงที่เป็นแม่สีแสงครบ นั่นคือ แสงสีแดง, เขียว และน้ำเงิน ดังนั้นการใช้งานอุปกรณ์แสดงผล LED ในสมัยใหม่จึงควรเรียนรู้ที่จะใช้งาน LED แบบ 3 สี RGB ที่บรรจุมาในตัวถังเดียวกัน ทำให้มีการทำการขับสัญญาณมาอย่างเหมาะสม ก็จะสั่งให้ LED ผสมแสงสีได้ มือใหม่จะได้เห็นการทำงานจริงๆ ของการผสมแม่สีแสง ให้ได้สีของแสง ดังที่ต้องการ และพิสูจน์ทฤษฎีให้เห็นจริงว่า แสงขาวเกิดขึ้นจากการรวมแม่สีแสงทั้งสามเข้าด้วยกัน ในที่นี้เลือกใช้ LED 3 สี RGB ในแบบแคลโทรดร์วม

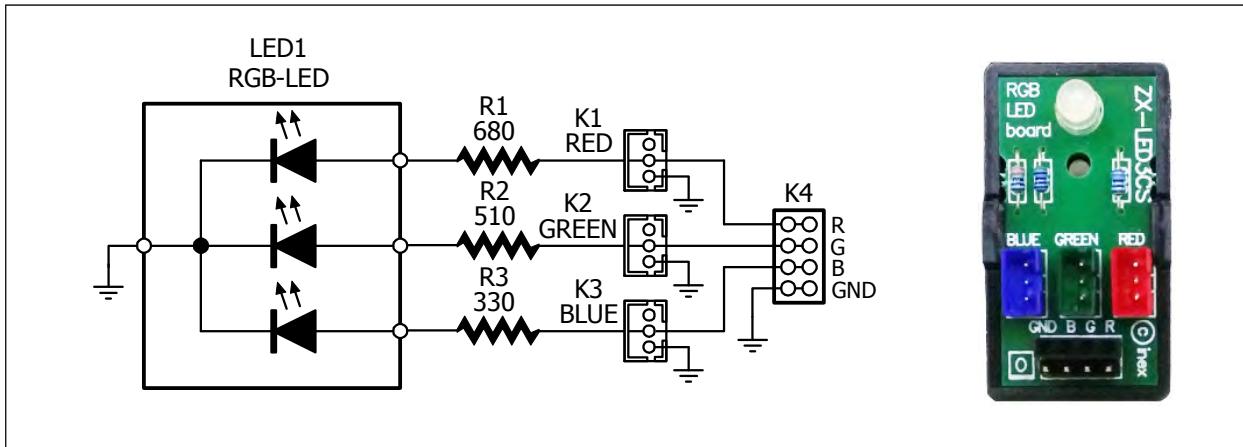
LED 3 สี RGB แบบแคลโทรดร์วมมีรูปร่างหน้าตาแสดงดังรูปที่ 4-6 มีขาต่อใช้งาน 4 ขาคือ ขาร่วม (common), ขาสำหรับขับสีแดง, สีเขียว และน้ำเงิน ในการใช้งานต้องต่อตัวต้านทานจำกัดกระแสไฟฟ้าที่ขาแอโนดของสีแดง, เขียว และน้ำเงิน แยกจากกัน ดังแสดงในรูปที่ 4-7 จะเห็นว่า ค่าของตัวต้านทานแต่ละตัวไม่เท่ากัน ทั้งนี้เนื่องจาก LED แต่ละสีที่รวมกันอยู่ภายใน LED 3 สีนี้มีความต้องการกระแสไฟฟ้าในการขับให้สว่างไม่เท่ากัน LED สีแดงจะต้องการกระแสไฟฟ้าต่ำสุด ถัดมาเป็นสีเขียว และสุดท้ายคือ สีน้ำเงิน (ในกรณีที่เป็น LED มาตรฐาน)

หากต้องการให้ LED ขับเป็นแสงสีใดก็ให้ป้อนแรงดันบวกเข้าที่ขาแอโนดของสีนั้น สำหรับความสว่างและสีที่ขับออกมากจะขึ้นกับขนาดของแรงดันที่ป้อนให้ หากป้อนแรงดันที่เหมาะสมจะทำให้ LED 3 สีนี้ขับแสงสีขาวออกมาก

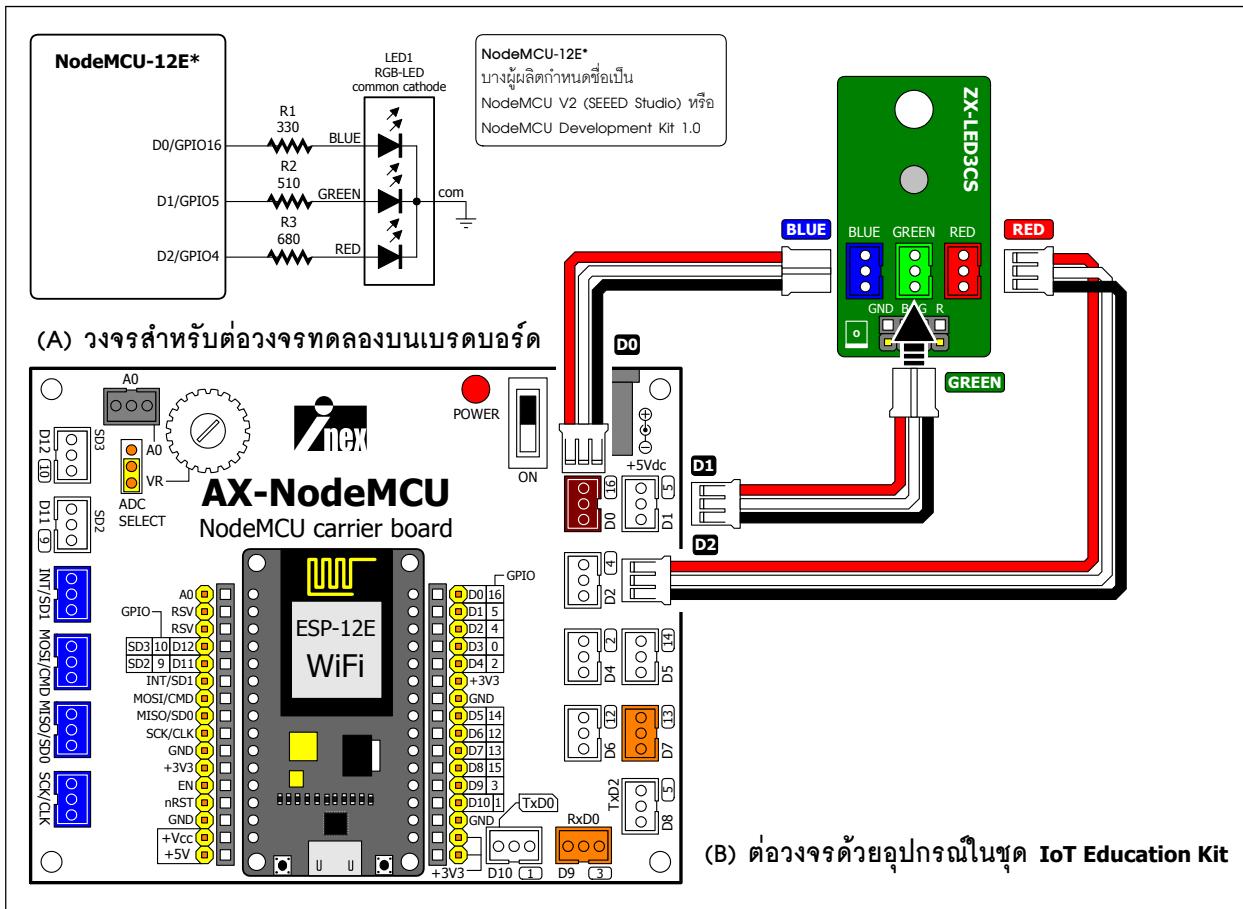


รูปที่ 4-7 วงจรใช้งาน LED 3 สี RGB แบบแคลโทรดร์วมเบื้องต้น

รูปที่ 4-6 การจัดขาของ LED 3 สี RGB แบบแคลโทรดร์วม



รูปที่ 4-8 วงจรミニบอร์ด LED 3 สี (ZX-LED3CS)



รูปที่ 4-9 วงจรสำหรับทดลองใช้งานขาพอร์ตเอาต์พุตของ NodeMCU-12E ในการขับ LED 3 สี RGB

#### 4.5.2.2 ZX-LED3CS มินิบอร์ด LED 3 สี

มีวงจรแสดงในรูปที่ 5 เป็นวงจรที่ต่ออยอดมาจากวงจรใช้งานพื้นฐานในรูปที่ 4-8 LED ที่ใช้เป็น LED 3 สี RGB แบบแคร์โพร์วม ตัวด้านท่าน R1 ถึง R3 มีค่าแตกต่างกัน เพื่อให้ LED แต่ละสีภายใน LED 3 สี RGB ทำงานได้ใกล้เคียงกันเมื่อได้รับแรงดันเท่าๆ กัน

#### 4.5.2.3 ตัวอย่างโปรแกรมใช้งานขาพอร์ต NodeMCU ขับ LED 3 สี RGB

เริ่มต้นด้วยการต่อวงจรตามรูปที่ 4-9 เช่นเดียวกับการทดลองก่อนหน้า การต่อวงจรทดลองทำได้ 2 ลักษณะคือต่อวงจรลงบนเบรดบอร์ดโดยใช้วงจรในรูปที่ 4-9 (A) เป็นแนวทาง หรือต่อวงจรด้วยอุปกรณ์ในชุด IoT Education Kit ดังรูปที่ 4-9 (B)

จากนั้นเขียนโปรแกรมที่ 4-2 ทำการคอมไพล์และอัปโหลดไปยัง NodeMCU-12E จากนั้นทำการรันโปรแกรมเพื่อทดสอบการทำงาน

LED 3 สีจะเริ่มต้นด้วยการคั่บก่อน จากนั้นถูกขับให้แสดงแสงสีแดง ตามด้วยสีเหลือง เจียว ฟ้า น้ำเงิน ชมพู และขาว แล้ววนกลับไปคั่บลงอีกรึ แต่ละแสงสีจะแสดงผลนาน 2 วินาที

```

#define B_pin D0           //Blue
#define G_pin D1           //Green
#define R_pin D2           //Red
int st_B = 0;
int st_G = 0;
int st_R = 0;
void setup()
{
    pinMode(B_pin, OUTPUT); // Set pin as output
    pinMode(G_pin, OUTPUT);
    pinMode(R_pin, OUTPUT);
}
void loop()
{
    displayRGB(0, 0, 0);   // Drive RGB LED with 3-bit data
    delay(2000);           // Delay 2 seconds
    displayRGB(1, 0, 0);
    delay(2000);
    displayRGB(1, 1, 0);
    delay(2000);
    displayRGB(0, 1, 0);
    delay(2000);
    displayRGB(0, 1, 1);
    delay(2000);
    displayRGB(0, 0, 1);
    delay(2000);
    displayRGB(1, 0, 1);
    delay(2000);
    displayRGB(1, 1, 1);
    delay(2000);
}
void displayRGB(int R, int G, int B)
// Send 3-bit digital data to 3 output pins function
{
    digitalWrite(B_pin, B);
    digitalWrite(G_pin, G);
    digitalWrite(R_pin, R);
}

```

### คำอธิบายโปรแกรม

เริ่มต้นด้วยการกำหนดขาพอร์ตสำหรับเชื่อมต่อ กับขาแອนดของ LED 3 สี ก่อนด้วยคำสั่ง `#define` จากนั้นกำหนดค่าสถานะเริ่มต้นของขาพอร์ตทั้งสามขาเป็น 0 ด้วยคำสั่ง `int` ในฟังก์ชัน `setup()` เป็นการกำหนดการทำงานของขาพอร์ตเป็นเอาร์พุตดิจิตอล

ในโปรแกรมทำงานหลักอยู่ในฟังก์ชัน `loop()` จะเป็นการเรียกใช้ฟังก์ชัน `displayRGB()` เพื่อส่งข้อมูล 3 บิตไปยังขาพอร์ตของ NodeMCU-12E เพื่อขับ LED ซึ่งมีด้วยกัน 8 ค่าคือ 000 ถึง 111 โดยค่า 000 ทำให้ LED ดับ ส่วนค่าอื่นๆ จะทำให้ LED ขับแสงที่แตกต่างกันรวม 7 สี การขับแสงสีแต่ละสีจะแสดงผลนาน 2 วินาที ตามค่าที่กำหนดในฟังก์ชัน `delay()`

---

**โปรแกรมที่ 4-2 ไฟล์ NodeMCU-LED3C.ino โปรแกรมสำหรับใช้งานขาพอร์ตเอาร์พุตของ NodeMCU-12E เพื่อขับ LED 3 สี RGB**

### 4.5.3 สร้างสัญญาณ PWM ผ่านพอร์ตเอาต์พุตของ NodeMCU

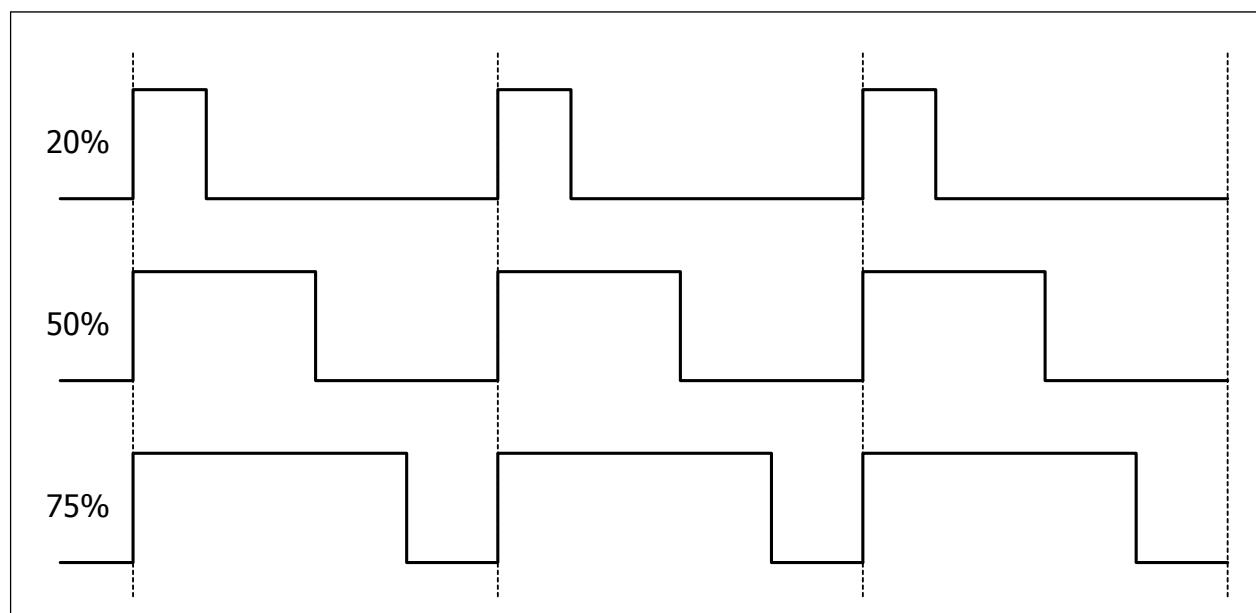
ใน Arduino มีฟังก์ชัน `analogWrite()` ทำให้ NodeMCU-12E สร้างสัญญาณอะนาล็อกส่งออกทางขาพอร์ตเอาต์พุตได้ โดยอาศัยเทคนิคการสร้างสัญญาณ PWM (Pulse Width Modulation) หรือ สัญญาณมอคุเลชั่นทางความกว้างพัลส์ ผู้ใช้งานสามารถปรับค่าดิวตี้ไซเกิลของสัญญาณพัลส์ได้ระหว่าง 0 ถึง 255 เมื่อค่าเป็น 0 แรงดันของขาพอร์ตที่กำหนดไว้จะมีค่าเป็น 0V หากมีค่าเป็น 255 แรงดันที่ขาพอร์ตจะเป็น +3.3V สำหรับค่า 0 ถึง 255 จะทำให้ขาที่กำหนดไว้มีค่าแรงดันเปลี่ยนแปลงสลับไปมาระหว่าง 0 และ +3.3V ถ้ามีค่าสูง ช่วงเวลาที่ขาพอร์ตนั้นมีแรงดัน +3.3V นานแค่ไหน

ถ้าค่าเป็น 51 สัญญาณพัลส์จะมีระดับสัญญาณ +3.3V เป็นเวลานาน 20% ของเวลา และมีแรงดัน 0V นาน 80% ของเวลา หรือมีค่าดิวตี้ไซเกิลเท่ากับ 20% นั่นเอง

ถ้ามีค่าเป็น 127 สัญญาณพัลส์จะมีระดับสัญญาณ +3.3V เป็นเวลานานครึ่งหนึ่งของเวลา และ 0V นานครึ่งหนึ่งของเวลา หรือมีค่าดิวตี้ไซเกิล 50%

ถ้ามีค่าเป็น 191 นั่นคือ สัญญาณพัลส์จะมีระดับสัญญาณ +3.3V เป็นเวลานานสามส่วนสี่ของเวลา และมีแรงดัน 0V นานหนึ่งส่วนสี่ของเวลา หรือมีค่าดิวตี้ไซเกิล 75%

ในรูปที่ 4-10 แสดงสัญญาณ PWM ที่ค่าดิวตี้ไซเกิลต่างๆ



รูปที่ 4-10 แสดงสัญญาณ PWM ที่ค่าดิวตี้ไซเกิลต่างๆ

ค่าแรงดันของสัญญาณพัลส์จะได้เป็นค่าเฉลี่ยของสัญญาณพัลส์ ซึ่งสามารถคำนวณได้จากความสัมพันธ์ทางคณิตศาสตร์ต่อไปนี้

$$\text{แรงดันเอาต์พุต} = (\text{ค่าเวลาของลอดจิกสูง} / \text{ค่าเวลาของลอดจิกต่ำ}) \times \text{ไฟเลี้ยง}$$

ผู้ใช้งานสามารถนำสัญญาณ PWM ที่ได้จากคำสั่งนี้ไปใช้ในการปรับความสว่างของ LED หรือต่อขยายกระแสเพื่อต่อปรับความเร็วของมอเตอร์ได้ หลังจากเรียกใช้คำสั่งนี้ที่ขาพอร์ตที่กำหนด จะมีสัญญาณ PWM ส่งออกมาอย่างต่อเนื่อง จนกว่าจะเรียกใช้ฟังก์ชัน `analogWrite` ในรอบใหม่ หรือเรียกใช้ฟังก์ชัน `digitalRead` หรือ `digitalWrite` ที่ขาพอร์ตเดียวกัน

#### 4.5.3.1 ฟังก์ชันสำหรับสร้างสัญญาณ PWM

ฟังก์ชันใน Arduino IDE ที่ใช้สร้างสัญญาณ PWM คือ `analogWrite()` สำหรับ NodeMCU-12E ทำงานร่วมกับฟังก์ชันนี้ได้เป็นอย่างดี โดยผู้พัฒนาโปรแกรมสำหรับกำหนดให้ขาพอร์ตดิจิตอลทุกขาของ NodeMCU เป็นขาพอร์ตเอาต์พุตเพื่อขับสัญญาณ PWM ได้ทั้งหมด นับเป็นความสามารถที่ยอดเยี่ยมของ NodeMCU

รายละเอียดของฟังก์ชัน `analogWrite()` มีดังนี้

##### รูปแบบ

`analogWrite(pin, value)`

ใช้ในการเขียนค่าอะนาลอกไปยังขาพอร์ตที่กำหนดไว้ เพื่อสร้างสัญญาณ PWM โดยมีค่าความถี่ประมาณ 490Hz

##### พารามิเตอร์

`pin` - หมายเลขขาพอร์ตของ NodeMCU

`value` - เป็นค่าดิจิต์ไซเกิล มีค่าระหว่าง 0 ถึง 255

เมื่อค่าเป็น 0 แรงดันของขาพอร์ตที่กำหนดจะเป็น 0V เมื่อมีค่าเป็น 255 แรงดันที่ขา

พอร์ตจะเป็น +5V สำหรับค่าระหว่าง 0 ถึง 255 จะทำให้ขาพอร์ตที่กำหนดได้มีค่าแรงดันเปลี่ยนแปลงในย่าน 0 ถึง 5V

##### ค่าที่ส่งกลับจากฟังก์ชัน

เลขจำนวนเต็มจาก 0 ถึง 255

ตัวอย่างที่ 4-3

```
#define LED_PIN D1      // LED at D1
int value = 0;          // variable to keep the actual value
void setup()
{
    pinMode(LED_PIN, OUTPUT);
}

void loop()
{
    for (value = 0; value <= 255; value += 5) // fade in (min to max)
    {
        analogWrite(LED_PIN, value);        // sets the value (0 to 255)
        delay(30);                         // waits for 30 milliseconds
    }
    for (value = 255; value >= 0; value -= 5) // fade out (max to min)
    {
        analogWrite(LED_PIN, value);
        delay(30);
    }
    delay(1000);
}
```

ตัวอย่างนี้กำหนดให้ขา D1 ทำงานเป็นขาพอร์ต PWM เพื่อสร้างสัญญาณ PWM ที่มีค่าดิจิต์ใช้เกลจาก 0 ถึง 255 ทำให้ LED ที่ต่ออยู่มีการติดสว่างໄล์จากดับไปสว่างที่สุด จากนั้นเปลี่ยนค่าดิจิต์ใช้เกลเป็นจาก 255 ลดลงไปถึง 0 ทำให้ LED ที่ติดสว่างที่สุด ค่อยๆ หรี่ลงจนดับ จากนั้นการทำงานจะวนกลับไปขับ LED ให้ติดสว่างอีกครั้ง

4.5.3.2 ตัวอย่างโปรแกรมใช้งานขาพอร์ต NodeMCU ขับ LED 3 สี RGB ด้วยสัญญาณ PWM

ใช้วงจรในรูปที่ 4-9 ในการทดลอง

จากนั้นเขียนโปรแกรมที่ 4-3 ทำการคอมไพล์และอัปโหลดไปยัง NodeMCU-12E จากนั้นทำการรันโปรแกรมเพื่อทดสอบการทำงาน

LED 3 สีจะเริ่มต้นด้วยการขับสีแดงก่อน แล้วค่อยๆ เปลี่ยนเป็นสีเหลือง ตามด้วยสีเขียวอ่อน จนถึงเขียวเต็มที่ แล้วเปลี่ยนเป็นเขียวอมฟ้า สีฟ้า สีน้ำเงิน ช่วงพุ่มนิ่ง จากนั้นค่อยๆ เปลี่ยนเป็นสีแดง วนกันไปอย่างร่องรอย การเปลี่ยนแปลงสีมาจากการเปลี่ยนแปลงของสัญญาณ PWM ที่ส่งออกมาจากแต่ละขาพอร์ตของ NodeMCU ตามที่กำหนดไว้ในโปรแกรมที่ 4-3

```

#define B_pin D0 //Blue
#define G_pin D1 //Green
#define R_pin D2 //Red
int st_R = 255;
int st_G = 0;
int st_B = 0;
void setup()
{
    pinMode(B_pin, OUTPUT); // Set pin as output
    pinMode(G_pin, OUTPUT);
    pinMode(R_pin, OUTPUT);
}
void loop()
{
    for (int i = 0; i < 256; i++) // Set loop
    {
        st_B = 0; // Clear BLUE data
        st_G++; // Increase GREEN data
        displayRGB(st_R, st_G, st_B); // Drive LED with PWM
        delay(30); // Short delay
    }
    for (int i = 0; i < 256; i++)
    {
        st_R--; // Decrease RED data
        displayRGB(st_R, st_G, st_B); // Drive LED with PWM
        delay(30); // Short delay
    }
    for (int i = 0; i < 256; i++)
    {
        st_R = 0; // Clear RED data
        st_B++; // Increase BLUE data
        displayRGB(st_R, st_G, st_B); // Drive LED with PWM
        delay(30); // Short delay
    }
    for (int i = 0; i < 256; i++)
    {
        st_G--; // Decrease GREEN data
        displayRGB(st_R, st_G, st_B); // Drive LED with PWM
        delay(30); // Short delay
    }
    for (int i = 0; i < 256; i++)
    {
        st_G = 0; // Clear GREEN data
        st_R++; // Increase RED data
        displayRGB(st_R, st_G, st_B); // Drive LED with PWM
        delay(30); // Short delay
    }
}

```

โปรแกรมที่ 4-3 ไฟล์ **NodeMCU\_LED3C-PWM.ino** โปรแกรมสำหรับใช้งานขาพอร์ตเอาต์พุตของ NodeMCU-12E เพื่อขับ LED 3 ลีด RGB ด้วยสัญญาณ PWM (มีต่อ)

```

for (int i = 0; i < 256; i++)
{
    st_B--;                                // Decrease BLUE data
    displayRGB(st_R, st_G, st_B); // Drive LED with PWM
    delay(30);                            // Short delay
}
}

void displayRGB(int R, int G, int B)
// Drive LED with PWM by using analogWrite function
{
    analogWrite(B_pin, B);
    analogWrite(G_pin, G);
    analogWrite(R_pin, R);
}

```

### คำอธิบายโปรแกรม

เริ่มต้นด้วยการกำหนดขาพอร์ตสำหรับเชื่อมต่อกับขาแອนดของ LED 3 สีก่อนด้วยคำสั่ง `#define` จากนั้นกำหนดค่าสถานะเริ่มต้นของขาพอร์ตทั้งสามขา จะเห็นว่า กำหนดค่าของขาพอร์ตที่ต่อ กับขาแອนดีสี แดงเป็น 255 ส่วนขาอื่นๆ เป็น 0 ในฟังก์ชัน `setup()` เป็นการกำหนดการทำงานของขาพอร์ตเป็นเอาต์พุต

ในโปรแกรมทำงานหลักอยู่ในฟังก์ชัน `loop()` จะเป็นการสร้างลูปเพื่อเพื่อหรือลดค่าตัวแปรที่ใช้ในการสร้างสัญญาณ PWM ให้แก่ขาพอร์ตต่างๆ ที่ใช้ขับ LED 3 สี RGB จากนั้นจึงเรียกใช้ฟังก์ชัน `displayRGB()` เพื่อสร้างสัญญาณ PWM ส่งออกไปยังขาพอร์ตของ NodeMCU-12E เพื่อขับ LED ทำให้ LED ขับแสงที่แตกต่างกัน โดยการเปลี่ยนเป็นสีต่างๆ จะมีความนุ่มนวลและค่อยเป็นค่อยไป ตามการทำงานของสัญญาณ PWM นั้นเอง

ฟังก์ชัน `delay(30);` เป็นฟังก์ชันหน่วงเวลา 30 มิลลิวินาที จึงช่วยให้ผู้พัฒนาโปรแกรมมองเห็นการเปลี่ยนสีของ LED ได้ทัน หากต้องการให้เห็นการเปลี่ยนแปลงอย่างละเอียดมากขึ้น ให้เพิ่มค่าหน่วงเวลา แต่ไม่ควรเพิ่มเกิน 50 เพราะจะทำให้เห็นการเปลี่ยนแปลงที่ช้าเกินไป

---

**โปรแกรมที่ 4-3 ไฟล์ NodeMCU\_LED3C-PWM.ino โปรแกรมสำหรับใช้งานขาพอร์ตเอาต์พุตของ NodeMCU-12E เพื่อขับ LED 3 สี RGB ด้วยสัญญาณ PWM (มีต่อ)**

## 4.5.4 ใช้งานพอร์ตอินพุตดิจิตอล

### 4.5.4.1 คุณสมบัติของขาพอร์ตอินพุต

ขาพอร์ตของ NodeMCU เมื่อถูกกำหนดเป็นอินพุตดิจิตอล จะมีสถานะเป็นอิมพีเดนซ์สูง ทำให้มีความต้องการกระแสไฟฟ้าจากอุปกรณ์ที่ต้องการอ่านค่าอินพุตน้อยมาก ไม่สามารถรับหรือจ่ายกระแสไฟกับวงจรภายนอก

สำหรับขาอินพุต เมื่อไม่มีอินพุตป้อนให้จะต้องกำหนดค่าแรงดันให้แน่นอน ทำได้โดยต่อตัวต้านทานพูลอัป (pull-up resistor) โดยต่อขาของตัวต้านทานขาหนึ่งไปยังไฟเลี้ยง หรือต่อพูลดาวน์ (pull-down) ซึ่งต่อขาหนึ่งของตัวต้านทานจากขาพอร์ตลงกราวด์ ค่าตัวต้านทานที่ใช้หัวไปคือ  $10\text{k}\Omega$  ดังรูปที่ 4-11

ขาพอร์ตดิจิตอลของ NodeMCU รองรับการกำหนดให้เป็นอินพุตได้ทั้งหมด เมื่อต้องการกำหนดเป็นอินพุต ดิจิลต้องกำหนดด้วยฟังก์ชัน `pinMode` และอ่านค่าอินพุตได้จากฟังก์ชัน `digitalRead` ซึ่งมีรูปแบบดังนี้

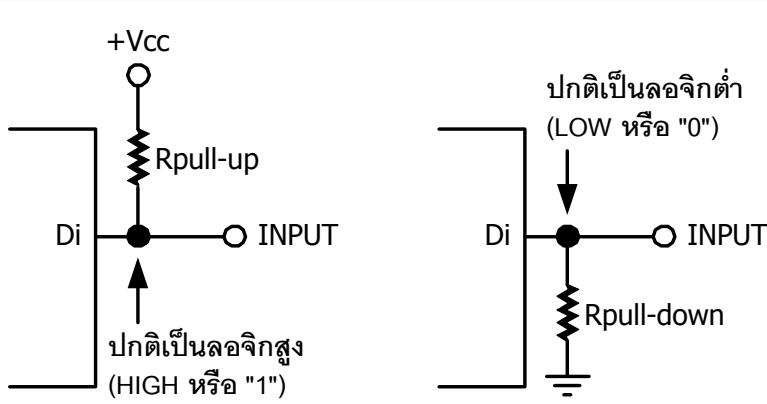
```
pinMode(pin, INPUT);
```

เมื่อ `pin` คือ หมายเลขขาพอร์ตที่ต้องการ

```
digitalRead(pin);
```

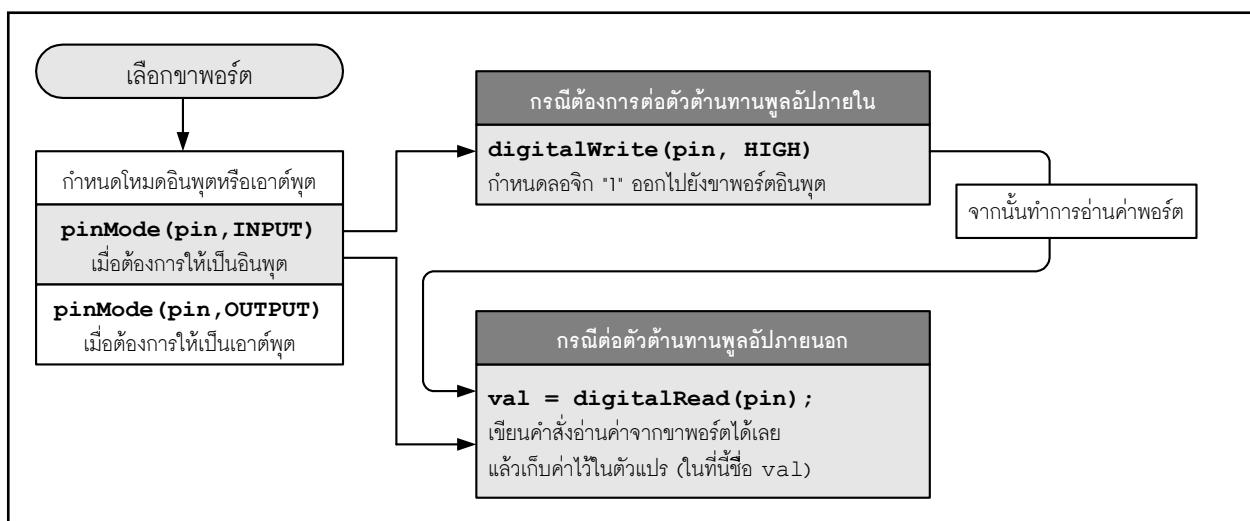
เมื่อ `pin` คือหมายเลขขาที่ต้องการอ่านค่าสถานะ

เมื่อฟังก์ชันทำงาน จะคืนค่าเป็นสถานะของขาที่ต้องการอ่านค่า โดยคืนค่าเป็น LOW (ค่าเป็น “0”) หรือ HIGH (ค่าเป็น “1”)



(ก) การต่อตัวต้านทานพูลอัป      (ข) การต่อตัวต้านทานพูลดาวน์

รูปที่ 4-11 แสดงการต่อตัวต้านทานเพื่อกำหนดสถานะของขาพอร์ตอินพุตของไมโครคอนโทรลเลอร์ ในขณะที่ยังไม่มีอินพุตส่งเข้ามา



รูปที่ 4-12 ໄດ້ອະແກມແສດງກລິກາຮກການກໍາເນດໃຫ້ຂາພອົບຂອງ NodeMCU-12E ເປັນພອົບອີນພຸຕ ດິຈິຕອລ (ບລືອກສື່ເຫາແທນກາຮກທຳການຫຼືເງື່ອນໄຂທີ່ຕ້ອງການ)

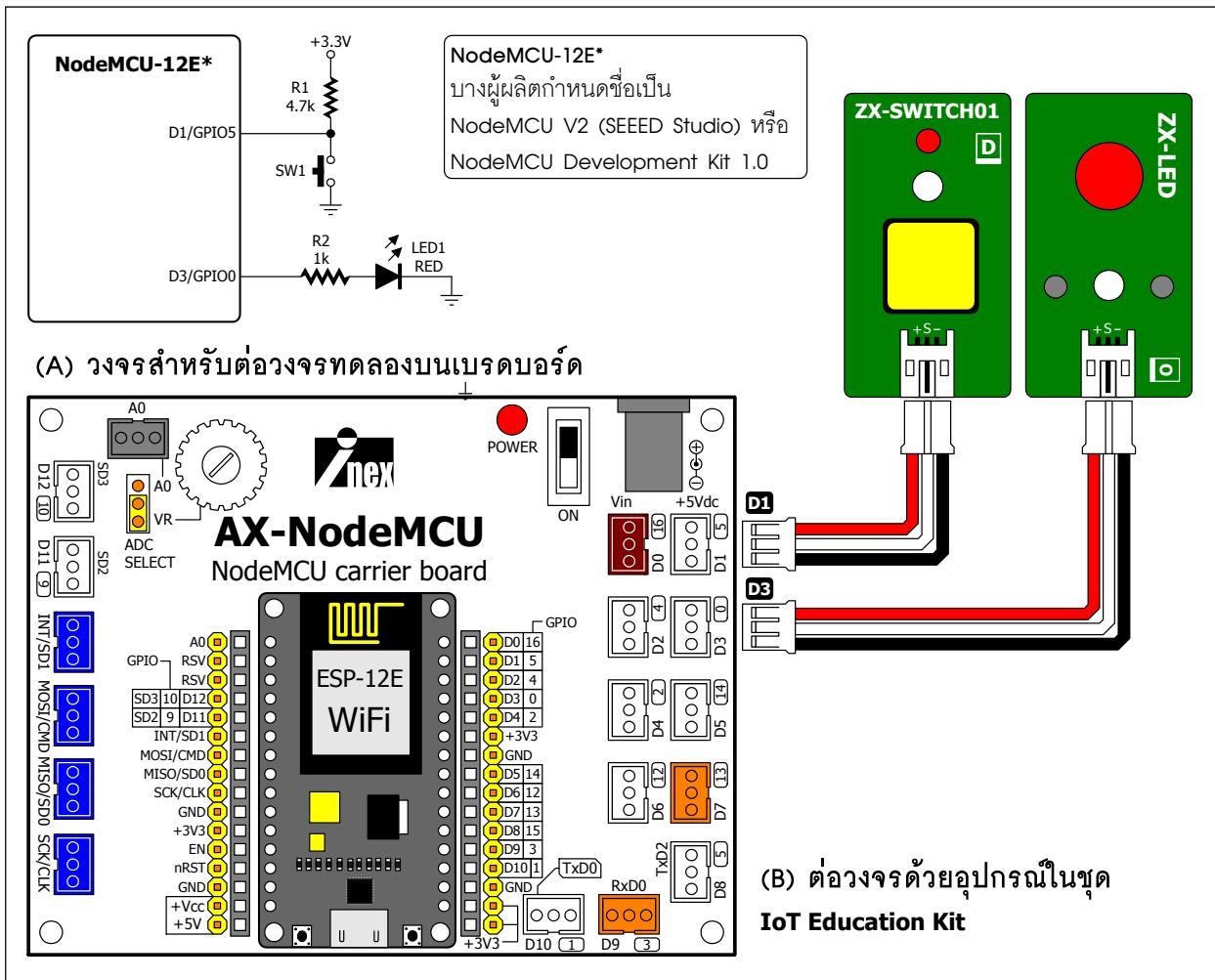
ในรูปที่ 4-12 ແສດງໄດ້ອະແກມກາຮກເຂົ້າສຳເນົາກໍາເນດໃຫ້ຂາພອົບຂອງ NodeMCU-12E ທຳການເປັນພອົບອີນພຸຕ ດິຈິຕອລ

#### 4.5.4.2 ຖດລອງຮັບຄ່າສວິຕີໜ້ອຍໆງ່າຍ

ໃນກາຮກທຳການນີ້ຈະຖດລອງອ່ານຄ່າສຕານະຂອງສວິຕີໜ້ອຍໆງ່າຍ (LED) ເມື່ອກົດສວິຕີໜ້ອຍໆງ່າຍ ທຳໃຫ້ LED ຕິດສ່ວ່າງ ເມື່ອປ່ອລ່ອຍສວິຕີໜ້ອຍໆງ່າຍ LED ຈະດັບ ໂດຍມີວິຈາແສດງໄດ້ດັ່ງຮູບທີ່ 4-13 ແລະເຂົ້າສຳເນົາໂປຣແກຣມໄດ້ດັ່ງໂປຣແກຣມທີ່ 4-4

ໃນກາຮກໃຫ້ຂາພອົບອີນພຸຕ ດິຈິຕອລ ທີ່ຕ້ອງຕໍ່ອັນດັບຕໍ່ອັນດັບ (ຕໍ່ອັນດັບຈາກໄຟເລີ່ຍ +3.3V ນາຍັງຫາອີນພຸຕ) ເພື່ອກົດສວິຕີໜ້ອຍໆງ່າຍ ທີ່ແນ່ນອນໃຫ້ກັບຫາອີນພຸຕ ບໍ່ໄມ້ມີກົດສວິຕີໜ້ອຍໆງ່າຍ ດັງວ່າໃນຮູບທີ່ 4-13 (A) ເມື່ອໄມ້ໄດ້ກົດສວິຕີໜ້ອຍໆງ່າຍ ທີ່ขาพอร์ຕ D1 ຈະມີສຕານະເປັນລອອິກສູງ ຮູ່ວິກ ຢ່າງ (HIGH) ເຊິ່ງ “1” ເມື່ອກົດສວິຕີໜ້ອຍໆງ່າຍ ຈະທຳໃຫ້ขา D1 ຕ່ອລົງກຽວດ້ວຍຕໍ່ອັນດັບຕໍ່ອັນດັບ ອ່ານຄ່າສຕານະເປັນລອອິກຕໍ່າ ຮູ່ວິກ (LOW) ເຊິ່ງ “0”

ກາຮກທຳການຂອງ LED1 ຈະຕຽບຂໍ້ມູນກັບສຕານະຂອງສວິຕີໜ້ອຍໆງ່າຍ ເຊິ່ງ “1” ເຊິ່ງ “0” ພໍອອົງກົດສວິຕີໜ້ອຍໆງ່າຍ ຈະອ່ານສຕານະຂອງຫາ D1 ໄດ້ລອອິກສູງ ຊຶ່ງຕ້ອງສ້າງໃຫ້ຫາ D3 ເປັນລອອິກຕໍ່າ ຮູ່ວິກ (HIGH) ເຊິ່ງ “1” ເພື່ອກົດສວິຕີໜ້ອຍໆງ່າຍ ອ່ານຄ່າສຕານະຂອງຫາ D1 ໄດ້ລອອິກ “0” ຕ້ອງສ້າງໃຫ້ຫາ D3 ເປັນ “1” ເພື່ອຂັບ LED1 ຕິດສ່ວ່າງ



รูปที่ 4-13 วงจรสำหรับทดลองใช้งานขาพอร์ตอินพุตของ NodeMCU-12E ในการรับค่าสวิตช์เพื่อขับ LED

```
#define sw1 D1                                // Define D1 as switch pin
#define ledPin1 D3                             // Define D1 as LED pin
int st_sw1 = 0;
void setup()
{
    pinMode(ledPin1, OUTPUT);                  // Set pin as output
    pinMode(sw1, INPUT);                      // set pin as input
}
void loop()
{
    st_sw1 = digitalRead(sw1);                // Read input
    digitalWrite(ledPin1, ~st_sw1); // Invert output
}
```

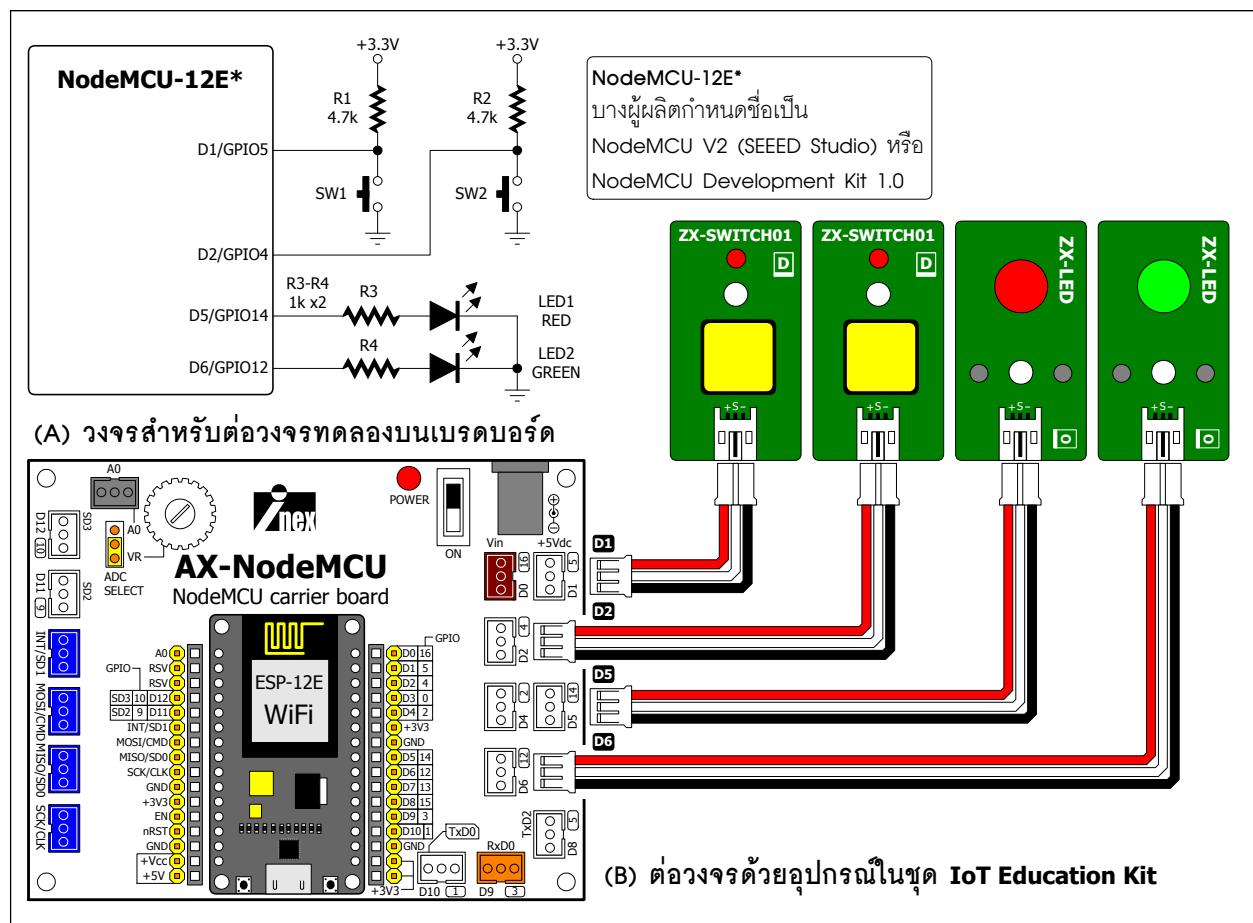
โปรแกรมที่ 4-4 ไฟล์ NodeMCU\_SW.ino โปรแกรมทดลองใช้งานขาพอร์ตอินพุตของ NodeMCU-12E ในการรับค่าสวิตช์เพื่อขับ LED

#### 4.5.4.3 ทดลองรับค่าสวิตช์ 2 ตัว

ในการทดลองนี้จะทดลองอ่านค่าสถานะของสวิตช์แบบกดติดปล่อยดับเพื่อควบคุม LED เมื่อกดสวิตช์ทำให้ LED ติดสว่าง เมื่อกดอีกครั้ง LED จะดับ โดยมีวงจรแสดงได้ดังรูปที่ 4-14 และเขียนเป็นโปรแกรมได้ดังโปรแกรมที่ 4-5

ในตัวอย่างนี้จะเพิ่มสวิตช์เป็น 2 ตัว โดยขาพอร์ตของ NodeMCU ที่ต่อ กับสวิตช์ต้องต่อตัว ต้านทาน pull-up เพื่อกำหนดสถานะที่แน่นอนให้กับขาอินพุตในภาวะที่ไม่มีการกดสวิตช์ ดังวงจรในรูปที่ 4-14 (A) หากใช้อุปกรณ์ในชุด IoT Education Kit ให้ทำการต่อวงจรตามรูปที่ 4-14 (B)

การทำงานของ LED ทั้งสองดวงจะเปลี่ยนตามการกดปล่อยสวิตช์ 1 ครั้ง นั่นคือ เมื่อกดครั้งแรก LED ติดสว่างค้าง เมื่อกดสวิตช์อีกครั้ง LED จะดับ โดยสวิตช์ที่ต่อ กับขาพอร์ต D1 ใช้ควบคุม LED ที่ขาพอร์ต D5 และสวิตช์ที่ต่อ กับขาพอร์ต D2 ใช้ควบคุม LED ที่ขาพอร์ต D6



รูปที่ 4-14 วงจรสำหรับทดลองใช้งานขาพอร์ตอินพุตของ NodeMCU-12E ในการรับค่าสวิตช์ 2 ตัวเพื่อขับ LED 2 ตัวในแบบกดติด กดดับ

```

#define sw1 D1           // Declare all variables
#define sw2 D2
#define ledPin1 D5
#define ledPin2 D6
int st_sw1 = 0;          // Set initial status
int st_sw2 = 0;
int st_1 = 0;
int st_2 = 0;
int last_st_sw1 = 1;
int last_st_sw2 = 1;

void setup()
{
    pinMode(ledPin1, OUTPUT); // Set pin mode
    pinMode(ledPin2, OUTPUT);
    pinMode(sw1, INPUT);
    pinMode(sw2, INPUT);
}

void loop()
{
    st_sw1 = digitalRead(sw1);           // Read input port1
    if ((st_sw1 == 0) && (last_st_sw1 == 1)) // Check current status
    {
        st_1 = ~st_1;                  // Toggle
        digitalWrite(ledPin1, st_1);    // Drive LED1
        delay(250);
    }
    last_st_sw1 = st_sw1;               // Update current status

    st_sw2 = digitalRead(sw2);           // Read input port2
    if ((st_sw2 == 1) && (last_st_sw2 == 0)) // Check current status
    {
        st_2 = ~st_2;                  // Toggle
        digitalWrite(ledPin2, st_2);    // Drive LED2
        delay(250);
    }
    last_st_sw2 = st_sw2;               // Update current status
}

```

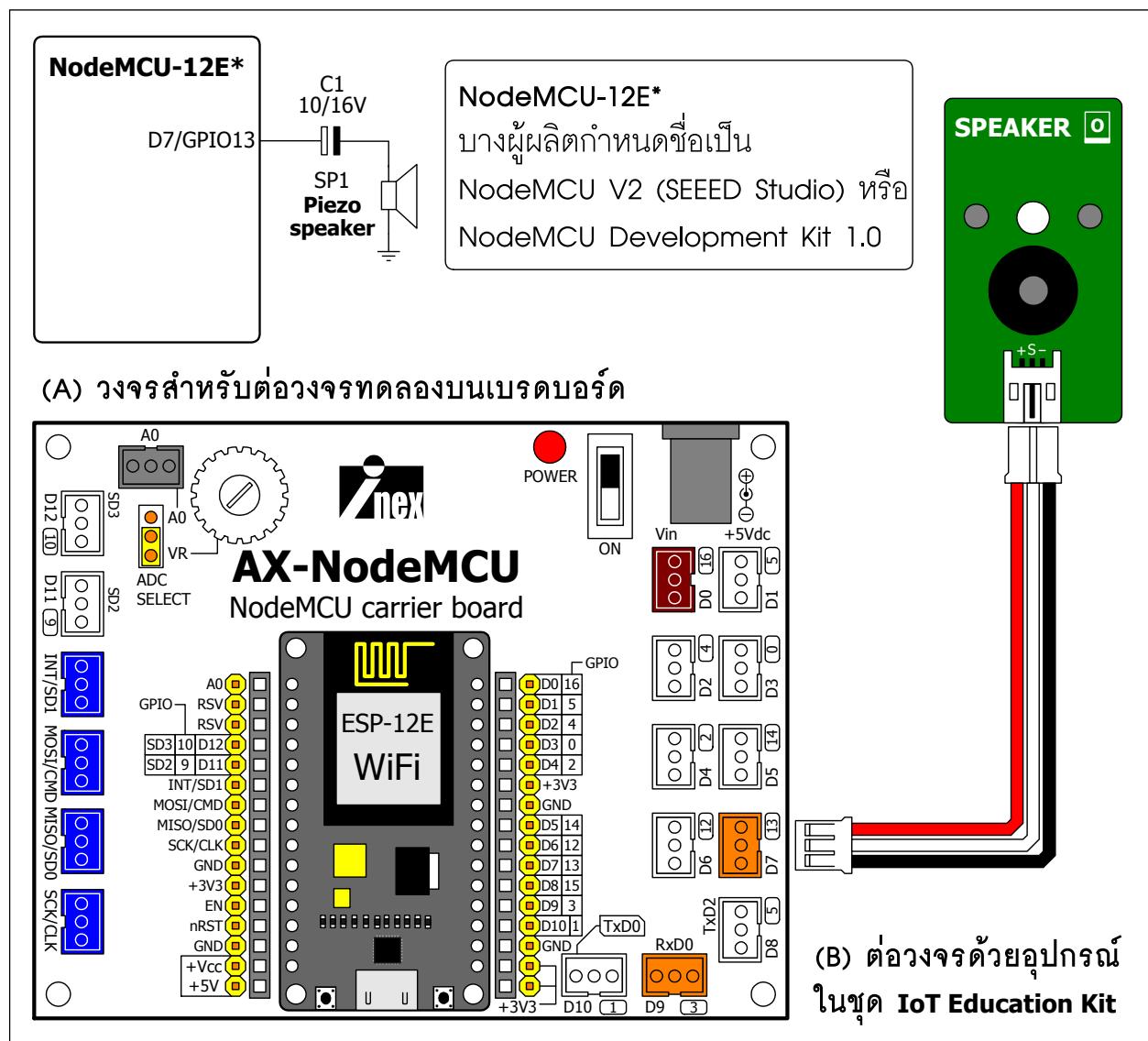
**โปรแกรมที่ 4-5 ไฟล์ NodeMCU\_SW2.ino** โปรแกรมทดลองใช้งานขาพอร์ตอินพุตของ NodeMCU-12E  
ในการรับค่าสวิตช์เพื่อขับ LED ในแบบท็อกเกิลหรือกดติดกดดับ

#### 4.5.5 NodeMCU กับสร้างสัญญาณเสียง

การสร้างเสียงเพื่อขับลำโพงที่เป็นชุดวงหรือลำโพงแบบเปียโซ NodeMCU-12E จะต้องสร้างสัญญาณกระแสลับที่กระแสไฟฟ้าสามารถไหลกลับทิศได้ จึงต้องใช้สัญญาณจากขาพอร์ตไมโครคอนโทรลเลอร์ 2 เส้น หรือถ้าต้องการประหยัดจะใช้สัญญาณเพียงเส้นเดียวที่ได้ แต่ต้องค่าตัวเก็บประจุอนุกรมกับขาเอาต์พุตที่ใช้ขับสัญญาณเสียงด้วย

อุปกรณ์ที่เพิ่มเข้ามาในการทดลองนี้คือ ZX-SPEAKER แผงวงจรลำโพงเปียโซ

ฟังก์ชันที่ใช้ในการกำหนดสัญญาณเสียงคือ `analogWrite(pin, DutyCycle)` ; โดยเลือกใช้ขาพอร์ตดิจิตอล (pin) ของ NodeMCU ชาใดก็ได้ ส่วนค่า DutyCycle กำหนดได้ตั้งแต่ 0 ถึง 1023 ค่าความถี่ตั้งต้นคือ 1kHz



รูปที่ 4-15 วงจรสำหรับทดลองใช้งานขาพอร์ตเอาต์พุตของ NodeMCU-12E ในการขับสัญญาณเสียง

```

int pinTone = D7;
void setup()
{
    analogWrite(pinTone, 255); // Drive the default frequency 1kHz
    delay(1000); // 1 second
}
void loop()
{
    analogWriteFreq(440); // Change frequency to 440Hz
    delay(500); // Drive 0.5 second
    analogWriteFreq(587); // Change frequency to 587Hz
    delay(500); // Drive 0.5 second
}

```

**โปรแกรมที่ 4-6** ไฟล์ **NodeMCU\_Tone.ino** โปรแกรมทดลองใช้งานขาพอร์ตเอาต์พุตของ **NodeMCU-12E** ขับสัญญาณเสียง

หากต้องการเปลี่ยนความถี่ของสัญญาณเสียง ทำได้โดยใช้ฟังก์ชัน **analogWriteFreq(NewFreq)** ; กำหนดค่าความถี่ใหม่ที่ต้องการลงในพารามิเตอร์ **NewFreq**

ในการทดลองนี้จะสร้างเสียง 3 ความถี่คือ 1kHz, 440 และ 587Hz ออกที่ลำโพงเปียโซ โดยเริ่มต้นด้วยสัญญาณความถี่ 1kHz นาน 1 วินาที จากนั้นจะเป็นสัญญาณความถี่ 440 และ 587Hz ความถี่ละ 0.5 วินาที สลับกันตลอดเวลา ใช้วงจรในรูปที่ 4-15 และ โปรแกรมที่ 4-6 ในการทดลอง

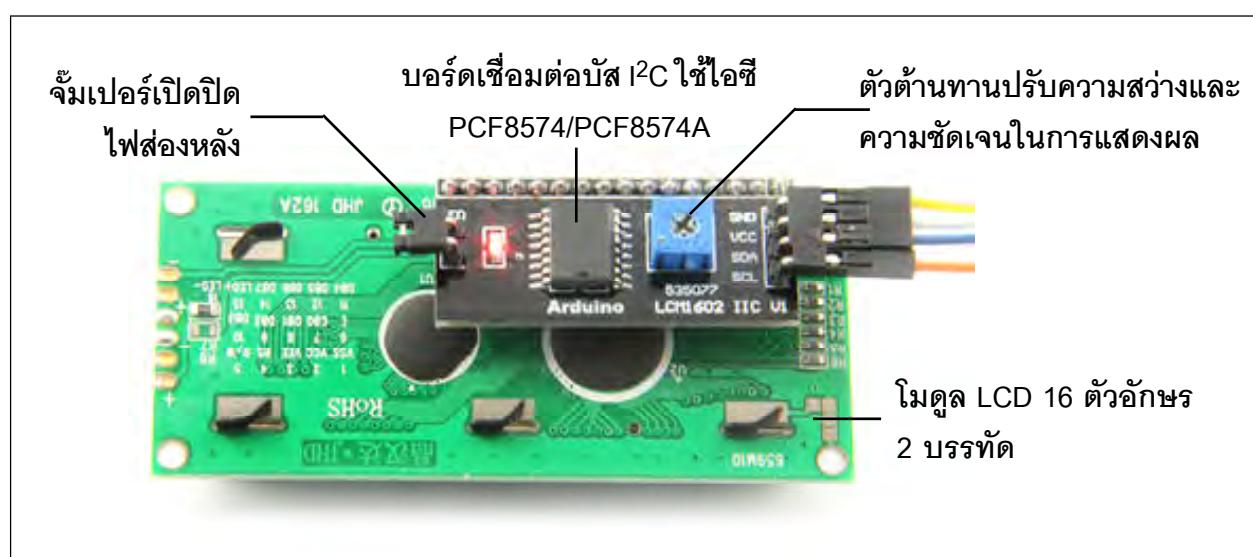
## 4.5.6 เชื่อมต่อ NodeMCU-12E กับโมดูล LCD 16 ตัวอักษร 2 บรรทัดผ่านบัส I<sup>2</sup>C

เนื่องจาก NodeMCU-12E มีพอร์ตอินพุตเอาต์พุตดิจิตอลบนบอร์ดจำนวน 13 ขา (ไม่รวมขาเชื่อมต่อบัส SPI จำนวน 3 ขา และอินพุตอินเตอร์รัปต์ 1 ขา) ดังนั้นการเชื่อมต่อกับอุปกรณ์แสดงผลสมัยใหม่ อาทิ โมดูล LCD หรือ โมดูลกราฟิก LCD และ โมดูล OLED จึงควรเลือกใช้แบบที่ใช้จำนวนขาพอร์ตเพื่อการติดต่อไม่มาก ในที่นี้ขอแนะนำ โมดูล LCD 16 ตัวอักษร 2 บรรทัดที่ติดต่อผ่านบัส I<sup>2</sup>C หรือบัสสองสาย เรียกว่า โมดูล I<sup>2</sup>C-LCD16x2

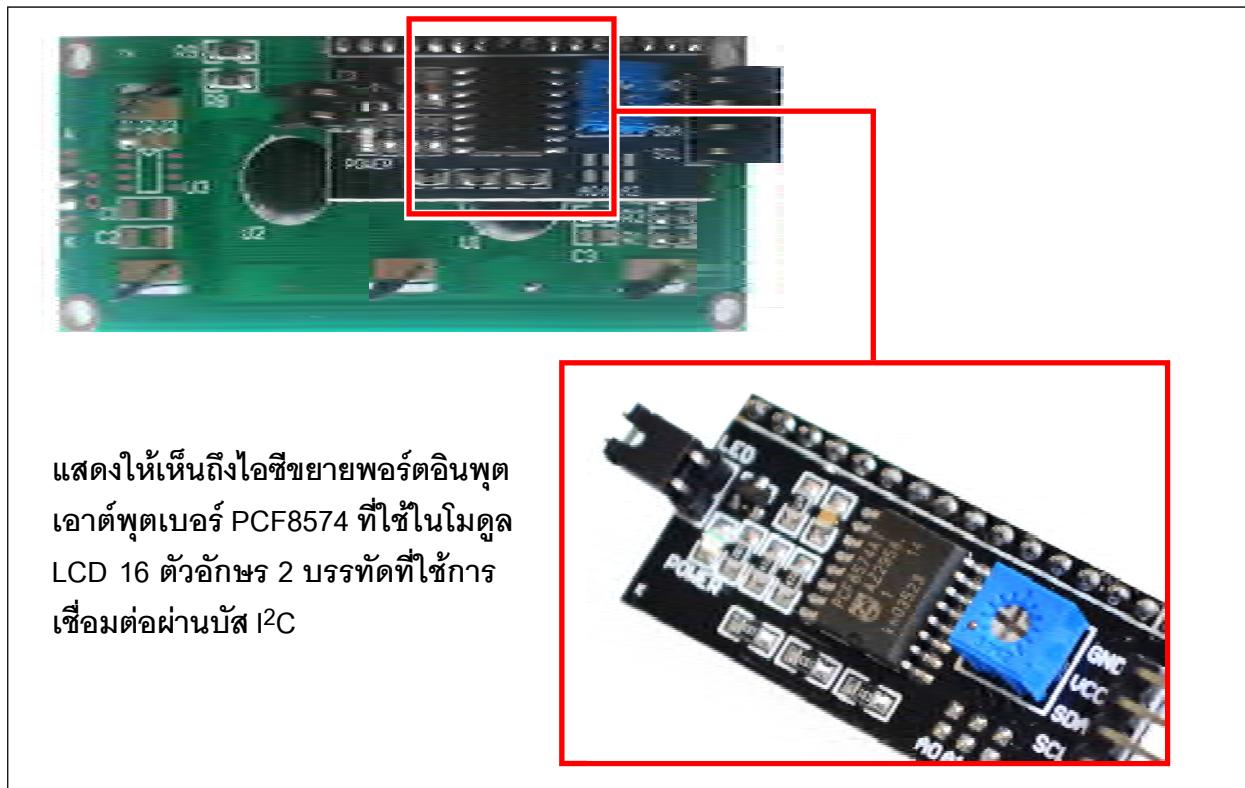
### 4.5.6.1 โครงสร้างทางฮาร์ดแวร์ของโมดูล I<sup>2</sup>C-LCD16x2

โดยแท้จริงแล้ว โมดูล I<sup>2</sup>C-LCD16x2 ที่คือ โมดูล LCD 16 ตัวอักษร 2 บรรทัดแบบมีไฟส่องหลัง ที่เป็นอุปกรณ์มาตรฐานนำมาเชื่อมต่อ กับบอร์ดขยายพอร์ตอินพุตเอาต์พุตผ่านบัส I<sup>2</sup>C ที่ใช้ไอซีเบอร์ PCF8574 และ/หรือ PCF8574A ดังแสดงในรูปที่ 4-16 โดยบอร์ดขยายพอร์ตนี้จะถูกบัดกรีประกอบเข้ากับด้านหลังของ โมดูล LCD บนบอร์ดมีตัวต้านทานปรับค่าได้เพื่อใช้ในการปรับความสว่างหรือความชัดเจนในการแสดงผล และมีจีนเปอร์เพื่อเลือกเปิดปิด LED ที่ใช้เป็นไฟส่องหลังของจอแสดงผล

การต่อใช้งานกระทำผ่านบัส I<sup>2</sup>C ด้วยขาสัญญาณ SDA (Serial Data) และ SCL (Serial Clock) พร้อมกับต่อไฟเลี้ยง +5V และกราวด์



รูปที่ 4-16 แสดงโครงสร้างทางฮาร์ดแวร์ของโมดูล I<sup>2</sup>C-LCD16x2 โมดูล LCD 16 ตัวอักษร 2 บรรทัดที่ติดต่อผ่านบัส I<sup>2</sup>C



รูปที่ 4-17 แสดงเบอร์ของไอซีบนบอร์ดเชื่อมต่อบัส I<sup>2</sup>C ที่ติดตั้งอยู่ด้านหลังของโมดูล I2C-LCD16x2  
โมดูล LCD 16 ตัวอักษร 2 บรรทัดที่ติดต่อผ่านบัส I<sup>2</sup>C

#### 4.5.6.2 แอดเดรสสำหรับการติดต่อ

อุปกรณ์ที่เชื่อมต่อกับบัส I<sup>2</sup>C ทุกตัวจะต้องมีค่าแอดเดรสประจำตัว เพื่อให้อุปกรณ์มาสแตอร์ หรือไมโครคอนโทรลเลอร์ (ในที่นี้คือ NodeMCU-12E) เลือกติดต่อได้อย่างถูกต้อง ทั้งนี้เนื่องจาก อุปกรณ์บนบัส I<sup>2</sup>C จะต่อพ่วงสายสัญญาณ SDA และ SCL ร่วมกัน ไม่ว่าจะมีอุปกรณ์กี่ตัวก็ตาม (สูงสุด 128 ตัว)

สำหรับโมดูล I2C-LCD16x2 มีค่าแอดเดรส 2 ค่า เนื่องจากบอร์ดเชื่อมต่อใช้ไอซี 2 เบอร์คือ PCF8574 และ PCF8574A ซึ่งทั้งสองตัวมีค่าแอดเดรสที่ต่างกัน โดยปกติโมดูล I2C-LCD16x2 จะใช้ แอดเดรส 0x27 เป็นหลัก (เนื่องจากใช้ไอซี PCF8574) สำหรับโมดูลที่ใช้ไอซี PCF8574A จะมีค่าแอดเดรสเป็น 0x3F

การสังเกตทำได้ง่ายมาก เพียงพลิกไปดูที่บอร์ดเชื่อมต่อ ไอซีตัวไหนๆ ที่สุดที่อยู่กึ่งกลางบอร์ดคือ ไอซี PCF8574 หรือ PCF8574A ใช้วิธีนี้ขยายดูที่บอร์ดของไอซี จะเห็นเบอร์ PCF8574 หรือ PCF8574A หากเบอร์ลับเลือนไป การตรวจสอบจะต้องเขียนโปรแกรมติดต่อ โดยให้เลือกค่าแอดเดรสเป็น 0x27 ก่อน หากติดต่อไม่ได้จะเปลี่ยนเป็น 0x3F

#### 4.5.6.3 ติดตั้งไลบรารี LiquidCrystal-I2C

ในการใช้งานโมดูล I2C-LCD16x2 กับ NodeMCU-12E และใช้ซอฟต์แวร์ Arduino IDE 1.6.5r2 ในการพัฒนาโปรแกรมจะต้องติดตั้งไลบรารี **LiquidCrystal\_I2C** เสียก่อน โดยดาวน์โหลดได้ที่ <https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library/archive/master.zip>

เมื่อดาวน์โหลดมาแล้วให้ดำเนินการดังนี้

- (1) ทำการแตกไฟล์ จะได้โฟลเดอร์ **Arduino-LiquidCrystal-I2C-library-master**
- (2) เปลี่ยนชื่อโฟลเดอร์เป็น **LiquidCrystal-I2C**
- (3) คัดลอกไปยัง **C:\Arduino1.6.5r2\libraries**

สำหรับไลบรารี **LiquidCrystal-I2C** ต้องใช้งานร่วมกับไลบรารี **Wire** ซึ่งเป็นไลบรารีหลักของซอฟต์แวร์ จะต้องมีการผนวกไฟล์ไลบรารีทั้งสองนี้ลงในตอนต้นของโปรแกรมเสมอด้วยคำสั่ง  
`#include`

นอกจากนี้ ไลบรารี **LiquidCrystal-I2C** กำหนดให้ใช้ขาพอร์ตของ NodeMCU-12E ดังนี้

ขาพอร์ต **D1/GPIO5** เป็นขาสัญญาณ **SCL**

ขาพอร์ต **D2/GPIO4** เป็นขาสัญญาณ **SDA**

#### 4.5.6.4 ตัวอย่างโปรแกรมแสดงผลบนโมดูล I2C-LCD16x2 อย่างง่าย

หลังจากติดตั้ง ไลบรารี **LiquidCrystal-I2C** เรียบร้อยแล้ว ทำการเชื่อมต่ออุปกรณ์ตามรูปที่ 4-17

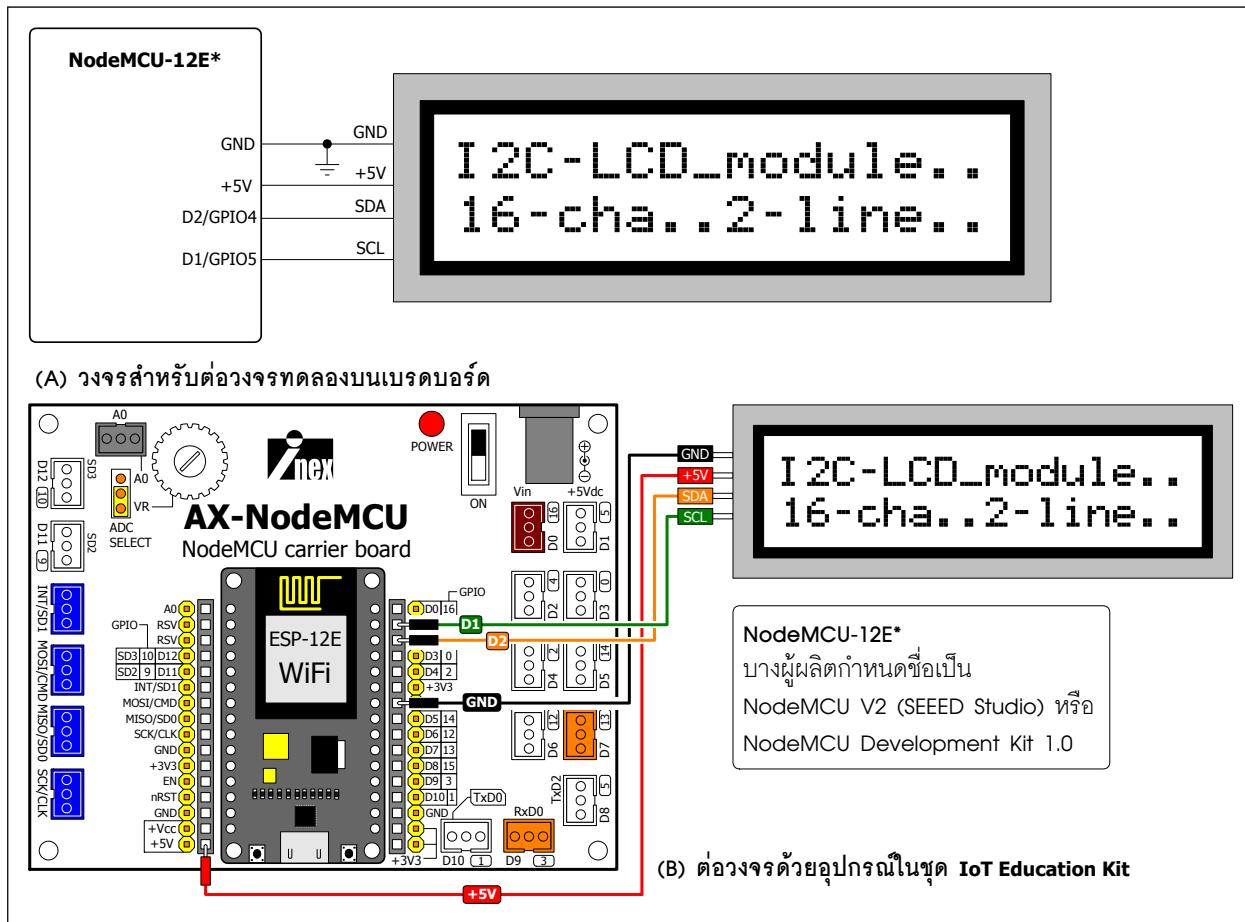
จากนั้นเขียนโปรแกรมที่ 4-7 แล้วอปปะโหลดไปยัง Node MCU ทำการรันโปรแกรม สังเกตการแสดงผลที่หน้าจอ LCD

เมื่อรันโปรแกรม ที่ข้อแสดงผลของโมดูล LCD แสดงข้อความ

**Line 1 Hello**

**Line 2 Hello**

โดยการเริ่มต้นแสดงผล จะเริ่มที่ตำแหน่งอักษรตัวแรกในบรรทัดบน จากนั้นแสดงข้อความ **Line 2 Hello** ที่บรรทัดล่าง โดยกำหนดให้เริ่มต้นที่อักษรตัวที่ 4



รูปที่ 4-18 วงจรทดลองการเชื่อมต่อ NodeMCU-12E กับโมดูล LCD ผ่านบัส I<sup>2</sup>C

```

//SCL as D1/GPIO5
//SDA as D2/GPIO4
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Initial I2C-LCD
// Address is 0x27 (for PCF8574) or 0x3F (for PCF8574A)
// Type 16 characters 2 lines
LiquidCrystal_I2C lcd(0x3F, 16, 2);
void setup()
{
    lcd.begin(); // Start
    lcd.backlight(); // Enable LED backlight
    lcd.setCursor(0, 0); // Set home cursor
    lcd.print("Line 1 Hello "); // Display message on line 1 (upper)
    lcd.setCursor(3, 1); // Set new position
    lcd.print("Line 2 Hello "); // Display message on line 2 (lower)
}
void loop
{ }

```

โปรแกรมที่ 4-7 ไฟล์ NodeMCU\_I2CLCD.ino ทดสอบการเชื่อมต่อ NodeMCU-12E กับโมดูล LCD ผ่านบัส I<sup>2</sup>C

