รายงานการปฏิบัติงานสหกิจศึกษา

การออกแบบและการติดตั้งระบบควบคุมแสงสว่างของโรงงาน โดยใช้อินเตอร์เน็ตของสรรพสิ่ง

Design and Installation of a Lighting Control System of Factory Using Internet of Things (IoT)

โดย

นาย จรัญ รักญาติ 6223200025

รายงานนี้เป็นส่วนหนึ่งของวิชาสหกิจศึกษาวิศวกรรมไฟฟ้า

ภาควิชาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสยาม

ภาคการศึกษา 1 ปีการศึกษา 2564

หัวข้อโครงงาน	การออกแบบและการติดตั้งระบบควบคุมแสงสว่างของโรงงานโดยใช้อินเตอร์เน็ตของสรรพสิ่ง Design and Installation of a Lighting Control System of Factory Using Internet of Things (IoT)
รายชื่อผู้จัดทำ	นาย จรัญ รักญาติ 6223200025
ภาควิชา	วิศวกรรมไฟฟ้า
อาจารย์ที่ปรึกษา	ผศ.ดร. ยงยุทธ นารายณ์

อนุมัติให้โครงงานนี้เป็นส่วนหนึ่งของการปฏิบัติงานสหกิจศึกษาภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสยาม ภาคการศึกษาที่ 1 ปีการศึกษา 2564

คณะกรรมการสอบโครงงาน

..... อาจารย์ที่ปรึกษา
(ผศ.ดร. ยงยุทธ นารายณ์)

..... พนักงานที่ปรึกษา
(นางสาว ชาคริยา ณ นคร)

..... กรรมการกลาง
(ผศ.วิภาวัลย์ นาคทรัพย์)

..... ผู้ช่วยอธิการบดีและผู้อำนวยการสำนักสหกิจศึกษา
(ผศ.ดร. มารุจ ลิมปะวัฒนะ)

จดหมายนำส่งรายงาน

วันที่ 10 ธันวาคม พ.ศ. 2564

เรื่อง ขอส่งรายงานการปฏิบัติงานสหกิจศึกษา

เรียน อาจารย์ที่ปรึกษาสหกิจศึกษา ภาควิชาวิศวกรรมไฟฟ้า
ผู้ช่วยศาสตราจารย์ ดร.ยงยุทธ นาราษฎร์

ตามที่ ผู้จัดทำ นายจรัญ รักญาติ รหัสนักศึกษา 6223200025 นักศึกษาภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์มหาวิทยาลัยสยามได้ไปปฏิบัติงานสหกิจศึกษาระหว่างวันที่ 23 สิงหาคม พ.ศ.2564 ถึง วันที่ 10 ธันวาคม พ.ศ.2564 ในตำแหน่งซ่อมบำรุงระบบไฟฟ้าและระบบสื่อสารให้กับ บริษัท เอส คิว ดี มาร์เก็ตติ้งจำกัดรวมไปถึงบริษัทในเครือของบริษัท เอส คิว ดี มาร์เก็ตติ้งจำกัด และได้รับมอบหมายงานจาก (หัวหน้าพนักงานที่ปรึกษา) ที่ให้ศึกษาและทำรายงานค้นคว้าเกี่ยวกับการออกแบบและการติดตั้งระบบควบคุมแสงสว่างของโรงงานโดยใช้อินเตอร์เน็ตของสรรพสิ่ง

บัดนี้การปฏิบัติงานสหกิจศึกษาได้สิ้นสุดลง ผู้จัดทำ จึงขอส่งรายงานดังกล่าวมาพร้อมกันนี้จำนวน 1 เล่ม เพื่อขอรับคำปรึกษาต่อไป

ขอแสดงความนับถือ

นาย จรัญ รักญาติ

นักศึกษาสหกิจศึกษา ภาควิชาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสยาม

กิตติกรรมประกาศ

(Acknowledgement)

การที่ผู้จัดทำได้มาปฏิบัติงานในโครงการสหกิจศึกษา ณ บริษัทเอสคิวดีมาร์เก็ตติ้งจำกัดตั้งแต่วันที่ 23 สิงหาคม พ.ศ.2564 ถึง วันที่ 10 ธันวาคม พ.ศ.2564รวมทั้งสิ้น 16 สัปดาห์ ส่งผลให้ผู้จัดทำได้รับความรู้และประสบการณ์การทำงานต่างๆสำหรับรายงานสหกิจศึกษาฉบับนี้สำเร็จลงได้ด้วยดีจากความร่วมมือและผู้สนับสนุนดังนี้

1) คุณสุพจน์ เอี่ยมวงศ์ทิรัญ Managing Director
2) คุณชาคริยา ณ นคร หัวหน้าพนักงานที่ปรึกษา
3) ผศ.ดร.ยงยุทธ นาราษฎร์ อาจารย์ที่ปรึกษาสหกิจศึกษา

และบุคคลท่านอื่นๆที่ไม่ได้กล่าวนามและทุกท่านที่ได้ให้คำแนะนำช่วยเหลือในการจัดทำรายงาน ผู้จัดทำขอขอบพระคุณผู้ที่มีส่วนเกี่ยวข้องทุกท่านที่มีส่วนร่วมในการให้ข้อมูลและเป็นที่ปรึกษาในการทำรายงานฉบับนี้จนเสร็จสมบูรณ์ตลอดจนให้การดูแลให้ความเข้าใจในชีวิตการทำงานจริงซึ่งหากรายงานฉบับนี้มีข้อผิดพลาดประการใดผู้จัดทำก็ขออภัยมา ณ ที่นี้

ผู้จัดทำ
นายจรัญ รักญาติ
วันที่ 28 มิถุนายน 2566

หัวข้อโครงงาน	การออกแบบและการติดตั้งระบบควบคุมแสงสว่างของโรงงานโดยใช้ อินเตอร์เน็ตของสรรพสิ่ง
ผู้จัดทำ	นายจรัญ รักญาติ 6223200025
อาจารย์ที่ปรึกษา	ผศ. ดร. ยงยุทธ นาราษฎร์
หลักสูตร	ปริญญาตรี (วิศวกรรมศาสตรบัณฑิต)
สาขาวิชา	วิศวกรรมไฟฟ้า
คณะ	วิศวกรรมศาสตร์
ภาคการศึกษา/ปีการศึกษา	1/2564

บทคัดย่อ

โครงงานสหกิจศึกษานี้นำเสนอ ระบบควบคุมแสงสว่างของโรงงานโดยใช้อินเตอร์เน็ตของสรรพสิ่ง ซึ่งเป็นสิ่งที่ภายในโรงงานควรติดตั้งระบบควบคุมแสงสว่างไว้นั้นเพื่อลดค่าใช้จ่ายภายในองค์กรได้อย่างชัดเจน การศึกษานี้ดำเนินการในระหว่างการฝึกปฏิบัติงานสหกิจศึกษากับบริษัท เอส คิว ดี มาร์เก็ตติ้ง จำกัด กระบวนการประกอบด้วย การศึกษาหลักการทำงานและการทดสอบส่วนประกอบต่างๆ ของระบบควบคุมแสงสว่างการรวบรวมและวิเคราะห์ปัญหาในการทำงานของระบบควบคุมแสงสว่างการแก้ไขหรือเปลี่ยนส่วนประกอบของระบบควบคุมแสงสว่างที่เสียหายและการทดสอบสมรรถาพในการทำงานของระบบควบคุมแสงสว่างที่ได้ทำการปรับปรุงนี้มีคุณลักษณะในการทำงานที่สมบูรณ์และปลอดภัยการออกแบบและการติดตั้งระบบควบคุมแสงสว่างของโรงงานโดยใช้อินเตอร์เน็ตของสรรพสิ่ง ได้ถูกนำเสนอไว้อย่างละเอียดในรายงานฉบับนี้

คำสำคัญ: ระบบควบคุมแสงสว่าง/การปรับปรุง/ส่วนประกอบของระบบควบคุม

Project Title:	Design and Installation of a Lighting Control System in a Factory Using Internet of Things (IoT)
Credits:	5 Units
By:	Mr. Charan Rukyate 6223200025
Advisor:	Asst. Prof. Dr. Yongyuth Naras
Degree:	Bachelor of Engineering
Major:	Electrical Engineering
Faculty:	Engineering
Semester/Academic Year:	1/2021

Abstract

This cooperative education report presents work experiences on the design and installation of a lighting control system in a factory using Internet of Things (IoT). Work experience was gained through study and practice at SQD Marketing Company Limited, performed during the cooperative education program of Siam University. The main objective of installing this system was to reduce electrical energy consumption of the lighting system. Operating principles and testing of components for the installation included Node MCU, power supply, relay module, monitoring devices, and LED lamps were first presented in detail. Then, the installation procedures as well as maintenance plans were presented clearly in this report.

Keywords: lighting control system, Internet of Things, Node MCU

Approved by

.....

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูปภาพ

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญ

SQD MARKETING เป็นตัวแทนจำหน่ายอุปกรณ์ไฟฟ้าภายในบ้านชั้นนำของประเทศไทย ผ่านมาตรฐานสากลที่คุณต้องการแบบครบถ้วนนอกจากนี้เรายังพัฒนานวัตกรรมสมัยใหม่อย่างต่อเนื่องโดยมีบริการซอฟต์แวร์ที่สามารถช่วยให้คุณลดการใช้พลังงานและประหยัดค่าไฟที่สายการผลิตได้อีกด้วยเนื่องจากปัจจุบันวิถีการใช้ชีวิตประจำวันของคนไทยอยู่ในสภาวะเร่งรีบตลอดทั้งที่ต้องออกไปทำงานหรือทำภารกิจนอกบ้านซึ่งบางครั้งพบว่าอาจจะลืมปิดไฟแสงสว่างปิดอุปกรณ์ไฟฟ้าต่างๆ หรือเกิดความไม่แน่ใจว่าได้ปิดไฟก่อนอออกจากบ้านหรือไม่ ซึ่งเป็นต้นเหตุให้เกิดความสิ้นเปลืองการใช้พลังงานไฟฟ้ารวมไปถึงอาจจะก่อให้เกิดภัยอัคคีภัยได้จากแนวคิดของ Internet of Things (IoT) เป็นแนวคิดที่จะให้อุปกรณ์อิเล็กทรอนิกส์สามารถสื่อสารด้วยกัน ระบบที่จะเปลี่ยนแปลงวิถีชีวิตของมนุษย์ โดยเป็นระบบที่ให้วัตถุหรือสิ่งของสามารถสื่อสารเชื่อมต่อกับอุปกรณ์หรือเครื่องจักรในโรงงานของ Jo et al. (2013) ได้นำเสนอเกี่ยวกับการจัดตารางเวลาตามอัตราค่าไฟฟ้าในแต่ละช่วงเวลาเพื่อช่วยประหยัดพลังงานของอุปกรณ์ทำความร้อนและอุปกรณ์ทำความเย็นภายในบ้านซึ่งจากเหตุผลข้างต้นโครงงานนี้จึงได้นำเสนอต้นแบบการเปิด-ปิดไฟโดยใช้ระบบแอพพลิเคชั่น Blynk บนอุปกรณ์ที่รองรับระบบปฏิบัติการ แอนดรอยด์ และ ไอโอเอส ผ่านอุปกรณ์ที่มีชื่อว่า ESP8266/ NodeMCU โดยเชื่อมเข้ากับชุดควบคุมสวิตช์เปิด-ปิดไฟ และเชื่อมต่อเข้ากับระบบเครือข่ายไร้สายภายในบ้านที่ความถี่ 2.4 GHz ตามมาตรฐาน IEEE 802.11 ซึ่งมีค่าใช้จ่ายในการติดตั้ง และบำรุงรักษาที่ต่ำและสะดวกต่อการใช้งานเพียงแค่สัมผัส

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 สร้างทักษะการโปรแกรมต่อการนำไปแก้ปัญหาอย่างมีประสิทธิภาพ

1.2.2 เพื่อสร้างสภาพแวดล้อมการทำงานที่มีความปลอดภัยและมีประสิทธิภาพมากขึ้น

1.3 ขอบเขตของโครงงาน

1.3.1 สามารถตรวจสอบระบบการทำงานได้ในระยะไกล

1.3.2 กำหนดเวลาการใช้งานได้อย่างชัดเจน

1.4 ประโยชน์ที่คาดว่าจะได้รับจากโครงงาน

1.4.1 สามารถตรวจสอบสถานะและการทำงานของระบบแสงสว่างได้แบบเรียลไทม์

1.4.2 อำนวยความสะดวก และยกระดับกิจการให้ Smart Factory

1.4.3 ลดการใช้พลังงานไฟฟ้าสำหรับแสงสว่างในโรงงานลงอย่างน้อย 30% ภายใน 6 เดือนหลังจากติดตั้งระบบ

บทที่ 2
เอกสารและโครงงานที่เกี่ยวข้อง

ความก้าวหน้าทางด้านเทคโนโลยีทำให้มีการพัฒนาคิดค้นสิ่งอำนวยความสะดวกต่อการดำรงชีวิตเป็นอันมาก เทคโนโลยีได้เข้ามาเสริมปัจจัยพื้นฐานการดำรงชีวิตได้เป็นอย่างดี เทคโนโลยีทำให้การสร้างที่พักอาศัยมีคุณภาพมาตรฐานสามารถผลิตสินค้าและให้บริการต่างๆ เพื่อตอบสนองความต้องการของมนุษย์มากขึ้น เทคโนโลยีทำให้ระบบการผลิตสามารถผลิตสินค้าได้เป็นจำนวนมาก มีราคาถูกลง สินค้าได้คุณภาพ เทคโนโลยีทำให้มีการติดต่อสื่อสารกันได้สะดวก

การเดินทางเชื่อมโยงถึงกันทำให้ประชากรในโลกติดต่อรับฟังข่าวสารกันได้ตลอดเวลาด้วยเหตุเหล่านี้โครงงานจึงเห็นความสำคัญที่จะนำเทคโนโลยีต่างๆมาประยุกต์ใช้ให้เกิดประโยชน์ในการสร้างเครื่องควบคุมอุปกรณ์ไฟฟ้าด้วยระบบ wi-fi ซึ่งได้ทำการศึกษาเอกสารและโครงงานที่เกี่ยวข้องโดยมีรายละเอียดหัวข้อดังต่อไปนี้

2.1 Module wi-fi Node MCU V2 ESP8266

ความเป็นมาของ Node MCU V2 ESP8266 การใช้งาน ESP8266 ในช่วงแรกนั้นจะเป็นการนำเอา ESP ไปใช้ในลักษณะของโมดูล Serial to Wi-Fi ทำให้ Microcontroller ของเราสามารถเชื่อมต่อกับระบบ Wi-Fi ได้อย่างง่ายดายผ่าน AT.Command ต่อมาเริ่มมีการนำ ESP มาใช้งานในรูปแบบ Standalone มากขึ้นโดยเขียน Firmware ลงไปบน ESP8266 โดยตรงเพื่อให้ ESP สามารถรับค่าจาก Sensor ต่างๆประมวลผลและรับส่งข้อมูลได้ด้วยตัวมันเพียงลำพัง

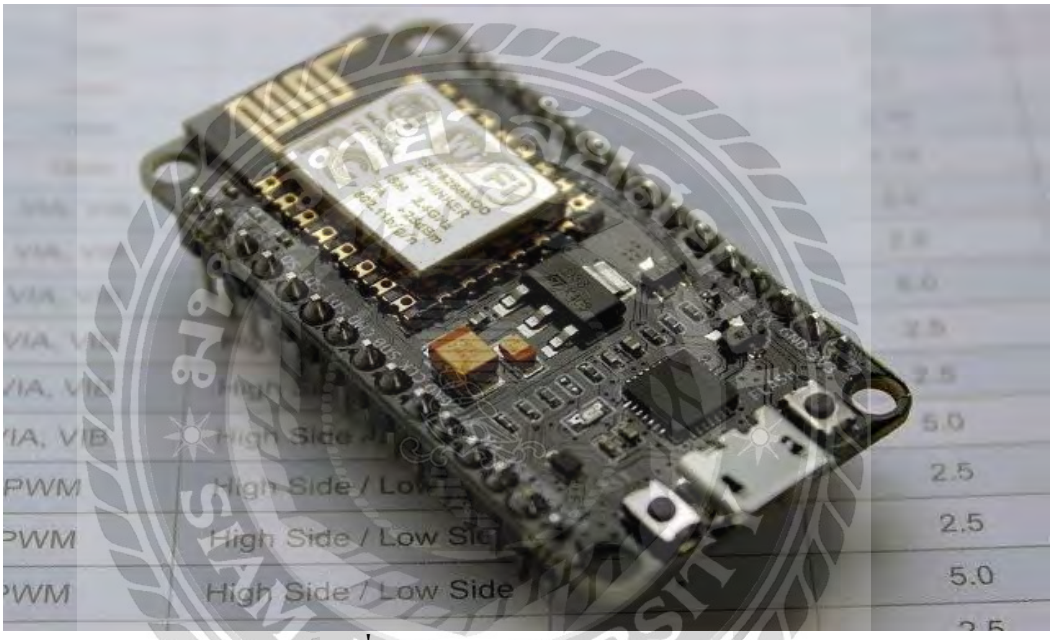
ซึ่งสามารถพัฒนาได้โดยใช้ SDK จากผู้ผลิตแต่การใช้งาน SDK ก็ไม่สะดวกกับการพัฒนามากนัก ต่อมาจึงมีการพัฒนามาใช้ Node MCU โดยใช้ภาษา Lua ในการพัฒนาซึ่งก็สามารถใช้งานได้ดีในระดับหนึ่งและในที่สุดวันนี้ ESP8266 ก็สามารถพัฒนาด้วย Platform Arduino ได้แล้วและใช้ Arduino IDE เป็น Tool ในการพัฒนาและโปรแกรมทำให้ขั้นตอนในกาติดตั้ง Environment ต่างๆง่ายขึ้นและยังใช้รูปแบบการเขียนโปรแกรมเช่นเดียวกับ Arduino ที่คุ้นเคยกันเป็นอย่างดี

2.1.1 Node MCU V2 ESP8266 / Node MCU.Development.KitV2

เป็นตัวที่พัฒนาจาก Node MCU Version เดิมโดยเป็นโมดูลที่ประกอบด้วย ESP8266-12E มีเสาอากาศแบบ PCB Antenna เชื่อมต่อเฮดเดอร์สำหรับขาสัญญาณต่างๆ ได้แก่ GPIO, PWM, I2C, 1-wire, ADC และมี SPI เพิ่มขึ้นมาจาก Version เดิมมีส่วนของ USB-to-TTL และพอร์ต Micro USB ซึ่งใช้ชิพ USB to Serial ของ silicon lab cp2102 เชื่อมต่อเข้ากับเครื่องคอมพิวเตอร์เพื่อพัฒนาโปรแกรม

สามารถติดตั้งเฟิร์มแวร์ Node MCU ได้และยังมีขนาดของ PCB ที่เล็กลงสามารถใช้งานกับ breadboard ได้ผู้ใช้สามารถเลือกพัฒนาด้วยสคริปต์ Lau โดยใช้เฟิร์มแวร์ Node MCU หรือใช้เป็นชุดพัฒนาด้วยโมดูล ESP8266 ก็ได้ซึ่งสามารถเขียนด้วย Arduino IDE

ได้โมดูลมี GPIO ให้ใช้ถึง 10 พอร์ต สามารถนำมาพัฒนาโปรเจ็คทางด้าน Internet of Things (IoT) เชื่อมต่อกับอุปกรณ์อื่นๆตามต้องการโมดูล ESP8266 มีหลายรุ่นให้เลือกใช้งานสำหรับผู้ที่สนใจทดลองใช้งานชิป ESP8266SoC บอร์ด Node MCU.v2(Node MCU-DevKit1.0) เป็นอีกตัวเลือกหนึ่งและมีจุดเด่นหลายข้อเมื่อเปรียบเทียบกับบอร์ดอื่นๆและซึ่งมีหลายจุดที่แตกต่างจาก Node MCU v1(Node MCU-Dev Kit) ซึ่งเป็นเวอร์ชั่น 2 แสดงดังรูปที่ 2.1

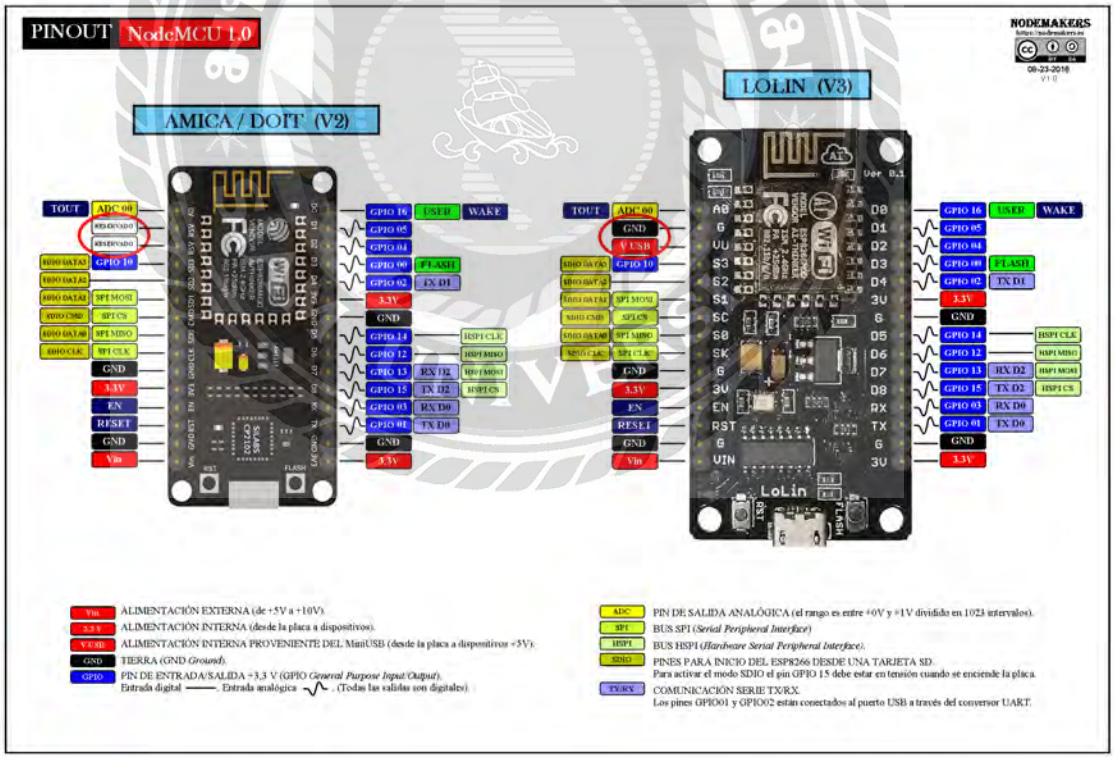


รูปที่ 2.1 Node MCU v2ESP8266-12

ข้อมูลสำคัญเชิงเทคนิคของบอร์ด Node MCU.v2

1. ใช้โมดูล ESP-12E (ESP8266SoCchip) ของบริษัท AI Thinker (ในขณะที่ Node MCU.v1 ใช้โมดูล ESP12) มีขาเพิ่มมาอีก 6 ขาเมื่อเปรียบเทียบกับ ESP-12
2. ใช้ชิป Flash ความจุ 32 Mbits (4MBytes)
3. มีขนาดแคบกว่า NodeMCU.v1 ดังนั้นเมื่อเสียบขาลงบนเบรดบอร์ดจะมีช่องเหลือด้านข้างทำให้สะดวกในการต่อวงจรบนเบรดบอร์ด
4. มีวงจรควบคุมแรงดัน 3.3V (@800mAmax) บนบอร์ดใช้ไอซีที่จ่ายกระแสได้มากกว่าบอร์ด NodeMCU.v1
5. ใช้ชิป CP2102 ของ Silabs ทำหน้าที่เป็นส่วนเชื่อมต่อ USB-to-Serial (แต่ NodeMCU.v1ใช้ชิป CH340G)

6. มีขาสำหรับ SPI สำหรับต่อกับการ์ด SD (เพิ่มจากเดิมที่มีขาสำหรับ HSPI)
7. มีขา GPIO3/RXD0 และ GPIO1/TXD0 ที่ต่อกับขา TXD และ RXD ของชิป CP2102 ตามลำดับ
8. มีขาGPIO13/RXD2 และ GPIO15/TXD2 (ใช้เป็นพอร์ต Serial เพิ่มอีกหนึ่งชุด)
9. ใช้คอนเนกเตอร์แบบ micro-USB สำหรับจ่ายแรงดันไฟเลี้ยง (VUSB) เท่ากับ+5V และสำหรับดาวน์โหลดเฟิร์มแวร์(แรงดัน VUSB ต่อผ่าน Schottky Diode1N5819 ไปยัง VDD-5V)
10. สามารถจ่ายแรงดันไฟเลี้ยง +5V จากภายนอกได้(ต่อเข้าที่ขา VDD5V)
11. มีปุ่มกด RST (รีเซตการทำงาน)และ Flash (สำหรับโปรแกรมเฟิร์มแวร์ใหม่)
12. มีขา A0 รับอินพุตแรงดันแบบแอนะล็อกสำหรับวงจร ADC (ขนาด10บิต)ที่อยู่ภายในชิป ผ่านวงจรแบ่งแรงดันด้วยตัวต้านทาน 100k/220k (ลดแรงดันอินพุตจา 0.3V-3V ลงมาให้อยู่ในช่วง 0V-1V)
13. ลักษณะการทำงานของ Node MCU.v2/v3



รูปที่ 2.2 ลักษณะการทำงาน Node MCU v2/v3ESP8266

2.2 Arduino

1. โครงสร้างโปรแกรมของ Arduin

ในการเขียนโปรแกรมสำหรับบอร์ด Unicon จะต้องเขียนโปรแกรมโดยใช้ภาษาของ Arduino (Arduino Programming Language) ซึ่งตัวภาษาของ Arduino เองก็นำเอาโอเพ่นซอร์สโปรเจคชื่อ Wiring มาพัฒนาต่อภาษาของ Arduino แบ่งได้เป็น 2 ส่วนหลักคือ

1) โครงสร้างภาษา (Structure) ตัวแปรและค่าคงที่

2) ฟังก์ชัน (Function)

Structure ภาษาของ Arduino จะอ้างอิงตามภาษา C/C++ จึงอาจกล่าวได้ ว่าการเขียนโปรแกรมสำหรับ Arduino (ซึ่งก็รวมถึงบอร์ด Unicon) ก็ คือการเขียนโปรแกรมภาษา C โดยเรียกใช้ฟังก์ชันและไลบรารีที่ทาง Arduino ได้เตรียมไว้ให้แล้วซึ่งสะดวกและทำให้ผู้ที่ไม่มีความรู้ด้านไมโครคอนโทรลเลอร์อย่างลึกซึ้งสามารถเขียนโปรแกรมสั่งงานได้ ในบทนี้จะอธิบายถึงโครงสร้างโปรแกรมของ Arduino

Arduino โปรแกรมของ Arduino แบ่งได้เป็นสองส่วนคือ void setup ( ) และ void loop ( ) โดยฟังก์ชัน setup ( ) เมื่อโปรแกรมทำงานจะทำคำสั่งของฟังก์ชันนี้เพียงครั้งเดียวใช้ในการกำหนดค่าเริ่มต้นของการทำงานส่วนฟังก์ชัน loop ( ) เป็นส่วนทำงานโปรแกรมจะทำคำสั่งในฟังก์ชันนี้ต่อเนื่องกันตลอดเวลาโดยปกติใช้กำหนดโหมดการทำงานของขาต่างๆกำหนดการสื่อสารแบบอนุกรม ฯลฯส่วนของ loop ( ) เป็นโค้ดโปรแกรมที่ทำงานเช่นอ่านค่าอินพุตประมวลผลสั่งงานเอาต์พุต ฯลฯ โดยส่วนกำหนดค่าเริ่มต้นเช่นตัวแปรจะต้องเขียนที่ส่วนหัวของโปรแกรมก่อนถึงตัวฟังก์ชันนอกจากนั้นยังต้องคำนึงถึงตัวพิมพ์เล็ก-ใหญ่ของตัวแปรและชื่อฟังก์ชันให้ถูกต้อง

ส่วนของฟังก์ชัน setup ( ) ฟังก์ชันนี้จะเขียนที่ส่วนต้นของโปรแกรมทำงานเมื่อโปรแกรมเริ่มต้นเพียงครั้งเดียว ใช้เพื่อกำหนดค่าของตัวแปร โหมดการทำงานของขาต่างๆ เริ่มต้นเรียกใช้ไลบรารี ฯลฯ

ตัวอย่างที่ 1

```
int buttonPin = 31;
void  setup  ( )
{
Serial.begin(9600);
pinMode(buttonPin,  INPUT);
}
Void  loop   ()
```

```
{
if (digitalRead(buttonPin) == HIGH)
Serial.println('H');
else
Serial.println('L');
delay(1000);
}
```

ในขณะที่โปรแกรมภาษา C มาตรฐานที่เขียนบน AVR GCC (เป็นโปรแกรมภาษา C ที่ ใช้ C คอมไพเลอร์แบบ GCC สำหรับไมโครคอนโทรลเลอร์ AVR) จะเขียนได้ดังนี้

```
int main(void)
{
Init ( );
Setup ( );        ตรงกับ void
Setup ( )
for (;;)
loop ( );                ตรงกับ void loop
return ;
```

ส่วนของฟังก์ชัน loop ( ) หลังจากที่เขียนฟังก์ชัน setup ( ) ที่ กำหนดค่าเริ่มต้นของโปรแกรมแล้วส่วนถัดมาคือ ฟังก์ชัน loop ( ) ซึ่งมีการทำงานตรงตามชื่อคือจะทำงานตามฟังก์ชั่นนี้วนต่อเนื่องตลอดเวลา ภายในฟังก์ชันนี้จะมีโปรแกรมของผู้ใช้เพื่อรับค่าจากพอร์ตประมวลแล้วสั่งเอาต์พุตออกขาต่างๆ เพื่อควบคุมการทำ งานของบอร์ด

ตัวอย่างที่ 2

```
int buttonPin = 31;
// setup initializes serial and the button pin
void setup()
{
Serial.begin(9600);
pinMode(buttonPin, INPUT);
}
```

```
Serial.println('L');
delay(1000);
// loop checks the button pin each time,
// and will send serial if it is pressed
void loop()
{
if (digitalRead(buttonPin) == HIGH)
Serial.println('H');
else
```

2.2.1 คำสั่งของโปรแกรม Arduino

คำสั่ง if ใช้ทดสอบเพื่อกำหนดเงื่อนไขการทำ งานของโปรแกรมเช่นถ้าอินพุตมีค่ามากกว่าค่าที่กำหนดไว้ จะให้ทำอะไรโดยมีรูปแบบการเขียนดังนี้

```
if (someVariable >50)
{
//do something here
}
```

ตัวโปรแกรมจะทดสอบว่าถ้าตัวแปร someVariable มีค่ามากกว่า 50 หรือไม่ ถ้าใช่ให้ทำอะไรถ้าไม่ใช่ให้ข้ามการทำงานส่วนนี้การทำงานของคำสั่งนี้จะทดสอบเงื่อนไขที่เขียนในเครื่องหมายวงเล็บถ้าเงื่อนไขเป็นจริงทำตามคำสั่งที่เขียนในวงเล็บปีกกา ถ้าเงื่อนไขเป็นเท็จข้ามการทำงานส่วนนี้ไปส่วนของการทดสอบเงื่อนไขที่ เขียนอยู่ภายในวงเล็บจะต้องใช้ตัวกระทำ เปรียบเทียบต่างๆ ดังนี้

```
x == y (x เท่ากับ y)
x != y (x ไม่เท่ากับ y)
x < y (x น้อยกว่า y)
x > y (x มากกว่า y)
x <= y (x น้อยกว่าหรือเท่ากับ y)
x >= y (x มากกว่าหรือเท่ากับ y)
```

เทคนิคสำหรับการเขียนโปรแกรมในการเปรียบเทียบตัวแปรให้ใช้ตัวกระทำ == (เช่น if (x=10)) ห้ามเขียนผิดเป็น = (เช่น if (x=10) คำสั่งที่เขียนผิดในแบบที่สองนี้ทำให้ผลการทดสอบเป็นจริงเสมอ

และเมื่อผ่านคำสั่งนี้แล้ว x มีค่าเท่ากับ 10 ทำให้การทำงานของโปรแกรมผิดเพี้ยนไปไม่เป็นตามที่กำหนดไว้เราสามารถใช้ คำ สั่ง if ในคำ สั่งควบคุมการแยกเส้นทางของโปรแกรมโดยใช้คำสั่ง if else

1) คำสั่ง if.else ใช้ทดสอบเพื่อกำหนดเงื่อนไขการทำงานของโปรแกรมได้มากกว่าคำสั่ง if ธรรมดา โดยสามารถกำหนดได้ว่าถ้าเงื่อนไขเป็นจริงให้ทำอะไร ถ้าเป็นเท็จให้ทำอะไร เช่น ถ้าค่าอินพุตอะนาลอกที่อ่านได้น้อยกว่า 500 ให้ทำอะไร ถ้าค่ามากกว่าหรือเท่ากับ 500 ให้ทำอีกอย่างจะเขียนคำสั่งได้ดังนี้

ตัวอย่างที่ 3

```
if (pinFiveInput < 500)
{
// do Thing A
}
else
{
// do Thing B
}
```

หลังคำสั่ง else สามารถตามด้วยคำสั่ง if สำหรับการทดสอบอื่นๆทำให้รูปแบบคำสั่งกลายเป็น if....else...if เป็นการทดสอบเงื่อนไขต่างๆ เมื่อเป็นจริงให้ทำตามที่ต้องการดังตัวอย่างต่อไปนี้

ตัวอย่างที่4

```
if (pinFiveInpsut < 500)
{
// do Thing A
}
else if (pinFiveInput >= 1000)
{
// do Thing B
}
else
{
// do Thing C
}
```

หลังคำสั่ง else สามารถตามด้วยคำสั่ง if ได้ไม่จำกัดจำนวน (สามารถใช้คำสั่ง switch case แทนคำสั่ง if.else.if สำหรับการทดสอบเงื่อนไขจำนวนมากๆได้)เมื่อใช้คำสั่ง if.else แล้วต้องกำหนดด้วยว่าถ้าทดสอบไม่ตรงกับเงื่อนไขใดๆ เลย ให้ทำอะไรโดยให้กำหนดที่คำสั่ง else ตัวสุดท้าย

2) คำสั่ง for ( )

คำสั่งนี้ใช้เพื่อสั่งให้คำสั่งที่อยู่ภายในวงเล็บปีกกาหลัง for มีการทำงานซ้ำกันตามจำนวนรอบที่ต้องการคำสั่งนี้มีประโยชน์มากสำหรับการทำงานใดๆที่ต้องทำซ้ำกันและทราบจำนวนรอบของการทำซ้ำที่แน่นอนมักใช้คู่กับตัวแปรอะเรย์ในการเก็บสะสมค่าที่อ่านได้จากอินพุตแอนะล็อก https://www.kroobannok.com/1818 หลายๆ ขาที่มีหมายเลขขาต่อเนื่องกันรูปแบบของคำ สั่ง for( ) แบ่งได้ 3 ส่วนดังนี้

```
for (initialization; condition; increment)
{
//
statement(s);
}
```

เริ่มต้นด้วย initialization ใช้กำหนดค่าเริ่มต้นของตัวแปรควบคุมการวนรอบในการทำงานแต่ละรอบจะทดสอบ condition ถ้าเงื่อนไขเป็นจริงทำตามคำสั่งในวงเล็บปีกกา แล้วมาเพิ่มหรือลดค่าตัวแปรตามที่สั่งในincrement แล้วทดสอบเงื่อนไขอีกทำซ้ำจนกว่าเงื่อนไขเป็นเท็จ

ตัวอย่างที่ 5

```
for (int i=1; i <= 8; i++)
{
//
 statement using the value i;
}
```

คำสั่ง for ของภาษา C ยืดหยุ่นกว่าคำสั่ง for ของภาษาคอมพิวเตอร์อื่นๆ มันสามารถละเว้นบางส่วนหรือทั้งสามส่วนของคำสั่ง for ได้อย่างไรก็ตามยังต้องมี เซมิโคลอนนอกจากนั้นยังนำคำสั่งภาษา C++ ที่ มีตัวแปรที่ไม่เกี่ยวข้องมาเขียนในส่วนของ initialization, condition และ increment ของคำสั่ง for ได้

3) คำสั่ง switch-case

ใช้ทดสอบเงื่อนไขเพื่อกำหนดการทำงานของโปรแกรมถ้าตัวแปรที่ทดสอบตรงกับเงื่อนไขใดก็ให้ทำงานตามที่กำหนดไว้

พารามิเตอร์ var ตัวแปรที่ต้องการทดสอบว่าตรงกับเงื่อนไขใด default ถ้าไม่ตรงกับเงื่อนไขใดๆ เลยให้ทำคำสั่งต่อท้ายนี้ break เป็นส่วนสำคัญมากใช้ เขียนต่อท้าย case ต่างๆ เมื่อพบเงื่อนไขนั้นแล้วทำตามคำสั่งต่างแล้วให้หยุดการทำ งานของคำ สั่ง switch-case ถ้าลืมเขียน break เมื่อพบเงื่อนไขทำตามเงื่อนไขแล้วโปรแกรมจะทำ งานตามเงื่อนไขต่อไปเรื่อยๆ จนกว่าจะพบคำสั่ง break

ตัวอย่างที่ 6

```
switch (var)
{
case 1:
//do something when var == 1
Break ;
switch (var)
{
case 1:
//do something when var == 1
Break ;
```

4) คำสั่ง while

เป็นคำสั่งวนรอบโดยจะทำคำสั่งที่เขียนในวงเล็บปีกกาอย่างต่อเนื่องจนกว่าเงื่อนที่เขียนในวงเล็บของคำสั่ง while ( ) จะเป็นเท็จคำสั่งที่ให้ทำซ้ำจะต้องมีการเปลี่ยนแปลงค่าตัวแปรที่ใช้ทดสอบ เช่น มีการเพิ่มค่าตัวแปรหรือมีเงื่อนไขภายนอก เช่น อ่านค่าจากตัวตรวจจับได้เรียบร้อยแล้วให้ หยุดการอ่านค่า มิฉะนั้นเงื่อนไขในวงเล็บของ while ( ) เป็นจริงตลอดเวลาทำให้คำสั่ง while ทำงานวนรอบไปเรื่อยๆไม่รู้จบ รูปแบบคำสั่ง

```
while(expression)
{
// statement(s)
}
```

พารามิเตอร์ expression เป็นคำสั่งทดสอบเงื่อนไข (ถูกหรือผิด)

ตัวอย่างที่ 7

```
var =0;
while(var < 200)
{
//do something repetitive 200 times>>var++;
}
```

2.2.2 ตัวกระทำทางคณิตศาสตร์

ประกอบด้วยตัวกระทำ 5 ตัวคือ + (บวก), - (ลบ), * (คูณ), / (หาร) และ % (หารเอาเศษ)

ตัวกระทำทางคณิตศาสตร์ บวก ลบ คูณ และหารใช้หาค่าผลรวมผลต่างผลคูณและผลหารค่าของตัวถูกกระทำสองตัวโดยให้คำตอบมีประเภทตรงกับตัวถูกกระทำ ทั้งสองตัว เช่น 9/4 ให้ คำตอบเท่ากับ 2 เนื่องจากทั้ง 9 และ 4 เป็นตัวแปรเลขจำนวนเต็ม (int) นอกจากนี้ตัวกระทำทางคณิตศาสตร์อาจทำให้เกิดโอเวอร์โฟลว (overflow) ถ้าผลลัพธ์ที่ได้ มีขนาดใหญ่เกินกว่าจะสามารถเก็บในตัวแปรประเภทนั้นถ้าตัวที่ถูกกระทำ ต่างประเภทกันผลลัพธ์ได้เป็นจะมีขนาดใหญ่ขึ้น

รูปแบบคำสั่ง

```
result = value1 + value2;
result = value1 - value2;
result = value1 * value2;
result = value1 / value2;
```

พารามิเตอร์

value1 : เป็นค่าของตัวแปรหรือค่าคงที่ ใดๆ

value2: เป็นค่าของตัวแปรหรือค่าคงที่ ใดๆ

ตัวอย่างที่ 8

```
y = y + 3;
x = x - 7;
i = j * 6;
r = r / 5;
```

**เทคนิคสำหรับการเขียนโปรแกรม

เลือกขนาดของตัวแปรให้ใหญ่ พอสำหรับเก็บค่าผลลัพธ์ที่มากที่สุดของการคำนวณและต้องทราบว่าที่ค่าใดตัวแปรที่เก็บจะมี การวนซ้ำค่ากลับและวนกลับอย่างไรตัวอย่างเช่น (0 ไป 1)หรือ (0 ไป -32768)

สำหรับการคำนวณที่ต้องการเศษส่วนให้ใช้ ตัวแปรประเภท float แต่ให้ระวังผลลบเช่น ตัวแปรมีขนาดใหญ่ คำนวณได้ช้า

ใช้ตัวกระทำ cast เช่น (int) myfloat ในการเปลี่ยนประเภทของตัวแปรชั่วคราวขณะที่โปรแกรมทำงาน

1) ตัวกระทำ % หารเอาเศษใช้หาค่าเศษที่ได้ของการหารเลขจำนวนเต็ม 2 ตัว ตัวกระทำหารเอาเศษไม่สามารถใช้งานกับตัวแปรเลขทศนิยม (float)

result = value1 % value2; พารามิเตอร์

value1 – เป็นตัวแปรประเภท byte,char,int หรือ long

value2 – เป็นตัวแปรประเภท byte,char,int หรือ long

ผลที่ได้ เศษจากการหารค่าเลขจำนวนเต็มเป็นข้อมูลชนิดเลขจำนวนเต็ม

ตัวอย่างที่ 9

```
x = 7 % 5; // x now contains 2
x = 9 % 5; // x now contains 4
x = 5 % 5; // x now contains 0
x = 4 % 5; // x now contains 4
```

ตัวกระทำหารเอาเศษนี้ใช้ประโยชน์มากในงานที่ต้องการให้เหตุการณ์เกิดขึ้นด้วยช่วงเวลาที่สม่ำเสมอหรือใช้ทำให้หน่วยความจำที่เก็บตัวแปรอะเรย์เกิดการล้นค่ากลับ (roll over)

ตัวอย่างที่ 10

```
// check a sensor every 10 times through a loop
void setup()
{
i++;
if ((i % 10) == 0)
{
x = analogRead(sensPin); // read sensor every ten times through loop
}       }
// setup a buffer that averages the last five samples of a sensor
```

```
int senVal[5];                    // create an array for sensor data
int i, j;                         // counter variables
long average;                     // variable to store average
...
void loop()
{                                 // input sensor data into oldest memory slot
sensVal[(i++) % 5] = analogRead(sensPin);
average = 0;
for (j=0; j<5; j++)
{
average += sensVal[j];            // add samples
}
average = average / 5;            // divide by tota
```

2) ตัวกระทำเปรียบเทียบ

ใช้ประกอบกับคำสั่ง if ( ) และ while ( ) เพื่อทดสอบเงื่อนไขหรือเปรียบเทียบค่าตัวแปรต่างโดยจะเขียนเป็นนิพจน์อยู่ภายในเครื่องหมาย ( )

x == y (x เท่ากับ y)
x != y (x ไม่เท่ากับ y)
x < y (x น้อยกว่า y)
x > y (x มากกว่า y)
x <= y (x น้อยกว่าหรือเท่ากับ y)
x >= y (x มากกว่าหรือเท่ากับ y)

3) ตัวกระทำทางตรรกะ

ใช้ในการเปรียบเทียบของคำสั่ง if ( ) มี 3 ตัวคือ &&, || และ !

&& (ตรรกะ และ) ให้ค่าเป็นจริงเมื่อผลการเปรียบเทียบทั้งสองข้างเป็นจริงทั้งคู่

ตัวอย่างที่ 11

```
if (x > 0 && x < 5)
{
//
```

```
...
}
```

ให้ค่าเป็นจริงเมื่อ x มากกว่า ( ) และน้อยกว่า 5 (มี ค่า 1 ถึง 4)

&& (ตรรกะ หรือ) ให้ค่าเป็นจริง เมื่อผลการเปรียบเทียบพบว่ามีตัวแปรใดเป็นจริงหรือเป็นจริงทั้งคู่

ตัวอย่างที่ 12

```
if (x > 0 || y > 0)
{
//
 ...
}
```

ให้ผลเป็นจริงเมื่อ x หรื อ y มี ค่ามากกว่า ()

(ใช้กลับผลเป็นตรงกันข้าม)ให้ค่าเป็นจริงเมื่อผลการเปรียบเทียบเป็นเท็จ

ตัวอย่างที่ 13

```
if (!x)
{
//          ...
}
```

ให้ผลเป็นจริงถ้า x เป็นเท็จ (เช่น ถ้า x = 0 ให้ผลเป็นจริง)

ข้อควรระวัง

ระวังเรื่องการเขียนโปรแกรมถ้าต้องการใช้ตัวกระทำตรรกะและต้องเขียนเครื่องหมาย && ถ้าลืมเขียนเป็น & จะเป็นตัวกระทำ และระดับบิตกับตัวแปรซึ่งให้ผลที่แตกต่างเช่นกันในการใช้ ตรรกะหรือให้ เขียนเป็น || (ขีดตั้งสองตัวติดกัน) ถ้าเขียนเป็น | (ขีดตั้งตัวเดียว) จะหมายถึงตัวกระทำหรือระดับบิตกับตัวแปรตัวกระทำ NOT ระดับบิต (~) จะแตกต่างจากตัวกลับผลให้เป็นตรงข้าม (!)ให้ เลือกใช้ ให้ถูกต้อง

ตัวอย่างที่ 14

```
if (a >= 10 && a <= 20)
{
// true if a is between 10 and 20
}
```

4) ตัวกระทำระดับบิต

ตัวกระทำระดับจะนำบิตของตัวแปรมาประมวลผลใช้ประโยชน์ในการแก้ปัญหาด้านการเขียนโปรแกรมได้หลากหลายตัวกระทำระดับของภาษา C (ซึ่งรวมถึง Arduino) มี 6 ตัวได้ แก่ & (bitwise AND),| (OR), ^ (Exclusive OR), ~ (NOT), << (เลื่อนบิตไปทางขวา) และ >> (เลื่อนบิตไปทางซ้าย)

5) ตัวกระทำระดับบิต AND (&)

คำสั่ง AND ในระดับบิตของภาษา C เขียนได้โดยใช้ & หนึ่งตัวโดยต้องเขียนระหว่างนิพจน์หรือตัวแปรที่เป็นเลขจำนวนเต็มการทำ งานจะนำข้อมูลแต่ละบิตของตัวแปรทั้งสองตัวมากระทำ ทางตรรกะและ (AND) โดยมีกฎดังนี้ถ้าอินพุตทั้งสองตัวเป็น “1” ทั้งคู่เอาต์พุตเป็น “1” กรณี อื่นๆเอาต์พุตเป็น “0” ดังตัวอย่างต่อไปนี้ ในการดูให้คู่ของตัวกระทำ ตามแนวตั้ง

0 0 1 1 Operand1

0 1 0 1 Operand2

0 0 0 1 Returned result

ใน Arduino ตัวแปรประเภท int จะมีขนาด 16 บิต ดังนั้นเมื่อใช้ตัวกระทำระดับบิต AND จะมีการกระทำตรรกะและพร้อมกันกับข้อมูลทั้ง 16 บิต ดังตัวอย่างในส่วนของโปรแกรมต่อไปนี้

ตัวอย่างที่ 15

```
int a = 92; // in binary: 0000000001011100
int b = 101; // in binary: 0000000001100101
int c = a & b; // result: 0000000001000100
// or 68 in decimal.
```

ในตัวอย่างนี้จะนำข้อมูลทั้ง 16 บิตของตัวแปร a และ b มากระทำ ทางตรรกะ AND แล้วนำผลลัพธ์ที่ได้ทั้ง 16 บิตไปเก็บที่ตัวแปร c ซึ่งได้ค่าเป็น 01000100 ในเลขฐานสองหรือเท่ากับ 68 ฐานสิบ นิยมใช้ตัวกระทำระดับบิต AND เพื่อเลือกข้อมูลบิตที่ต้องการ(อาจเป็นหนึ่งบิตหรือหลายบิต)จากตัวแปร int ซึ่งการเลือกเพียงบางบิตนี้จะเรียกว่า masking

6) ตัวกระทำระดับบิต OR (|)

คำสั่งระดับบิต OR ของภาษาซีเขียนได้โดยใช้เครื่องหมาย | หนึ่งตัวโดยต้องเขียนระหว่างนิพจน์หรือตัวแปรเป็นเลขจำนวนเต็มการทำงานจะนำข้อมูลแต่ละบิตของตัวแปรทั้งสองตัวมากระทำทางตรรกะหรือ (OR)โดยมีกฎดังนี้ถ้าอินพุตตัวใดตัวหนึ่งหรือทั้งสองตัวเป็น“1” เอาต์พุตเป็น “1” กรณีที่อินพุตเป็น “0” ทั้งคู่เอาต์พุตจึงจะเป็น “0” ดังตัวอย่างต่อไปนี้

0 0 1 1 Operand1

0 1 0 1 Operand2

0 1 1 1 Returned result

ตัวอย่างที่ 16

ส่วนของโปรแกรมแสดงการใช้ ตัวกระทำ ระดับบิ ต OR

```
int a = 92; // in binary: 0000000001011100
int b = 101; // in binary: 0000000001100101
int c = a | b; // result: 0000000001111101
// or 125 in decimal.
```

โปรแกรมแสดงการใช้ตัวกระทำระดับบิต AND และ ORตัวอย่างงานที่ใช้ตัวกระทำระดับบิต AND และ OR เป็นงานที่โปรแกรมเมอร์เรียกว่า Read-Modify - Write on a port สำหรับไมโครคอนโทรลเลอร์ 8 บิต ค่าที่อ่านหรือเขียนไปยังพอร์ตมีขนาด 8 บิต ซึ่งแสดงค่าอินพุตที่ขาทั้ง 8 ขา การเขียนค่าไปยังพอร์ตจะเขียนค่าครั้งเดียวได้ทั้ง 8 บิต ซึ่งแสดงค่าอินพุตที่ขาทั้ง 8 ขา การเขียนค่าไปยังพอร์ตจะเขียนค่าครั้งเดียวได้ทั้ง 8 บิต

ตัวแปรชื่อ PORTD เป็นค่าที่ใช้แทนสถานะของขาดิจิตอลหมายเลข 0,1,2,3,4,5,6,7 ถ้าบิตใดมีค่าเป็นทำให้ขานั้นมีค่าลอจิกเป็น HIGH (อย่าลืมกำหนดให้ขาพอร์ตนั้นๆทำงานเป็นเอาต์พุตด้วยคำสั่ง pinMode() ก่อน) ดังนั้นถ้ากำหนดค่าให้ PORTD = B00110001;ก็คือต้องการให้ขา 2,3 และ 7 เป็น HIGH ในกรณีนี้ไม่ต้องเปลี่ยนค่าสถานะของขา 0 และ 1 ซึ่งปกติแล้วฮาร์ดแวร์ ของ Arduino ใช้ในการสื่อสารแบบอนุกรมถ้าไปเปลี่ยนค่าแล้วจะกระทบต่อการสื่อสารแบบอนุกรม

อัลกอริธึมสำหรับโปรแกรมเป็นดังนี้

- อ่านค่าจาก PORTD แล้วล้างค่าเฉพาะบิตที่ต้องการควบคุม (ใช้ตัวกระทำ แบบบิต AND)
- นำค่า PORTD ที่แก้ไขจากข้างต้นมารวมกับค่าบิตที่ต้องการควบคุม(ใช้ตัวกระทำแบบบิต OR)

ซึ่งเขียนเป็นโปรแกรมได้ดังนี้

ตัวอย่างที่ 17

```
int i;                                  // counter variable
int j;
void setup()
{
DDRD = DDRD | B11111100;
// set direction bits for pins 2 to 7,
// leave 0 and 1 untouched (xx | 00 == xx)
// same as pinMode(pin,OUTPUT) for pins 2 to 7
Serial.begin(9600);
}
```

```
void loop()
{
for (i=0; i<64; i++)
{
PORTD = PORTD & B00000011;
// clear out bits 2 - 7, leave pins 0
// and 1 untouched (xx & 11 == xx)
j = (i << 2);
// shift variable up to pins 2 - 7
// to avoid pins 0 and 1
PORTD = PORTD | j;
// combine the port information with
// the new information for LED pins
Serial.println(PORTD, BIN);
// debug to show masking
delay(100);
}
}
```

7) คำสั่งระดับบิต Exclusive OR (^)

เป็นโอเปอเรเตอร์พิเศษที่ไม่ค่อยได้ ใช้ ในภาษา C/C++ ตัวกระทำระดับบิต exclusive OR (หรือ XOR)จะเขียนโดยใช้สัญลักษณ์เครื่องหมาย ^ ตัวกระทำ นี้มีการทำ งานใกล้เคียงกับตัวกระทำ ระดับบิต OR แต่ต่างกันเมื่ออินพุตเป็น “1” ทั้งคู่จะให้เอาต์พุตเป็น “0” แสดงการทำงานได้ดังนี้

0 0 1 1 Operand1

0 1 0 1 Operand2

0 1 1 0 Returned result

หรือกล่าวได้อีกอย่างว่าตัวกระทำระดับบิ ต XOR จะให้เอาต์พุตเป็น “0” เมื่ออินพุตทั้งสองตัวมีค่าเหมือนกัน และให้เอาต์พุตเป็น “1” เมื่ออินพุตทั้งสองมีค่าต่างกัน

ตัวอย่างที่ 18

```
int x = 12; // binary: 1100
int y = 10; // binary: 1010
int z = x ^ y; // binary: 0110, or decimal 6
```

ตัวกระทำระดับบิต XOR จะใช้มากในการสลับค่าบางบิตของตัวแปร int เช่น กลับจาก “0” เป็น “1” หรือกลับจาก “1” เป็น “0”

เมื่อใช้ตัวกระทำ ระดับบิต XOR ถ้าบิตของ mask เป็น “1” ทำให้บิตนั้นถูกสลับค่า ถ้า mask มีค่าเป็น“1” บิตนั้นมีค่าคงเดิมตัวอย่างต่อไปนี้เป็นโปรแกรมแสดงการสั่งให้ขาดิจิตอล 5 (Di 5) มีการกลับลอจิกตลอดเวลา

ตัวอย่างที่ 19

```
// Blink_Pin_5
// demo for Exclusive OR
void setup()
{
DDRD = DDRD | B00100000;
// set digital pin five as OUTPUT
Serial.begin(9600);
}
void loop()
{
PORTD = PORTD ^ B00100000;
// invert bit 5 (digital pin 5),
// leave others untouched
delay(100);
}
```

7) ตัวกระทำระดับบิต NOT (~)

ตัวกระทำ ระดับบิ ต NOT จะเขียนโดยใช้ สัญลักษณ์เครื่องหมาย ~ ตัวกระทำ นี้จะใช้งานกับตัวถูกกระทำเพียงตัวเดียวที่อยู่ขวามือโดยทำการสลับบิตทุกบิตให้มีค่าตรงกันข้ามคือจาก “0” เป็น “1” และจาก “1” เป็น“0” ดังตัวอย่างนี้

0 1 Operand1

1 0 ~ operand1

```
int a = 103;            // binary: 0000000001100111
int b = ~a;             // binary: 1111111110011000
```

เมื่อกระทำแล้วทำให้ตัวแปร b มีค่า -104 (ฐานสิบ) ซึ่งคำตอบที่ได้ติดลบเนื่องจากบิตที่มีความสำคัญสูงสุด(บิตซ้ายมือสุด) ของตัวแปร int อันเป็นบิตแจ้งว่าตัวเลขเป็นบวกหรือลบมีค่าเป็น “1” แสดงว่าค่าที่ได้นี้ติดลบโดยในคอมพิวเตอร์จะเก็บค่าตัวเลขทั้งบวกและลบ

ตามระบบทูคอมพลีเมนติ (2’scomplement)การประกาศตัวแปร int ซึ่งมีความหมายเหมือนกับการประกาศตัวแปรเป็น signed int ต้องระวังค่าของตัวแปจะติดลบได้

8) คำสั่งเลื่อนบิตไปทางซ้าย (<<) และเลื่อนบิตไปทางขวา (>>)

ในภาษา C/C++ มีตัวกระทำเลื่อนบิตไปทางซ้าย << และเลื่อนบิตไปทางขวา >> ตัวกระทำนี้จะสั่งเลื่อนบิตของตัวถูกกระทำที่เขียนด้านซ้ายมือไปทางซ้ายหรือไปทางขวาตามจำนวนบิตที่ระบุไว้ในด้านขวามือของตัวกระทำรูปแบบคำสั่ง

```
variable << number_of_bits
variable >> number_of_bits
```

พารามิเตอร์variable เป็นตัวแปรเลขจำนวนเต็มที่มีจำนวนบิตน้อยกว่าหรือเท่ากับ 32 บิต (หรือตัวแปรประเภท byte, int หรือ long)

ตัวอย่างที่ 20

```
int a = 5;              // binary: 0000000000000101
int b = a << 3;     // binary: 0000000000101000,
// or 40 in decimal
int c = b >> 3;     // binary: 0000000000000101,
// or back to 5
```

ตัวอย่างที่ 21

เมื่อสั่งเลื่อนค่าตัวแปร x ไปทางซ้ายจำนวน y บิ ต (x << y) บิตข้อมูลที่อยู่ด้านซ้ายสุดของ x จำนวน y ตัวจะหายไปเนื่องจากถูกเลื่อนหายไปทางซ้ายมือ

```
int a = 5;              // binary: 0000000000000101
int b = a << 14;   // binary: 0100000000000000
```

การเลื่อนบิตไปทางซ้ายทำ ให้ค่าของตัวแปรด้านซ้ายมือของตัวกระทำถูกคูณด้วยค่าสองยกกำลังบิตที่เลื่อนไปทางซ้ายมือดังนี้

1 << 0 == 1

1 << 1 == 2

1 << 2 == 4

1 << 3 == 8

1 << 8 == 256

1 << 9 == 512

1 << 10 == 1024

เมื่อสั่งเลื่อนตัวแปร x ไปทางขวามือจำนวน y บิต (x >> y) จะมีผลแตกต่างกันขึ้นกับประเภทของตัวแปรถ้า x เป็นตัวแปรประเภท int ค่าที่เก็บได้มีทั้งค่าบวกและลบโดยบิตซ้ายมือสุดจะเป็น sign bit ถ้าเป็นค่าลบค่าบิตซ้ายมือสุดจะมี ค่าเป็น 1 กรณีนี้เมื่อสั่งเลื่อนบิตไปทางขวามือแล้วโปรแกรมจะนำค่า sign bit นี้มาเติมให้กับบิตทางซ้ายมือสุดไปเรื่อยๆปรากฏการณ์นี้เรียกว่า sign extension มี ตัวอย่างดังนี้

ตัวอย่างที่ 22

```
int x = -16; // binary: 1111111111110000
int y = x >> 3; // binary: 1111111111111110
```

ถ้าต้องการเลื่อนบิตไปทางขวามือแล้วให้ ค่า 0 มาเติมยังบิตซ้ายมือสุด (ซึ่งจะเกิดกับกรณีที่ตัวแปรเป็นประเภท unsigned int) เราสามารถทำได้โดยใช้การเปลี่ยนประเภทตัวแปรชั่วคราว (typecast) เพื่อเปลี่ยนให้ตัวแปร x เป็น unsigned int ชั่วคราวดังตัวอย่างต่อไปนี้

```
int x = -16;          // binary: 1111111111110000
int y = unsigned(x) >> 3;
// binary: 0001111111111110
```

ถ้าระมัดระวังเรื่อง sign extension แล้วคุณสามารถใช้ตัวกระทำ เลื่อนบิตไปทางขวามือสำหรับการหารค่าตัวแปรด้วย 2 ยกกำลังต่างๆได้ดังนี้

ตัวอย่างที่ 23

```
int x = 1000;
int y = x >> 3;    // integer division of
// 1000 by 8, causing y = 125.
```

2.2.3 ไวยากรณ์ภาษา C/C++ ของ Arduino

1).; (เซมิ โคลอน - semicolon)ใช้เขียนแจ้งว่า จบคำสั่ง

ตัวอย่างที่ 24

```
int a = 13;
```

บรรทัดคำสั่งที่ลืมเขียนปิดท้ายด้วยเซมิโคลอนจะทำให้แปลโปรแกรมไม่ผ่านโดยตัวแปรภาษาอาจจะแจ้งให้ทราบว่าไม่พบเครื่องหมายเซมิโคลอน หรือแจ้งเป็นการผิดพลาดอื่นๆ บางกรณีที่ตรวจสอบบรรทัดที่แจ้งว่าเกิดการผิดพลาดแล้วไม่พบที่ผิดให้ตรวจสอบบรรทัดก่อนหน้านั้น

2) { } (วงเล็บปีกกา curly brace)

เครื่องหมายวงเล็บปีกกาเป็นส่วนสำคัญของภาษาซีโดยมีการใช้งานต่างตำแหน่ง สร้างความสับสนให้กับผู้ที่เริ่มต้นวงเล็บปีกกาเปิด { จะต้องเขียนตามด้วยวงเล็บปีกกาปิด }ด้วยเสมอหรือที่เรียกว่าวงเล็บต้องครบคู่ในซอฟต์แวร์ Arduino IDE ที่ใช้เขียนโปรแกรมจะมีความสามารถในการตรวจสอบการครบคู่ของเครื่องหมายวงเล็บผู้ใช้งานเพียงแค่คลิกที่วงเล็บมันจะแสดงวงเล็บที่เหลือซึ่งเป็นคู่ของมัน

สำหรับโปรแกรมเมอร์มือใหม่และโปรแกรมเมอร์ที่ย้ายจากภาษา BASIC มาเป็นภาษา C มักจะสับสนกับการใช้เครื่องหมายวงเล็บแท้ที่จริงแล้วเครื่องหมายปีกกาปิดนี้เทียบได้กับคำ สั่ง RETURN ของ subroutine (function) หรือแทนคำสั่ง ENDIF ในการเปรียบเทียบและแทนคำสั่ง NEXT ของคำสั่งวนรอบ FOR

เนื่องจากมีการใช้วงเล็บปีกกาได้หลายดังนั้นเมื่อต้องการเขียนคำสั่งที่ต้องใช้เครื่องหมายวงเล็บเมื่อเขียนวงเล็บเปิดแล้วให้เขียนเครื่องหมายวงเล็บปิดทันทีถัดมาจึงค่อยเคาะปุ่ม Enter ในระหว่างเครื่องหมายวงเล็บเพื่อขึ้นบรรทัดใหม่ แล้วเขียนคำสั่งสิ่งที่ต้องการ ถ้าทำได้ตามนี้วงเล็บจะครบคู่แน่นอนสำหรับวงเล็บที่ไม่ครบคู่ทำให้เกิดการผิดพลาดตอนคอมไพล์โปรแกรมถ้าเป็นโปรแกรมขนาดใหญ่จะหาที่ผิดได้ยากตำแหน่งที่อยู่ของเครื่องหมายวงเล็บแต่ละตัวจะมีผลอย่างมากต่อไวยากรณ์ของภาษาคอมพิวเตอร์การย้ายตำแหน่งวงเล็บไปเพียงหนึ่งหรือสองบรรทัดทำให้ตัวโปรแกรมทำงานผิดไป

ตำแหน่งที่ใช้วงเล็บปีกกา

ฟังก์ชัน (Function)

ตัวอย่างที่25

```
void myfunction(datatype argument){
statements(s)
}
```

คำสั่งวนรอบ (Loops)

```
while (boolean expression)
{
statement(s)
}
do
{
statement(s)
} while (boolean expression);
for (initialisation; termination condition; incrementing expr)
```

```
{
statement(s)
}
```

คำสั่งทดสอบเงื่อนไข (condition)

```
if (boolean expression)
{
statement(s)
}
else if (boolean expression)
{
statement(s)
}
else
{
statement(s)
}
```

3) // และ /*...* หมายเหตุ บรรทัดเดียวและหลายบรรทัด

เป็นส่วนของโปรแกรมที่ผู้ใช้เขียนเพิ่มเติมว่าโปรแกรมทำงานอย่างไรโดยส่วนที่เป็นหมายเหตุจะไม่ถูกคอมไพล์ไม่นำไปประมวลผลมีประโยชน์มากสำหรับการตรวจสอบโปรแกรมภายหลังหรือใช้แจ้งให้เพื่อนร่วมงานหรือบุคคลอื่นทราบว่าบรรทัดนี้ใช้ทำอะไรตัวหมายเหตุภาษา C มี 2 ประเภทคือ

- หมายเหตุบรรทัดเดียวเขียนเครื่องสเลช // 2 ตัวหน้าบรรทัด
- หมายเหตุหลายบรรทัดเขียนเครื่องหมายสเลช / คู่กับดอกจัน *คร่อมข้อความที่เป็นหมายเหตุ เช่น/* blabla */

4) #define

เป็นคำสั่งที่ใช้งานมากในการกำหนดค่าคงที่ให้กับโปรแกรมในการกำหนดค่าคงที่ไม่ได้เปลืองพื้นที่หน่วยความจำของไมโครคอนโทรลเลอร์แต่อย่างไรเมื่อถึงขั้นตอนแปลภาษาคอมไพเลอร์จะแทนที่ตัวอักษรข้อความด้วยค่าที่กำหนดไว้ใน Arduino จะใช้ คำสั่ง #define ตรงกับภาษา Cรูปแบบ

```
#define constantName value
```

อย่าลืมเครื่องหมาย #

ตัวอย่างที่ 25

```
#define ledPin 3
```

เป็นการกำหนดให้ ตัวแปร ledPin เท่ากับค่าคงที่

เทคนิคสำหรับการเขียนโปรแกรมท้ายคำสั่ง #define ไม่ต้องมีเครื่องหมายเซมิโคลอนถ้าใส่เกินแล้วเวลาคอมไพล์ โปรแกรมจะแจ้งว่าเกิดการผิดพลาดในบรรทัดถัดไป

5) #include

ใช้สั่งให้รวมไฟล์อื่นๆเข้ากับไฟล์โปรแกรมหลักก่อนแล้วจึงทำการคอมไพล์โปรแกรมรูปแบบคำสั่ง

```
#include <file>
#include  “file”
```

ตัวอย่างที่ 26

```
#include  <stdio.h>
#include  "lcd.h"
```

บรรทัดแรกจะสั่งให้เรียกไฟล์ stdio.h มารวมกับไฟล์โปรแกรมหลักโดยค้นหาไฟล์จากตำแหน่งที่เก็บไฟล์ระบบของ Arduino โดยปกติเป็นไฟล์มาตรฐานที่มาพร้อมกับ Arduino

บรรทัดที่ 2 สั่งให้รวมไฟล์ lcd.h มารวมกับไฟล์โปรแกรมหลักโดยหาไฟล์จากตำแหน่งที่อยู่ของไฟล์ภาษา C หลักก่อนปกติเป็นไฟล์ที่ผู้ใช้สร้างขึ้นเอง

2.2.4 ตัวแปร

ตัวแปรเป็นตัวอักษรหลายตัวๆ ที่กำหนดขึ้นในโปรแกรมเพื่อใช้ในการเก็บค่าข้อมูลต่างๆ เช่น ค่าที่อ่านได้จากตัวตรวจจับที่ต่ออยู่กับขาพอร์ตแอนะล็อก.ของ Arduino ตัวแปรมีหลายประเภทดังนี้

1) char ตัวแปรประเภทตัวอักขระ

เป็นตัวแปรที่มีขนาด 1 ไบต์ (8 บิต)มีไว้เพื่อเก็บค่าตัวอักษรตัวอักษรในภาษาซีจะเขียนอยู่ในเครื่องหมายคำพูดขีดเดียวเช่น ‘A’ (สำหรับข้อความที่ประกอบจากตัวอักษรหลายตัวเขียนต่อกันจะเขียนอยู่ในเครื่องหมายคำพูดปกติเช่น “ABC”) คุณสามารถสั่งกระทำทางคณิตศาสตร์กับตัวอักษรได้ในกรณีจะนำค่ารหัสASCII ของตัวอักษรมาใช้เช่น ‘A’ +1 มีค่าเท่ากับ 66 เนื่องจากค่ารหัส ASCII ของตัวอักษร A เท่ากับ 65

รูปแบบคำสั่ง

```
char sign = ' ';
```

พารามิเตอร์

```
char var = ‘x’;
```

var คือชื่อของตัวแปรประเภท char ที่ต้องการ

x คือค่าที่ต้องการกำหนดให้ตัวแปรในที่นี้เป็นตัวอักษรหนึ่งตัว

2) byte ตัวแปรประเภทตัวเลข 8 บิตหรือ 1 ไบต์

ตัวแปร byte ใช้เก็บค่าตัวเลขขนาด 8 บิต มีค่าได้จาก 0 – 255

ตัวอย่างที่ 26

```
byte b = B10010111; // "B" is the binary formatter (151 decimal)
```

3) int ตัวแปรประเภทตัวเลขจำนวนเต็ม

ย่อจาก interger ซึ่งแปลว่าเลขจำนวนเต็ม int เป็นตัวแปรพื้นฐานสำหรับเก็บตัวเลขตัวแปรหนึ่งตัวมีขนาด 2 ไบต์ เก็บค่าได้ จาก -32,768 ถึง 32,767 (ค่าต่ำสุดจาก -215 ค่าสูงสุดจาก (215- 1)

ในการเก็บค่าตัวเลขติดลบจะใช้เทคนิคที่เรียกว่าทูคอมพลีเมนต์ (2's complement) บิตสูงสุดบางทีจะเรียกว่าเป็นบิตเครื่องหมายหรือ sign bit ถ้ามีค่าเป็น "1" แสดงว่าค่าติดลบ

ใน Arduino จะจัดการกับตัวเลขค่าติดลบให้เองทำให้นำค่าตัวแปรไปคำนวณได้อย่างถูกต้อง อย่างไรก็ตามเมื่อนำตัวเลขค่าติดลบนี้ไปเลื่อนบิตไปทางขวา (>>) จะมีปัญหาเรื่องค่าของตัวเลขที่ผิดพลาด

รูปแบบคำสั่ง

int var = val;

พารามิเตอร์

var คือชื่อของตัวแปรประเภท int ที่ต้องการ

val คือค่าที่ต้องการกำหนดให้กับตัวแปร

ตัวอย่างที่ 27

```
int ledPin = 31;
```

เทคนิคสำหรับการเขียนโปรแกรมเมื่อตัวแปรมีค่ามากกว่าค่าสูงสุดที่เก็บได้จะเกิดการ " ล้นกลับ " (roll over) ไปยังค่าต่ำสุดที่เก็บได้และเมื่อมีค่าน้อยกว่าค่าต่ำสุดที่เก็บได้จะล้นกลับไปยังค่าสูงสุดดังตัวอย่างต่อไปนี้

ตัวอย่างที่ 28

```
int x
x = -32,768;
x = x - 1; // x now contains 32,767
// - rolls over in neg. direction
x = 32,767;
x = x + 1; // x now contains -32,768 - rolls over
```

4) unsigned int ตัวแปรประเภทเลขจำนวนเต็มไม่คิดเครื่องหมาย

ตัวแปรประเภทนี้กับตัวแปร int ตรงที่ใช้หน่วยความจำ 2 ไบต์ แต่จะเก็บเลขจำนวนเต็มบวกเท่านั้นโดยเก็บค่า 0 ถึง 65,535 (216 -1)

รูปแบบคำสั่ง

unsigned int var = val;

พารามิเตอร์

var คือชื่อของตัวแปร int ที่ต้องการ

val คือค่าที่ต้องการกำหนดให้กับตัวแปร

ตัวอย่างที่ 29

unsigned int ledPin = 31;

เทคนิคสำหรับการเขียนโปรแกรมเมื่อตัวแปรมีค่ามากกว่าค่าสูงสุดจะล้นกลับไปค่าต่ำสุดและเมื่อมีค่าน้อยกว่าค่าต่ำสุดจะล้นกลับเป็นค่าสูงสุดดังตัวอย่าง

ตัวอย่างที่ 30

```
unsigned int x
x = 0;
x = x - 1; // x now contains 65535
// - rolls over in neg direction
x = x + 1; // x now contains 0 - rolls over
```

5) long ตัวแปรประเภทเลขจำนวนเต็ม 32 บิต

เป็นตัวแปรเก็บค่าเลขจำนวนเต็มที่ขยายความจุ เพิ่มจากตัวแปร int โดยตัวแปร long หนึ่งตัวกินพื้นที่หน่วยความจำ 32 บิต (4 ไบต์) เก็บค่าได้ จาก -2,147,483,648 ถึง 2,147,483,647

รูปแบบคำสั่ง

long var = val;

พารามิเตอร์

var คือชื่อของตัวแปร long ที่ต้องการ

val คือค่าที่ต้องการกำหนดให้กับตัวแปร

ตัวอย่างที่ 31

```
long time;
void setup()
{
Serial.begin(9600);
}
void loop()
{
Serial.print("Time: ");
time = millis();
Serial.println(time); //prints time since program started
delay(1000); // wait a second so as not to send massive amounts of data
}
```

6) unsigned long ตัวแปรประเภทเลขจำนวนเต็ม 32 บิต แบบไม่คิดเครื่องหมายเป็นตัวแปรเก็บค่าเลขจำนวนเต็มบวกตัวแปรหนึ่งตัวกินพื้นที่หน่วยความจำ 32 บิต (4 ไบต์)เก็บค่าได้จาก0 ถึง 4,294,967,295 หรื อ 232 – 1

รูปแบบคำสั่ง

```
unsigned long var = val;
```

พารามิเตอร์

var คือชื่อของตัวแปร unsigned long ที่ต้องการ

val คือค่าที่ต้องการกำหนดให้กับตัวแปร

ตัวอย่างที่ 32

```
long time;
void setup()
{
Serial.begin(9600);
}
void  loop()
{
Serial.print("Time:  ");
```

```
time = millis();
Serial.println(time); //prints time since program started
delay(1000); // wait a second so as not to send massive amounts of data
}
```

7) float ตัวแปรประเภทเลขทศนิยม

เป็นตัวแปรสำหรับเก็บค่าเลขเลขทศนิยมนิยมใช้เก็บค่าสัญญาณอะนาลอกหรือค่าที่ ต่อเนื่องตัวแปรแบบนี้เก็บค่าได้ละเอียดกว่าตัวแปร int โดยเก็บค่าได้ในช่วง 3.4028235 x 1038 ถึง -3.4028235 x 1038 ตัวแปรหนึ่งตัวจะใช้พื้นที่ หน่วยความจำ 32 บิต (4 ไบต์)ในการคำนวณคณิตศาสตร์กับตัวแปร float จะช้ากว่าการคำนวณของตัวแปร int ดังนั้นพยายามหลีกเลี่ยงการคำนวณกับตัวแปร float เช่นในคำสั่งวนรอบที่ทำงานด้วยความเร็วสูงสุดสำหรับฟังก์ชันทางเวลาที่ต้องแม่นยำอย่างมากโปรแกรมเมอร์บางคนจะทำการแปลงตัวเลขทศนิยมให้เป็นเลขจำนวนเต็มก่อนแล้วจึงคำนวณเพื่อให้ทำงานได้เร็วขึ้น

จะเห็นได้ว่าการคำนวณคณิตศาสตร์ของเลข floating point จะมีการใช้งานมากสำหรับการคำนวณค่าข้อมูลที่รับจากภายนอกซึ่งเป็นตัวเลขทศนิยมซึ่งทางผู้ศึกษาระบบไมโครคอนโทรลเลอร์มักจะมองข้ามไป

รูปแบบคำสั่ง

```
float var = val;
```

พารามิเตอร์var คือชื่อของตัวแปร float ที่ต้องการ

val คือค่าที่ต้องการกำหนดให้กับตัวแปร

ตัวอย่างที่ 33

```
float myfloat;
float sensorCalbrate = 1.117;
```

ตัวอย่างที่ 34

```
int x;
int y;
float z;
x = 1;
y = x / 2; // y now contains 0,
// integers can’t hold fractions
z = (float)x / 2.0;
// z now contains .5
```

```
// (you have to use 2.0, not 2)
```

8) double : ตัวแปรประเภทเลขทศนิยมความละเอียดสองเท่า

เป็นตัวแปรทศนิยมความละเอียดสองเท่ามีขนาด 8 ไบต์ค่าสูงสุดที่เก็บได้คือ 1.7976931348623157 x 10308ใน Arduino มีหน่วยความจำจำกัดจึงไม่นิยมใช้ตัวแปรประเภทนี้

9) string : ตัวแปรประเภทข้อความ

เป็นตัวแปรเก็บข้อความซึ่งในภาษาซีจะนิยามเป็นอะเรย์ของตัวแปรประเภท char

ตัวอย่างที่ 34 ตัวอย่างการประกาศตัวแปรสตริง

```
char Str1[15];
char Str2[8] = {‘a’,‘r’,‘d’,‘u’,‘i’,‘n’,‘o’};
char Str3[8] = {‘a’,‘r’,‘d’,‘u’,‘i’,‘n’, ‘o’,’\0'};
char Str4[ ] = “arduino”;
char Str5[8] = “arduino”;
char Str6[15] = “arduino”
```

Str1 เป็นการประกาศตัวแปรสตริงโดยไม่ได้กำหนดค่าเริ่มต้น

Str2 ประกาศตัวแปรสตริงพร้อมกำหนดค่าให้กับข้อความทีละตัวอักษรจากตัวอย่าง คอมไพเลอร์จะเพิ่ม null character ให้เอง

Str3 ประกาศตัวแปรสตริงพร้อมกำหนดค่าให้กับข้อความทีละตัวอักษรจากตัวอย่างเพิ่มค่า nullstring เอง

Str4 ประกาศตัวแปรสตริงพร้อมกำหนดค่าตัวแปรในเครื่องหมายคำพูดจากตัวอย่างไม่ได้กำหนดขนาดตัวแปรคอมไพเลอร์จะกำหนดขนาดให้เองตามจำนวนตัวอักษร + 1 สำหรับ null string

Str5 ประกาศตัวแปรสตริงพร้อมกำหนดค่าตัวแปรในเครื่องหมายคำพูดจากตัวอย่างกำหนดขนาดตัวแปรเอง

Str6 ประกาศตัวแปรสตริงโดยกำหนดขนาดเผื่อไว้สำหรับข้อความอื่นที่ยาวมากกว่านี้

10) การเพิ่มตัวอักษรแจ้งว่าจบข้อความ (null termination)

ในตัวแปรสตริงของภาษาซีกำหนดให้ตัวอักษรสุดท้ายเป็นตัวแจ้งการจบข้อความ (null string) ซึ่งก็คือตัวอักษร \0 ในการกำหนดขนาดของตัวแปร(ค่าในวงเล็บเหลี่ยม)จะต้องกำหนดให้เท่ากับจำนวน

ตัวอักษร+ 1 ดังในตัวแปร Str2 และ Str3 ในตัวอย่างที่ 3-35 ที่ข้อความ Arduino มีตัวอักษร 7 ตัวในการประกาศตัวแปรต้องระบุเป็น [8]

ในการประกาศตัวแปรสตริงต้องเผื่อพื้นที่สำหรับเก็บตัวอักษรแจ้งว่าจบข้อความมิฉะนั้นคอมไพเลอร์จะแจ้งเตือนว่าเกิดการผิดพลาด ในตัวอย่างที่ 3-35 ตัวแปร Strl และ Str 6 เก็บข้อความได้สูงสุด 14 ตัวอักษร

11) เครื่องหมายคำพูดขีดเดียวและสองขีด

ปกติแล้วจะกำหนดค่าตัวแปรสตริงภายในเครื่องหมายคำพูดเช่น "Abc" สำหรับตัวแปรตัวอักษร (char) จะกำหนดค่าภายในเครื่องหมายคำพูดขีดเดียว 'A'

12) ตัวแปรอะเรย์ (array)

ตัวแปรอะเรย์เป็นตัวแปรหลายตัวที่ถูกเก็บรวมอยู่ในตัวแปรชื่อเดียวกันโดยอ้างถึงตัวแปรแต่ละตัวด้วยหมายเลขดัชนีที่เขียนอยู่ในวงเล็บสี่เหลี่ยมตัวแปรอะเรย์ของ Arduino จะอ้างอิงตามภาษาซีตัวแปรอะเรย์อาจจะซับซ้อนแต่ใช้แค่ตัวแปรอะเรย์อย่างง่ายจะตรงไปตรงมาตัวอย่างการประกาศตัวแปรอะเรย์

ตัวอย่างที่ 35

```
int myInts[6];
int myPins[] = {2, 4, 8, 3, 6};
int mySensVals[6] = {2, 4, -8, 3, 2};
char message[6] = “hello”;
```

เราสามารถประกาศตัวแปรอะเรย์ได้โดยยังไม่กำหนดค่าดังตัวแปร myIntsในตัวแปร myPins จะประกาศตัวแปรอะเรย์โดยไม่ระบุขนาดซึ่งทำได้เมื่อประกาศตัวแปรแล้วกำหนดค่าทันทีเพื่อให้คอมไพเลอร์นับว่าตัวแปรมี สมาชิกกี่ตัวและกำหนดค่าได้ถูกต้อง

ท้ายที่สุด สามารถประกาศตัวแปรและกำหนดขนาดของตัวแปรอะเรย์ ดังตัวแปร my Sens Vals ในการประกาศอะเรย์ของตัวแปร char จะต้องเผื่อที่สำหรับเก็บค่าตัวอักษรแจ้งว่าจบข้อความด้วย

13) การใช้งานตัวแปรอะเรย์

การใช้งานตัวแปรอะเรย์ทำได้โดยการพิมพ์ชื่อตัวแปรพร้อมกับระบุค่าดัชนีภายในเครื่องหมายวงเล็บสี่เหลี่ยมค่าดัชนีของตัวแปรอะเรย์เริ่มต้นด้วยค่า 0 ดังนั้นค่าของตัวแปร mySensVals มีค่าดังนี้

mySensVals[0] == 2, mySensVals[1] == 4 , ฯลฯ

การกำหนดค่าให้กับตัวแปรอะเรย์

```
mySensVals[0] = 10;
```

การเรียกค่าสมาชิกของตัวแปรอะเรย์

```
x = mySensVals[4];
```

14) อะเรย์ และคำสั่งวนรอบ for

โดยทั่วไปเราจะพบการใช้งานตัวแปรอะเรย์ภายในคำสั่ง for โดยใช้ ค่าตัวแปรนับรอบของคำสั่ง for เป็นค่าดัชนีของตัวแปรอะเรย์ ดังตัวอย่างการพิมพ์ค่าสมาชิกแต่ละตัวของตัวแปรอะเรย์ ผ่านพอร์ตอนุกรมให้เขียนโปรแกรมดังนี้

ตัวอย่างที่ 36

```
int i;
for (i = 0; i < 5; i = i + 1)
{
Serial.println(myPins[i]);
}
```

15) ขอบเขตของตัวแปร

ตัวแปรในภาษาซีที่ใช้ใน Arduino จะมีคุณสมบัติที่เรียกว่า “ขอบเขตของตัวแปร” (scope) ซึ่งแตกต่างจากภาษา BASIC ซึ่งตัวแปรทุกตัวมีสถานะเท่าเทียมกันหมดคือ เป็นแบบ global

ก) ตัวแปรโลคอลและโกลบอลตัวแปรแบบโกลบอล (global variable) เป็นตัวแปรที่ทุกฟังก์ชันในโปรแกรมรู้จักโดยต้องประกาศตัวแปรนอกฟังก์ชั่นสำหรับตัวแปรแบบโลคอลหรือตัวแปรท้องถิ่นเป็นตัวแปรที่ประกาศตัวแปรอยู่ภายในเครื่องหมายวงเล็บปีกกาของฟังก์ชั่นและรู้จักเฉพาะแค่ภายในฟังก์ชั่นนั้น

เมื่อโปรแกรมเริ่มมีขนาดใหญ่และซับซ้อนมากขึ้นการใช้ตัวแปรโลคอลจะมีประโยชน์มาก เนื่องจากแน่ใจได้ว่ามีแค่ฟังก์ชันนั้นเท่านั้นที่สามารถใช้งานตัวแปรช่วยป้องกันการเกิดการผิดพลาดเมื่อฟังก์ชั่นทำการแก้ไขค่าตัวแปรที่ใช้งานโดยฟังก์ชันอื่น

ตัวอย่างที่ 37

```
int gPWMval;            // any function will see this variable
void setup(){
// ...
}
void loop(){
int i;                                  // "i" is only “visible” inside of "loop"
float f;                                // "f" is only “visible” inside of "loop"
}
```

ข) ตัวแปรสแตติก (static)

เป็นคำสงวน (keyword) ที่ใช้ตอนประกาศตัวแปรที่มีขอบเขตใช้งานแค่ภายในฟังก์ชันเท่านั้นโดยต่างจากตัวแปรโลคอลตรงที่ตัวแปรแบบโลคอลจะถูกสร้างและลบทิ้งทุกครั้งที่เรียกใช้ฟังก์ชัน สำหรับตัวแปรสแตติกเมื่อจบการทำ งานของฟังก์ชันค่าตัวแปรจะยังคงอยู่ (ไม่ถูกลบทิ้ง)เป็นการรักษาค่าตัวแปรไว้ระหว่างการเรียกใช้งานฟังก์ชัน

ตัวแปรที่ประกาศเป็น static จะถูกสร้างและกำหนดค่าในครั้งแรกที่เรียกใช้ฟังก์ชัน

16) ค่าคงที่

ค่าคงที่เป็นกลุ่มตัวอักษรหรือข้อความที่ได้กำหนดค่าไว้ล่วงหน้าแล้วตัวคอมไฟเลอร์ของ Arduino จะรู้จักกับค่าคงที่เหล่านี้แล้วไม่จำเป็นต้องประกาศหรือกำหนดค่าคงที่

17) HIGH, LOW :ใช้กำหนดค่าทางตรรกะ

ในการอ่านหรือเขียนค่าให้กับขาที่เป็นดิจิตอลค่าที่เป็นได้ มี 2 ค่าคือ HIGH หรือ LOW เท่านั้น

HIGH เป็นการกำหนดค่าให้ขาดิจิตอลนั้นมีแรงดันเท่ากับ +5V ในการอ่านค่าถ้าอ่านได้ +3V หรือมากกว่าไมโครคอนโทรลเลอร์จะอ่านค่าได้ เป็น HIGH ค่าคงที่ของ HIGH ก็คือ “1” หรือเทียบเป็นตรรกะคือจริง (TRUE)

LOW เป็นการกำหนดค่าให้ขาดิจิตอลนั้นมีแรงดันเท่ากับ 0V ในการอ่านค่าถ้าอ่านได้ +2V หรือน้อยกว่าไมโครคอนโทรลเลอร์จะอ่านค่าได้ เป็น LOW ค่าคงที่ของ LOW ก็คือ “0”หรือเทียบเป็นตรรกะคือเท็จ (FALSE)

18) INPUT, OUTPUT : กำหนดทิศทางของขาพอร์ตดิจิตอล

ขาพอร์ตดิจิตอลทำหน้าที่ได้ 2 อย่างคือเป็นอินพุต (INPUT) หรือเอาต์พุต (OUTPUT)ซึ่งค่าคงที่นี้ก็ระบุไว้ชัดเจน

19) ตัวกระทำอื่นๆที่เกี่ยวข้องกับตัวแปร

cast : การเปลี่ยนประเภทตัวแปรชั่วคราวcast เป็นตัวกระทำที่ใช้สั่งให้เปลี่ยนประเภทของตัวแปรไปเป็นประเภทอื่นและบังคับให้คำนวณค่าตัวแปรเป็นประเภทใหม่

รูปแบบคำสั่ง(type)variableเมื่อ Type เป็นประเภทของตัวแปรใดๆ (เช่น int, float, long) Variable เป็นตัวแปรหรือค่าคงที่ ใดๆ

ตัวอย่างที่ 38

```
int i;
float f;
f = 3.6;
i = (int) f;              // now i is 3
```

ในการเปลี่ยนประเภทตัวแปรจาก float เป็น int ค่าที่ได้จะถูกตัดเศษออกดังนั้น (int)3.2และ (int) 3.7 มีค่าเท่ากันคือ 3

20) sizeof แจ้งขนาดของตัวแปรใช้แจ้งบอกจำนวนไบต์ของตัวแปรที่ต้องการทราบค่าซึ่งเป็นทั้งตัวแปรปกติและตัวแปรอะเรย์รูปแบบคำสั่งเขียนได้ทั้งสองแบบดังนี้

```
Sizeof (variable)
sizeof variable
```

เมื่อ Variable คือตัวแปรปกติหรือตัวแปรอะเรย์ (int, float, long) ที่ต้องการทราบขนาด

ตัวอย่างที่ 39

ตัวกระทำ sizeof มีประโยชน์อย่างมากในการจัดการกับตัวแปรอะเรย์ (รวมถึงตัวแปรสตริง)

ตัวอย่างต่อไปนี้จะพิมพ์ข้อความออกทางพอร์ตอนุกรมครั้งละหนึ่งตัวอักษรให้ทดลองเปลี่ยนข้อความ

```
char myStr[ ] = "this is a test";
int i;
void setup()
{
Serial.begin(9600);
}
void loop()
{
for (i = 0; i < sizeof(myStr) - 1; i++)
{
Serial.print(i, DEC);
Serial.print(" = ");
Serial.println(myStr[i], BYTE);
}
}
```

13. ค่าสงวนของ Arduino

ค่าสงวน คือ ค่าคงที่ ตัวแปรและฟังก์ชันที่ได้กำหนดไว้เป็นส่วนหนึ่งของภาษา C ของ Arduino ห้ามนำคำเหล่านี้ไปตั้งชื่อตัวแปรแสดงได้ดังนี้

# Constants	switch	analogRead	min
HIGH	throw	analogWrite	new
LOW	try	attachInterrupts	noInterrupts
INPUT	unsigned	asin	noTone
OUTPUT	void	atan	null
SERIAL	while	atan2	parseInt
DISPLAY	# Other	available	parseFloat
PI	+=	begin	pinMode
HALF_PI	+[	bit	print
TWO_PI	]	bitRead	println
LSBFIRST	=&&	bitWrite	pulseIn
MSBFIRST	=	bitSet	radians
CHANGE	||	bitClear	read
FALLING	=	boolean	readBytes
RISING	,/	byte	readBytesUntil
false	/	case	return
true	?:	ceil	round
null	<<	char	serial
# Port Constants	<<	char	serial1
DDRB	=	class	serial2
PINB	log	constrain	serial3
PORTB	&&	cos	setTimeout
DDRC	!|	default	Setup
PINC	|	delay	shiftIn
PORTC	^^	delayMicroseconds	shiftOut
DDRD	=	detachInterrupts	sin

PIND	++	digitalWrite	sq
PORTD	!=	digitalRead	sqrt
# Datatypes	-%/	else	tan
boolean	*	exp	this
byte	*{	false	tone
char	}	find	true
class	—/ /**	findUntil	write
default	.-	float	# USB
do	=	floor	Keyboard
double	==	for	Mouse
int	<<	HALF_PI	press
long	=	if	release
private	()	int	releaseAll
protected	>>	log	accept
public	;	loop	click
return	abs	map	move
short	acos	max	isPressed
signed		micros	
static		millis	

2.3 Adapter

Adaptor อแดปเตอร์ คือหม้อแปลงไฟฟ้า จากไฟฟ้ากระแสสลับ (AC) ที่ใช้ทั่วๆ ไปตามบ้าน ที่มีความต่างศักย์ 220 โวลต์ ให้เป็นไฟฟ้ากระแสตรง (DC) ที่มีความต่างศักย์ต่ำลง เพื่อให้สามารถจ่ายกระแสไฟฟ้ากับเครื่องใช้ไฟฟ้าได้ หรือจ่ายไฟให้กับตัวกล้องวงจรปิด แสดงในรูปที่ 2.3

รูปที่ 2.3 Adaptor

1) การเลือกใช้ Adaptor

กระแสไฟฟ้าที่จ่ายให้กับตัวกล้อง 12VDC 1A มีการจ่ายไฟเกินหรือไฟขาดหรือไม่ สามารถจ่ายไฟได้คุณภาพคงที่หรือไม่ เพราะปกติของกล้องวงจรปิด ถ้าเป็นกล้องของ AVTECH นั้น ส่วนใหญ่แล้วจะให้ค่าเพิ่มหรือลดได้ไม่เกิน 10% ค่าตัวเลขก็คือ ไม่ควรเกิน 13.1V และไม่ควรต่ำกว่า 10.8V

2) ความร้อนที่เกิดขึ้นจากการใช้งาน

อุปกรณ์ไฟฟ้าทุกอย่างมีความร้อนและตัวอแดปเตอร์ ถ้าเป็นแบบคุณภาพต่ำ ตัว Case ขออแดปเตอร์ที่เป็นพลาสติกสามารถติดไฟได้ง่ายๆเหมือนกันดังนั้นควรเลือกซื้ออแดปเตอร์จากแหล่งผลิตที่ไวใจได้

3) มาตรฐานวัสดุอุปกรณ์ที่ใช้

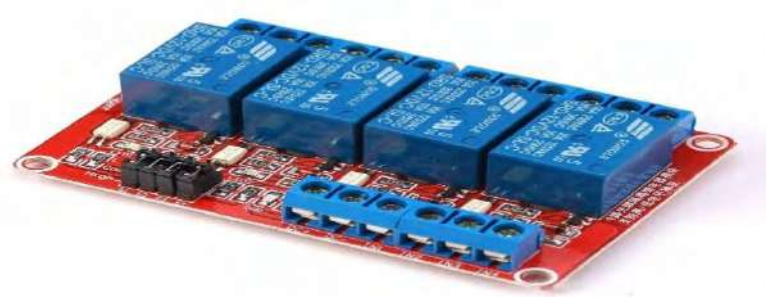
บางครั้งเราก็ไม่สามารถรู้ได้ว่าอุปกรณ์ที่ใช้ข้างในเป็นแบบไหนบ้าง ของแท้หรือไม่หรือการที่ตรวจเช็ค และดูตามท้องตลาด ตอนนี้มีการพัฒนาการ Cppy จะมีตั้งแต่ Copy เกรด A B C D และลักษณะเหมือนกัน

4) การตรวจสอบ

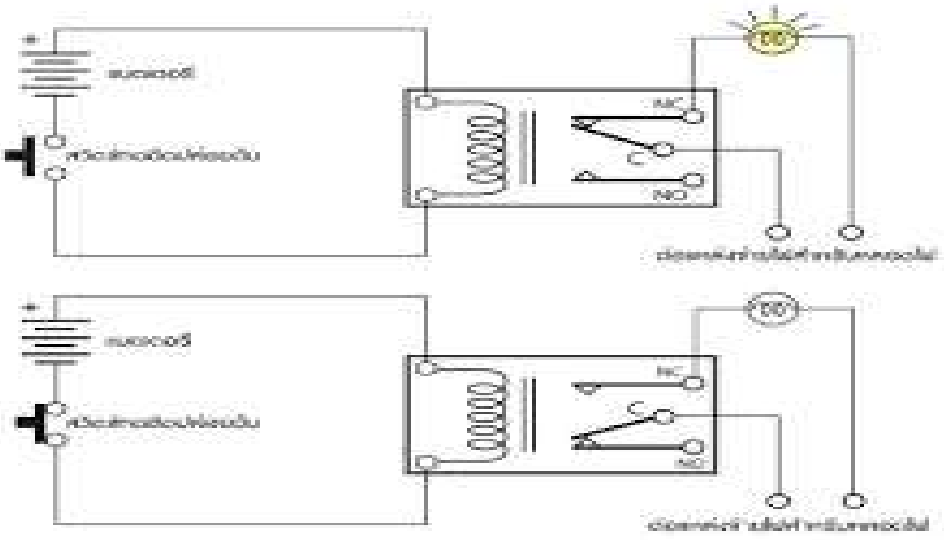
บางครั้งผู้ผลิตทำออกมาขายแล้วอาจจะไม่ได้ QC เพราะอาจจะเนื่องจากลดค่าแรง ลดเวลา ทำให้ราคาอแดปเตอร์ถูกลงซึ่งถ้าเกิดเสีย 1 ตัวก็ไม่คุ้มแล้ว

2.4 Relay Module

1) Relay รีเลย์ (Relay) เป็นอุปกรณ์ไฟฟ้าชนิดหนึ่ง ซึ่งทำหน้าที่ตัดต่อวงจรแบบเดียวกับสวิตช์ โดยควบคุมการทำงานด้วยไฟฟ้า Relay มีหลายประเภท ตั้งแต่ Relay ขนาดเล็กที่ใช้ในงานอิเล็กทรอนิกส์ทั่วไป จนถึง Relay ขนาดใหญ่ที่ใช้ในงานไฟฟ้าแรงสูง โดยมีรูปร่างหน้าตาแตกต่างกันออกไป แต่มีหลักการทำงานที่คล้ายคลึงกัน สำหรับการนำ Relay ไปใช้งาน จะใช้ในการตัดต่อวงจร ทั้งนี้ Relay ยังสามารถเลือกใช้งานได้หลากหลายรูปแบบ แสดงในรูปที่ 2.4 และ 2.5



รูปที่ 2.4 Relay Module



รูปที่ 2.5 วงจร Relay

2) อุปกรณ์ภายในของ Relay

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

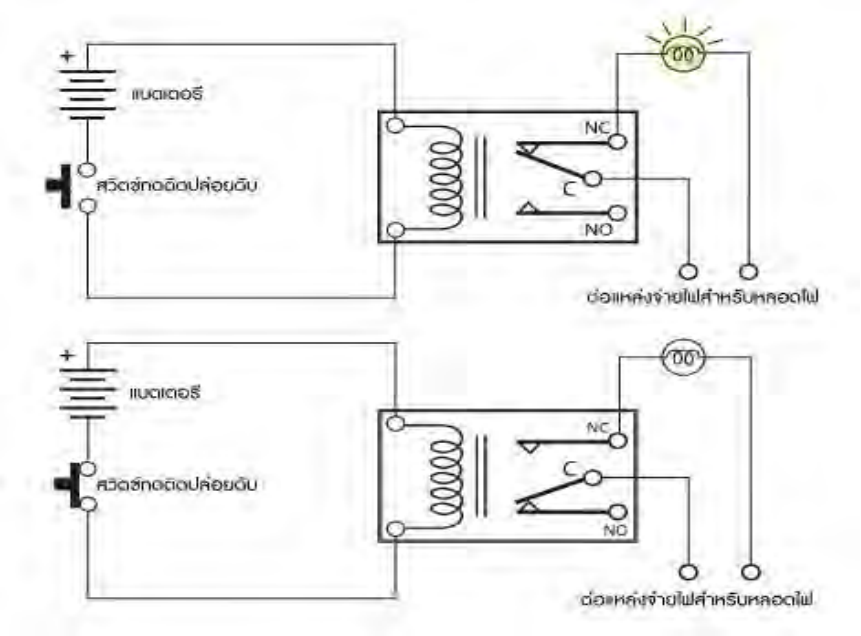
หน้าสัมผัส NO (Normally Open) เป็นหน้าสัมผัสปกติเปิด โดยในสภาวะปกติจะลอยอยู่ ไม่ถูกต่อกับขา COM (Common) แต่จะเชื่อมต่อกันเมื่อมีกระแสไฟไหลผ่านขดลวดขา COM(Common) เป็นขาที่ถูกใช้งานร่วมกันระหว่าง NC และ NO ขึ้นอยู่กับว่า ขณะนั้นมีกระแสไฟฟ้าไหลผ่านขดลวดหรือไม่ หน้าสัมผัสใน Relay 1 ตัวอาจมีมากกว่า 1 ชุด ขึ้นอยู่กับผู้ผลิตและลักษณะของงานที่ถูกนำไปใช้ จำนวนหน้าสัมผัสถูกแบ่งออกดังนี้

สวิตช์จะถูกแยกประเภทตามจำนวน Pole และจำนวน Throw ซึ่งจำนวน Pole (SP-Single Pole, DP-Double Pole, 3P-Triple Pole, etc.) จะบอกถึงจำนวนวงจรที่ทำการเปิด-ปิดหรือจำนวนของขา COM นั่นเอง และจำนวน Throw (ST, DT) จะบอกถึงจำนวนของตัวเลือกของ Pole ตัวอย่างเช่น SPST- Single

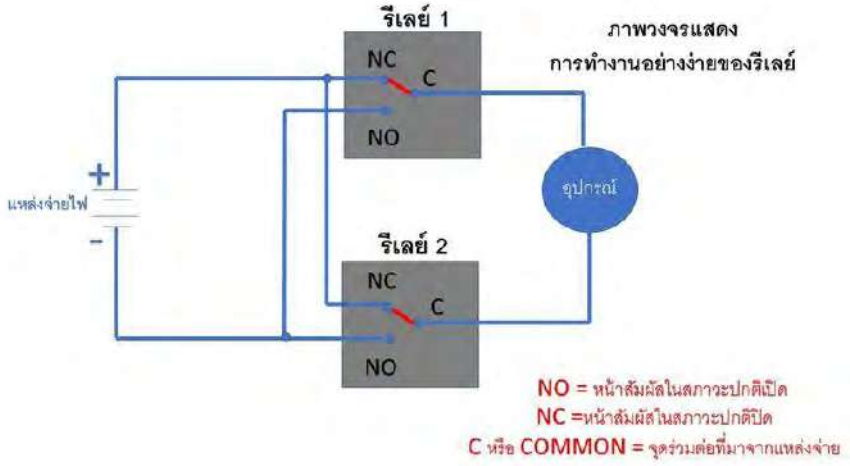
Pole Single Throw สวิตช์จะสามารถเลือกได้เพียงอย่างเดียวโดยจะเป็นปกติเปิด (NO-Normally Open) หรือปกติปิด (NC-Normally Close) แต่ถ้าเป็น SPDT- Single Pole Double Throw สวิตช์จะมีหนึ่งคู่เป็นปกติเปิด (NO) และอีกหนึ่งคู่เป็นปกติปิดเสมอ (NC) แสดงในรูปที่ 2.6 และ 2.7

SPST คือ Single Pole Single Throw DPST คือ Double Pole Single Throw

SPDT คือ Single Pole Double Throw DPDT คือ Double Pole Double Throw



รูปที่ 2.6 การสัมผัสหน้าของ Relay

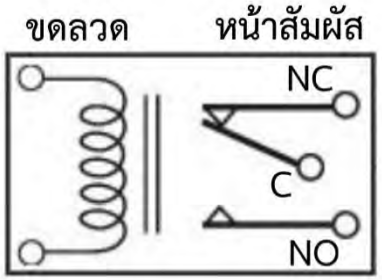


รูปที่ 2.7 วงจรการทำงาน Relay

จากส่วนประกอบข้างต้นที่ได้กล่าวไปในบทความนี้เราจะใช้งาน Relay แบบ SPDT (Single Pole Double Throw) หลักการทำงานของ Relay นั้นในส่วนของขดลวดเมื่อมีกระแสไฟฟ้าไหลผ่านจะทำให้ขดลวดเกิดการเหนี่ยวนำและทำหน้าที่เสมือนแม่เหล็กไฟฟ้า ส่งผลให้ขา COM ที่เชื่อมต่ออยู่กับหน้าสัมผัส NC (ในสภาวะที่ยังไม่เกิดการเหนี่ยวนำ) ย้ายกลับเชื่อมต่อกับหน้าสัมผัส NO แทน และปล่อยให้ขา NC ลอย เมื่อมองที่ขา NC กับ COM และ NO กับ COM แล้วจะเห็นว่ามีการทำงานติด-ดับ ลักษณะคล้ายการทำงานของสวิชต์ เราสามารถอาศัยคุณสมบัตินี้ไปประยุกต์ใช้งานได้

3) การสัมผัสแบบของ Relay

หน้าสัมผัสแบบ A (Form A) หมายถึง หน้าสัมผัสของ Relay ในสภาพปกติจะเปิดอยู่ (Normally open) และหน้าสัมผัสเป็นแบบ SPST ถ้าจะเขียนเป็นสัญลักษณ์ได้คือ แสดงในรูปที่ 2.8 และ 2.9



รูปที่ 2.8 หน้าสัมผัสของ Relay ในสภาพปกติเปิด

หน้าสัมผัสแบบ B (Form B) หมายถึงหน้าสัมผัสของ Relay ในสภาพปกติจะปิด (Normally close) และเป็นแบบ SPST เขียนเป็นสัญลักษณ์ได้คือ

รูปที่ 2.9 หน้าสัมผัสของ Relay ในสภาพปกติจะปิด

หน้าสัมผัสแบบ C (Form C) แบบนี้เรียกว่า "break, make หรือ transfer" เป็นหน้าสัมผัสแบบ SPDT เขียนสัญลักษณ์ได้ดังนี้

ข้อมูลเพิ่มเติม

- รองรับ แรงดัน AC 250V กระแส 10A และ DC 30V กระแส 10A
- หา trigger ใช้แรงดัน 5V กินกระแสอยู่ที่ 5mA
- สามารถตั้ง Active High หรือ Active Low ด้วยการใช้ Jumper
- ขนาดของบอร์ด : 50mm * 26mm * 18.5mm

การต่อใช้งาน

- DC + : ป้อนแรงดัน +12V
- DC - : ป้อนขั่วลบ
- IN1 : ป้อน Active High หรือ Active Low
- IN2 : ป้อน Active High หรือ Active Low
- IN3 : ป้อน Active High หรือ Active Low
- IN4 : ป้อน Active High หรือ Active Low

บทที่ 3

รายละเอียดการปฏิบัติงาน

3.1 ชื่อและที่ตั้งของสถานประกอบการ

ชื่อสถานประกอบการ : บริษัท เอส คิว ดี มาร์เก็ตติ้ง จำกัด

ที่อยู่ : 1527/1 ถนนเจริญนคร แขวงบางลำภูล่าง เขตคลองสาน กรุงเทพมหานคร 10600

Tel:02-431-0100-9

อีเมล์ : sqdgroups.com

แสดงในรูปที่ 3.1

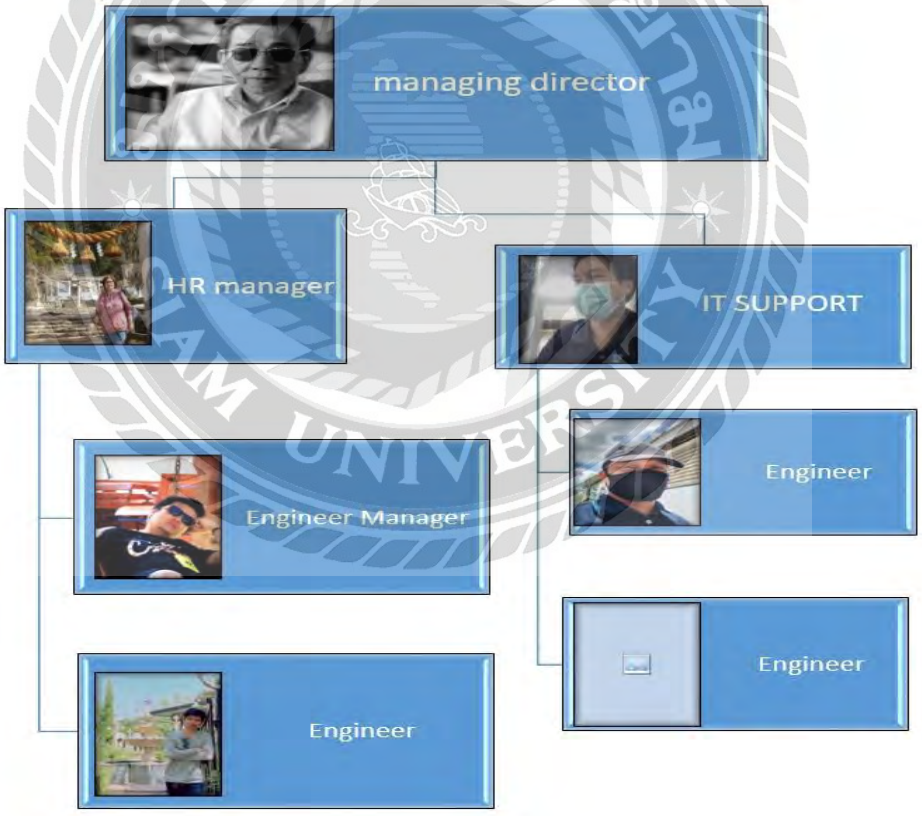
รูปที่ 3.1 ที่ตั้งบริษัทของสถานประกอบการ

3.2 ลักษณะการประกอบการและการให้บริการหลักขององค์กร

บริษัท เอส คิว ดี มาร์เก็ตติ้ง จำกัด เริ่มก่อตั้งในปีวันที่ 5 ม.ค. พ.ศ. 2539 โดยเริ่มต้นการขายส่งอุปกรณ์และชิ้นส่วนอิเล็กทรอนิกส์รวมไปถึงอุปกรณ์ไฟฟ้าประเภทต่างๆและมีการขยายกลุ่มสินค้าต่างๆเรื่อยมาจนถึงปัจจุบันจากจุดเริ่มต้นเล็กๆ และค่อยๆ เติบโตเรื่อยมาด้วยการขยายและปรับปรุงรูปแบบสินค้าให้มีความหลากหลายมากขึ้น เพื่อตอบสนองความต้องการของลูกค้าที่ต้องการในรูปแบบที่ซับซ้อนขึ้น

โดยนอกจากธุรกิจทั่วไปและธุรกิจเฉพาะทางแล้วยังได้ขยายผลิตภัณฑ์ไปในรูปแบบต่าง ๆ เป็นต้นและยังเป็นหนึ่งในผู้ริเริ่มในการนำมาใช้ในรูปแบบต่างๆจวบจนปัจจุบันเป็นผู้ผลิตอุปกรณ์ไฟฟ้าให้กับหน่วยงานและองค์กรต่างๆทั้งภาครัฐและเอกชนทั่วประเทศ

3.3 รูปแบบการจัดการองค์กรและการบริหารงาน



รูปที่ 3.2 แผนผังองค์กร

3.4 ตำแหน่งและลักษณะงานที่นักศึกษาได้รับมอบหมาย

3.4.1 ตำแหน่งที่นักศึกษาได้รับมอบหมายคือผู้ช่วยวิศวกร

3.4.2 ลักษณะงานที่นักศึกษาได้รับมอบหมายคือ ควบคุมเครื่องจักร ผลิตงาน ตามแผนงานและงานอื่นๆ ตามที่ได้รับมอบหมาย

3.5 ชื่อและตำแหน่งงานของพนักงานที่ปรึกษา

3.5.1 ชื่อพนักงานที่ปรึกษา นายกิติพงษ์ เทียนถาวร

3.5.2 ตำแหน่งผู้จัดการฝ่ายคอมพิวเตอร์

3.6 ระยะเวลาที่ปฏิบัติงาน

3.6.1 ระยะเวลาในการดำเนินงาน ตั้งแต่วันที่ 23 สิงหาคม ถึงวันที่ 10 ธันวาคม พ.ศ. 2564

3.6.2 วันเวลาในการปฏิบัติสหกิจศึกษา เวลา 08.00 – 17.00 น. หยุดตามปฏิทินบริษัท

3.7 ขั้นตอนและวิธีดำเนินงาน

3.7.1 ขั้นตอนการดำเนินงาน

ตารางที่ 3.1 ขั้นตอนการดำเนินการ

ลำดับ	ขั้นตอนการดำเนินการ	สิงหาคม 64			กันยายน 64				ตุลาคม 64				ธันวาคม 64		
1	ศึกษาการทำงาน														
2	รวบรวมปัญหาการหยุดกระบวนการผลิต														
3	ยื่นเสนอโครงงาน														
4	อนุมัติโครงงาน														
5	ดำเนินการ														
6	ติดตามผลการดำเนินงาน														
7	สรุปผล														
8	ขยายผลทำแผน PM														
9	จัดทำรูปเล่มโครงงาน														

3.8 อุปกรณ์เครื่องมือที่ใช้

3.8.1 เครื่องมือช่าง เช่น ไขควง คีม ประแจ สว่าน หินเจียร เป็นต้น

3.8.2 สีกันสนิม Module wi-fi, ESP8266 Arduino, Adapter, Breaker, Relay

3.8.3 สายไฟ หม้อแปลง

บทที่ 4
ผลการปฏิบัติงานตามโครงงาน

4.1 การสำรวจระบบไฟฟ้าแสงสว่างเดิมที่จะปรับเปลี่ยนให้สามารถควบคุมด้วย IoT

การตรวจสอบและออกแบบระบบควบคุมไฟฟ้าในอาคารด้วย IoT (Internet of Things) ที่จะนำมาปรับเปลี่ยนระบบเดิม เริ่มต้นจาการสำรวจวงจรไฟฟ้าแสงสว่างของระบบเดิม เพื่อวางแผนการปรับเปลี่ยนว่าจะติดตั้งอุปกรณ์ควบคุมต่างๆ ที่เพิ่มมาในตำแหน่งใดบ้าง

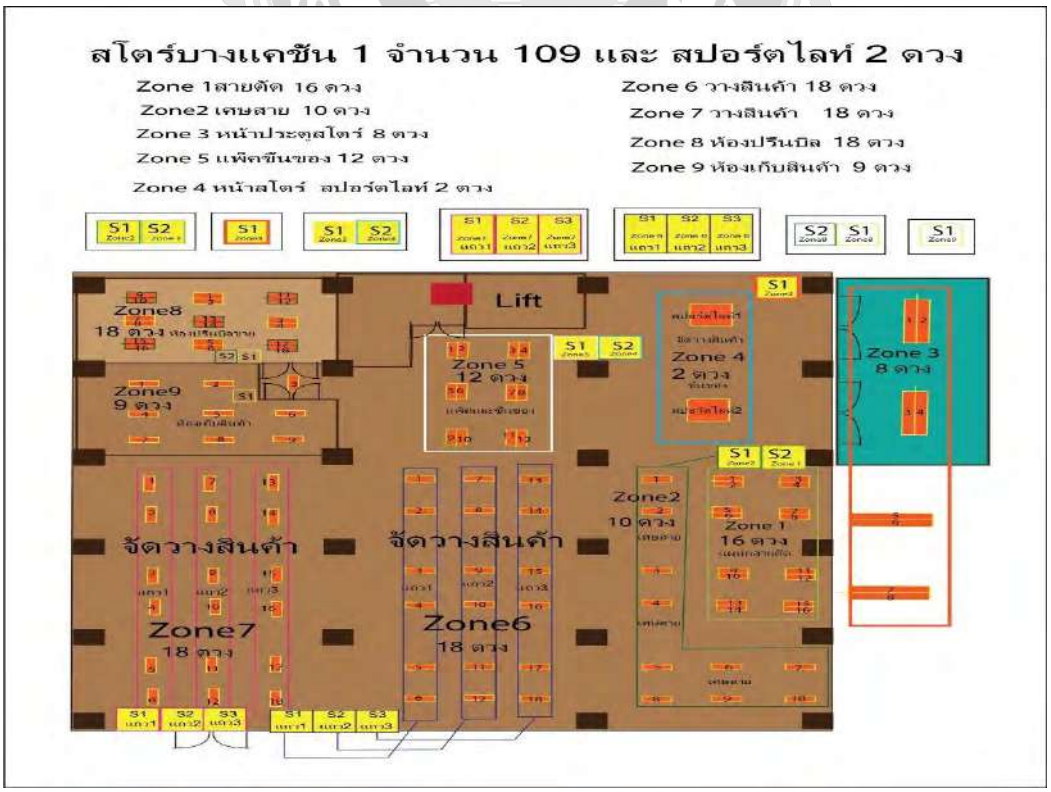
รูปที่ 4.1 การสำรวจระบบไฟฟ้าแสงสว่างที่ต้องการปรับเปลี่ยนระบบการควบคุมเป็น IoT

รูปที่ 4.2 การสำรวจระบบไฟฟ้าแสงสว่างที่ต้องการปรับเปลี่ยนระบบการควบคุมเป็น IoT

รูปที่ 4.3 การสำรวจระบบไฟฟ้าแสงสว่างที่ต้องการปรับเปลี่ยนระบบการควบคุมเป็น IoT

4.2 การออกระบบไฟฟ้าแสงสว่างที่ปรับเปลี่ยนให้สามารถควบคุมด้วย IoT

เมื่อดำเนินการสำรวจวงจรไฟฟ้าแสงสว่างของระบบเดิมแล้ว ทำการตรวจสอบพื้นที่ที่ต้องการติดตั้งอุปกรณ์เช่นขนาดห้อง ตำแหน่งปลั๊กไฟและวงจรแสงสว่าง ทำให้สามารถกำหนดโซนและจุดที่จะติดตั้งอุปกรณ์ที่สามารถควบคุมด้วยระบบ IoT ซึ่งจะนำมาใช้ใหม่ได้ดังรูปที่ 4.4

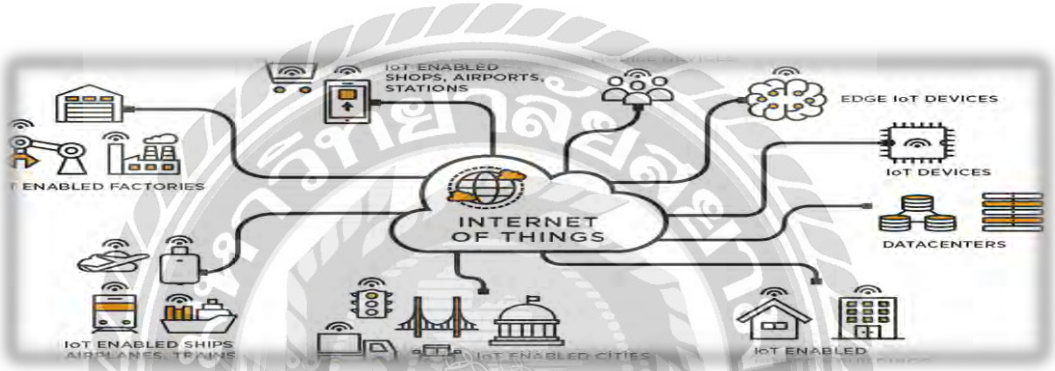


รูปที่ 4.4 ออกแบบระบบควบคุมไฟฟ้าแสงสว่างในอาคารโดยการกำหนดโซน

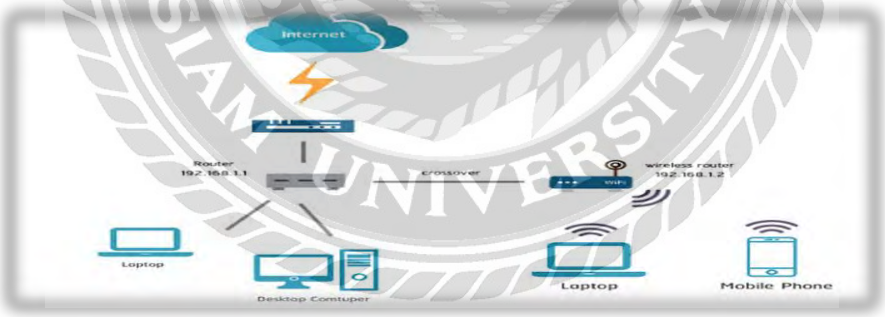
หลังจากการออกแบบระบบควบคุมไฟฟ้าในอาคารและได้กำหนดโซนเรียบร้อยแล้ว จากนั้นทำการสำรวจโครงสร้างเครือข่าย ตรวจสอบตำแหน่งที่สามารถวางเกตเวย์ได้ และประเมินเครือข่าย สำรวจสัญญาณ Wi-Fi สำรวจเครือข่ายอื่นๆในพื้นที่ให้ครอบคลุม

4.3 การออกแบบระบบเครือข่ายและการเชื่อมต่อ

วางแผนโครงสร้างเครือข่ายกำหนดการเชื่อมต่อระหว่างอุปกรณ์ IoT กับเกตเวย์ เลือกประเภทเครือข่ายที่เหมาะสม เช่น Wi-Fi, Bluetooth เลือกอุปกรณ์ IoT เลือกเซนเซอร์ (เช่น เซนเซอร์แสง, เซนเซอร์อุณหภูมิ) เลือกอุปกรณ์ควบคุม (เช่น สมาร์ทปลั๊ก, สมาร์ทสวิตช์) เลือกเกตเวย์ที่สามารถเชื่อมต่อกับอุปกรณ์ทั้งหมด



รูปที่ 4.5 รูปแบบระบบเครือข่ายและการเชื่อมต่อของระบบ IoT



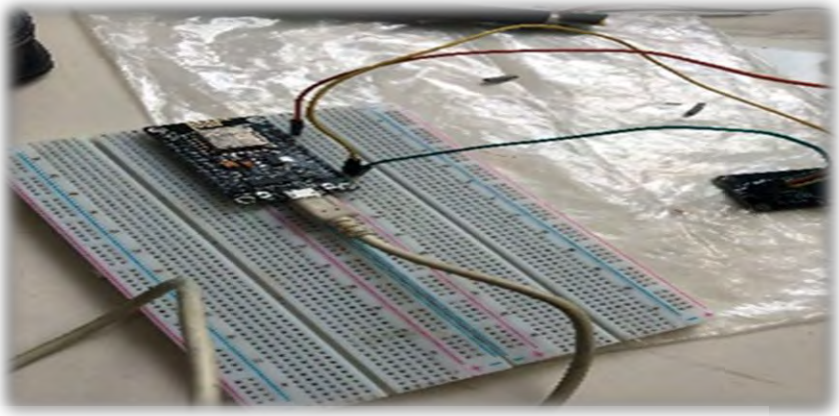
รูปที่ 4.6 รูปแบบระบบเครือข่ายและการเชื่อมต่อของระบบ IoT

เมื่อเชื่อต่อโครงข่ายทั้งหมดเข้าด้วยกันแล้ว ก็สามารถทำการติดตั้งและฮาร์ดแวร์และโค้ดคำสั่ง

4.4 ติดตั้งและฮาร์ดแวร์และโค้ดคำสั่ง

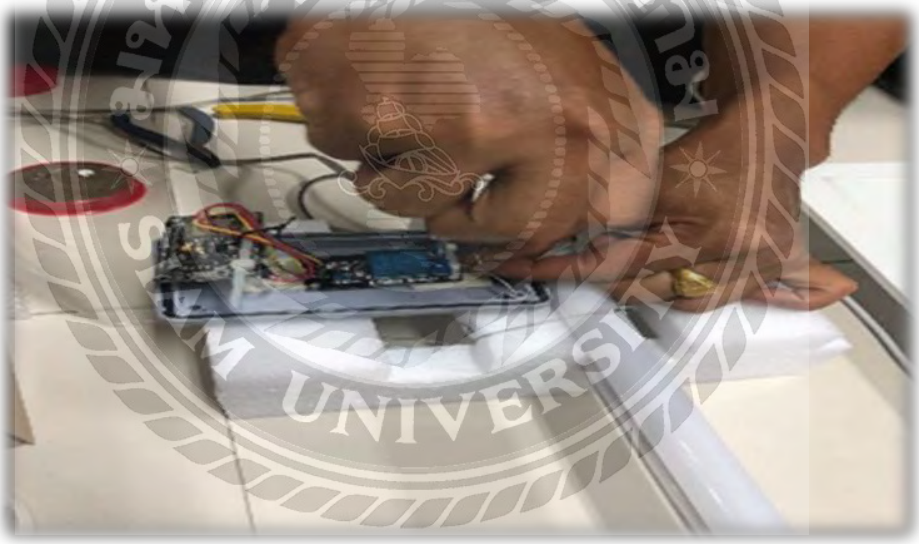
ทำการติดตั้งอุปกรณ์ IoT ติดตั้งเซนเซอร์และอุปกรณ์ควบคุมตามตำแหน่งที่กำหนดไว้ว่าเชื่อมต่ออุปกรณ์ทั้งหมดเข้ากับเครือข่าย ทดสอบการทำงานของระบบตรวจสอบการทำงานของ

อุปกรณ์แต่ละตัว ทดสอบการทำงานร่วมกันของระบบทั้งหมดปรับแต่งและแก้ไขปัญหาที่พบระหว่างการทดสอบ



รูปที่ 4.7 ทดสอบอุปกรณ์ IoT

หลังจากทำทดสอบว่าอุปกรณ์สามารถเชื่อมต่อกับสัญญาณ Wi-Fi ได้อย่างเสถียรละเริ่มทำการทดสอบการทำงานก่อนนำไปติดตั้งในจุดที่ด้กำหนดไว้



รูปที่ 4.8 ทดสอบการทำงานของระบบ

ทำการแก้ไขอุปกรณ์เนื่องจาก สายไฟที่ต่อเข้ากับอุปกรณ์ IoT มีการชำรุดจึงทำให้อุปกรณ์เชื่อมต่อหยุดการทำงาน

รูปที่ 4.9 แก้ไขปัญหาที่พบระหว่างการทดสอบ

ตัวอย่างโค้ดโปรแกรมที่กำหนดการทำงานของอุปกรณ์ IoT

การเขียนโปรแกรมควบคุม Relay ติดตั้ง Arduino IDE และ ตั้งค่า NodeMCU ESP8266

```
#define Lamp_1 D5          // Fill-in information from your Blynk Template here
#define Lamp_2 D6
#define BLYNK_TEMPLATE_ID "TMPLjJ2HjePN"
#define BLYNK_DEVICE_NAME "Quickstart Template"
#define BLYNK_FIRMWARE_VERSION        "0.1.0"
#define BLYNK_PRINT Serial    //#define BLYNK_DEBUG
#define APP_DEBUG    // Uncomment your board, or configure a custom board in Settings.h
//#define USE_SPARKFUN_BLYNK_BOARD
#define USE_NODE_MCU_BOARD//#define USE_WITTY_CLOUD_BOARD
#include "BlynkEdgent.h"
WidgetLCD lcd_showmsg(V0);
WidgetLED led_1(V10);
WidgetLED led_2(V11);
void setup()
{
Serial.begin(115200);
lcd_showmsg.clear();
lcd_showmsg.print(3, 0, "ระบบแสงสวาง");
```

```
pinMode(Lamp_1, OUTPUT);//D5
pinMode(Lamp_2, OUTPUT);//D6
digitalWrite(Lamp_1, LOW);
digitalWrite(Lamp_2, LOW);
BlynkEdgent.begin();
delay(2000);
}
void loop()
{
BlynkEdgent.run();
delay(10);
}
//------------BUT1-------------//
BLYNK_WRITE(V1)
{
Serial.println(param.asInt());
if (param.asInt())
{
lcd_showmsg.clear();
lcd_showmsg.print(4, 0, "ระบบสงงสว่าง");
lcd_showmsg.print(2, 1, "กันสาด ขวา");
digitalWrite(Lamp_1, HIGH);// ติด
led_1.on();
}
else
{
lcd_showmsg .clear();
lcd_showmsg.print(4, 0, "ระบบสงงสว่าง");
lcd_showmsg.print(4, 1, "กันสาด ขวา");
digitalWrite(Lamp_1, LOW);//ดับ
led_1.off();
}
```

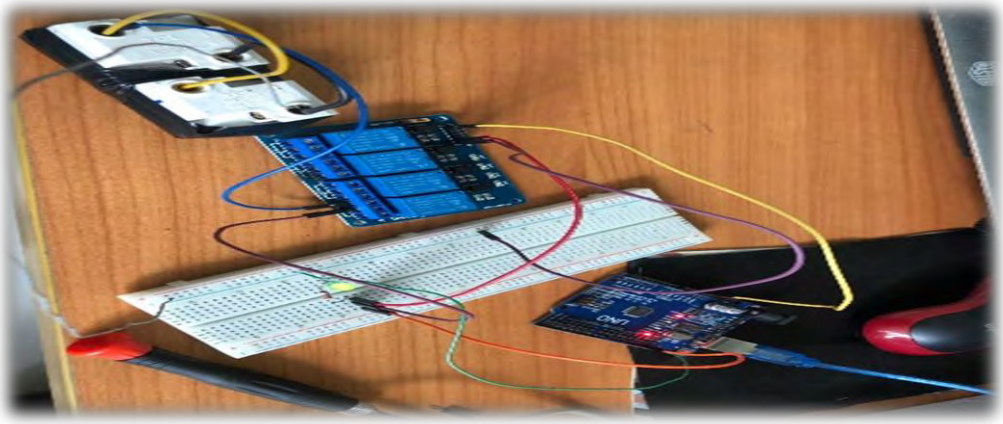
```
}
//------------BUT2-------------//
BLYNK_WRITE(V2)
{
Serial.println(param.asInt());
if (param.asInt())
{
lcd_showmsg.clear();
lcd_showmsg.print(4, 0, "ระบบสงงสว่าง");
lcd_showmsg.print(2, 1, "กันสาดซ้าย");
digitalWrite(Lamp_2, HIGH);
led_2.on();
}
else
{
lcd_showmsg.clear();
lcd_showmsg.print(4, 0, "ระบบสงงสว่าง");
lcd_showmsg.print(2, 1, "กันสาดซ้าย");
digitalWrite(Lamp_2, LOW);
led_2.off();
}
}
//------------BUT3-------------//
BLYNK_WRITE(V3)
{
Serial.println(param.asInt());
if (param.asInt())
{
lcd_showmsg.clear();
lcd_showmsg.print(4, 0, "ระบบสงงสว่าง");
lcd_showmsg.print(2, 1, "ติดทั้งหมด");
digitalWrite(Lamp_1, HIGH);
```

```
digitalWrite(Lamp_2, HIGH);
led_1.on();
led_2.on();
}
else
{
lcd_showmsg .clear();
lcd_showmsg.print(4, 0, "ระบบสงงสว่าง");
lcd_showmsg.print(2, 1, "ดับทั้งหมด");
digitalWrite(Lamp_1, LOW);
digitalWrite(Lamp_2, LOW);
led_1.off();
led_2.off ( ); } }
```

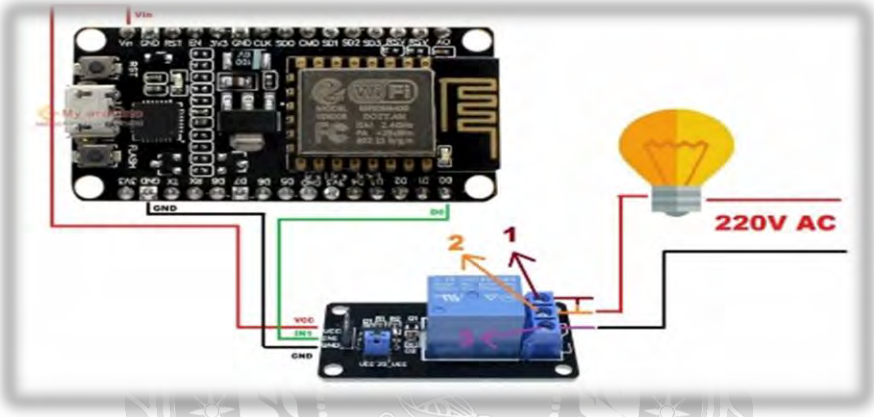
4.5 การทดสอบการทำงานของระบบควบคุม

ทำการเชื่อมต่อรีเลย์ไปยังหลอดไฟ จากนั้นสั่งโปรแกรมทำงาน แล้วตรวจสอบว่ารีเลย์ทำงานและจ่ายไฟให้กับหลอดไฟตามเงื่อนไขที่กำหนดไว้หรือไม่

รูปที่ 4.10 อุปกรณ์ไฟฟ้าที่เชื่อมต่อกับ Relay จะทำงานตาม Relay



รูปที่ 4.11 การทดสอบและตรวจสอบว่าการเชื่อมต่อของสายไฟถูกต้อง



รูปที่ 4.12 ทำการตรวจสอบว่าอุปกรณ์ IoT สามารถทำงานร่วมกับอุปกรณ์และระบบ

คำแนะนำเพิ่มเติม ความปลอดภัยเมื่อทำงานกับแรงดันไฟฟ้าสูง ควรระมัดระวังและใช้แหล่งจ่ายไฟที่ปลอดภัยการขยายโปรเจ็กต์คุณสามารถใช้ Relay หลายตัวเพื่อควบคุมอุปกรณ์หลายชิ้น และสามารถเชื่อมต่อกับอินเทอร์เน็ตเพื่อควบคุมผ่านเว็บหรือแอปพลิเคชันมือถือ

4.6 สร้างระบบรักษาความปลอดภัย

ตั้งค่าระบบรักษาความปลอดภัย ตั้งค่ารหัสผ่านและการเข้ารหัสข้อมูล ติดตั้งโปรแกรมป้องกันไวรัสและมัลแวร์ อัปเดตเฟิร์มแวร์และซอฟต์แวร์ ตรวจสอบและอัปเดตเฟิร์มแวร์และซอฟต์แวร์ของอุปกรณ์เป็นประจำ

รูปที่ 4.13 ตั้งค่าระบบรักษาความปลอดภัย

ได้ทำการตรวจสอบว่าข้อมูลที่ส่งผ่านระบบมีการเข้ารหัสและไม่สามารถถูกดักจับหรือแก้ไขได้

4.7 การบำรุงรักษา

วางแผนการบำรุงรักษาจัดตารางการตรวจสอบและบำรุงรักษาอุปกรณ์ เตรียมแผนการเปลี่ยนอะไหล่หรืออุปกรณ์เมื่อเกิดปัญหาเตรียมการแก้ไขปัญหาทางเทคนิค การดูแลระบบการปฏิบัติตามขั้นตอนเหล่านี้จะช่วยให้การออกแบบและติดตั้งระบบควบคุมไฟฟ้าในอาคารด้วย IoT เป็นไปอย่างมีประสิทธิภาพและปลอดภัย ชุดควบคุม IoT ควรมีการบำรุงรักษาอย่างน้อยเดือนละ 1 ครั้ง มีรายละเอียดดังนี้

ชุดควบคุม Relay, IoT และชุด Power Ac และ Power Dc

1. ตรวจเช็คการระบายความร้อน
2. ทำความสะอาด/ตรวจเช็คสายไฟ
3. ตรวจเช็คสายไฟฟ้าสายรอบๆ เครื่อง
4. ทำความสะอาดตู้ไฟฟ้าและอุปกรณ์ภายในตู้
5. ทดสอบการทำงานและการควบคุมของระบบหลังทำการตรวจเช็ค
6. ตรวจเช็คสภาพและทำความสะอาด บอร์ดควบคุม

บทที่ 5

สรุปผลและข้อเสนอแนะ

5.1 สรุปผลการปฏิบัติงาน

การปฏิบัติงานที่บริษัท เอสคิวดีมาร์เก็ตติ้ง จำกัด ตั้งแต่วันที่ 23 สิงหาคม พ.ศ.2564 ถึงวันที่ 10 ธันวาคม พ.ศ.2564 นั้นส่งผลให้ผู้จัดทำได้รับความรู้และประสบการณ์ต่างๆ ที่มีค่ามากมาย โดยได้ปฏิบัติตามผังแสดงข้อมูลการทำงาน ทำให้ได้ประสบการณ์และทักษะทางปฏิบัติจากการปฏิบัติสหกิจศึกษาครั้งนี้ได้บูรณาการความรู้ที่ได้จากห้องเรียนไปใช้ในการปฏิบัติงานจริงซึ่งเป็นประโยชน์ในการปฏิบัติงานในอนาคตและเป็นประโยชน์ต่อองค์กรรวมถึงผู้ปฏิบัติงาน

5.2 ประโยชน์ด้านสังคม

5.2.1 ได้เรียนรู้ระบบการบริหารองค์กร

5.2.2 ได้เรียนรู้การประสานงานกับเพื่อนร่วมงาน

5.2.4 ได้เรียนรู้หน้าที่ของแต่ละแผนก

5.2.5 ได้เรียนรู้การทำงานเป็นทีม

5.3 ประโยชน์ด้านการทำงาน

5.3.1 ได้ประสบการณ์ใหม่ ที่แตกต่างจากห้องเรียน

5.3.2 ได้สัมผัสการทำงานจริง และวิเคราะห์แก้ปัญหา

5.3.3 ได้รู้จักขั้นตอนการทำงานของระบบควบคุมแสงสว่างภายในโรงงานด้วย(IoT)

5.3.4 ได้รู้จักวิธีการบำรุงรักษาระบบแสงสว่าง

5.4 ปัญหาในการปฏิบัติงาน

5.4.1 ทักษะในการเขียนโค๊ดยังไม่เพียงพอ ทำให้ต้องใช้เวลามากในการเขียน

5.4.2 ยังขาดทักษะในการแก้ปัญหาเฉพาะหน้า ซึ่งต้องฝึกฝนให้มากขึ้น

5.5 การแก้ไขปัญหาในการปฏิบัติงาน

5.5.1 ปรึกษาพี่เลี้ยงหรือผู้ที่มีความรู้ด้านการเขียนโค๊ดเพื่อควบคุมอุปกรณ์ IoT

5.5.2 เรียนรู้การแก้ปัญหาเฉพาะหน้าจากประสบการณ์การทำงานในแต่ละวัน

5.6 ข้อเสนอแนะในการปฏิบัติงาน

5.4.1 ควรมีการขยายผลการใช้ระบบ IoT ไปเครื่องจักรอื่นๆ

5.4.2 ควรมีระบบการ Training ให้กับนักศึกษาสหกิจศึกษาในเรื่องที่จะต้องปฏิบัติงานจริง เพื่อให้สามารถปฏิบัติงานได้อย่างมีประสิทธิภาพ

บรรณานุกรม

ชไนเดอร์ อิเล็คทริค. (2566). *Busway I-Line.* https://www.se.com/th/th/download/document/SBGI17V2/

ชไนเดอร์ อิเล็คทริค. (2566). *RM6 – Medium Voltage Distribution***.** https://www.se.com/th/th/search/?q=RM6+-+MediumIoT+Voltage+Distribution.& submit +search +query =Search

นายช่างมาแชร์. (2562) *IoT (Internet Of Things) – ในโรงงานอุตสาหกรรม*.https://naichangmashare. com/2020/11/27/iot-industry

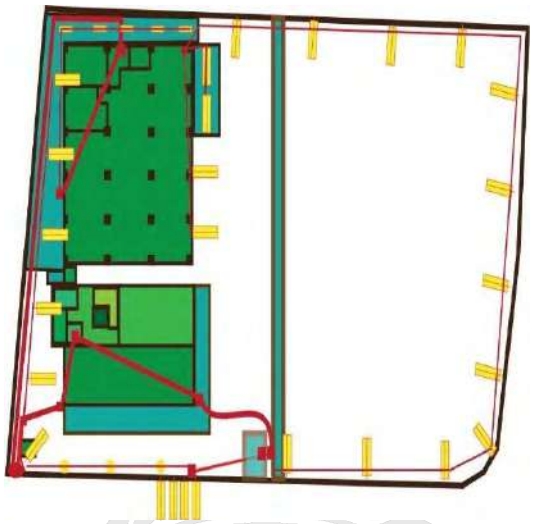
โรบอทสยาม อุปกรณ์หุ่นยนต์ Arduino. (2565*). มินิโปรเจค Arduino***.** https://www.robotsiam. com/article/37nodemcu-relay

Rakesh Ranjan Jena. (2018). *Interface Node MCU ESP 8 2 6 6 with LUA Programming Language*. http://www.rrjprince.com/2018/07/interfacing-node-muc-esp-8266-with-lua.html

ภาคผนวก

ภาคผนวก ก

ผังโครงสร้างของระบบแสงสว่างภายในและภายนอกอาคาร



รูปที่ 1 โครงสร้าง

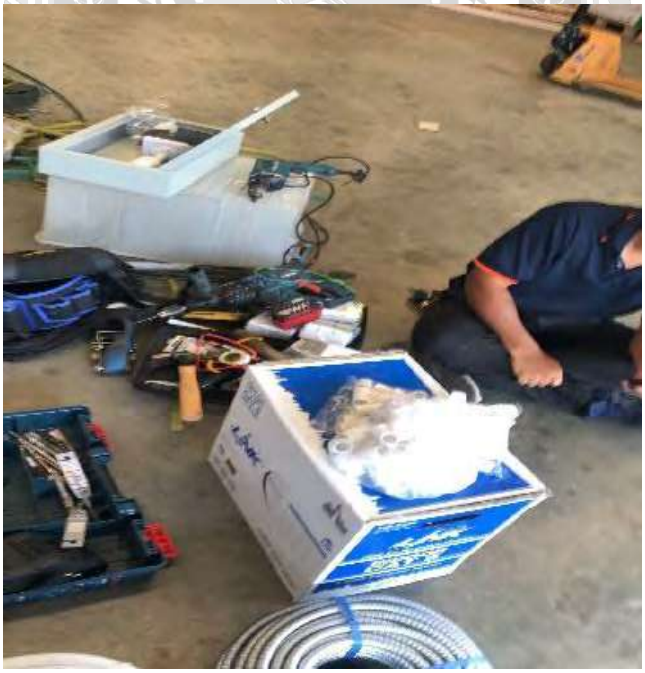
การติดตั้งระบบควบคุมแสงสว่าง

รูปที่ 2 การติดตั้งระบบควบคุมแสงสว่าง

การต่อวงจร Iot และวงจรแสงสว่าง

รูปที่ **3** การต่อวงจร Iot และวงจรแสงสว่าง

การต่อวงจร IoT และกล้องวงจรปิดภายในอาคาร



รูปที่ 4 การต่อวงจร IoT และกล้องวงจรปิดภายในอาคาร

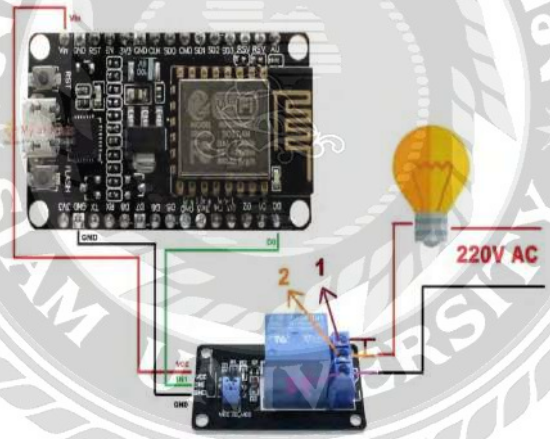
การตรวจเช็คความเร็วของ Internet

รูปที่ 5 Test Speed Internet

รูปที่ 6 Test Speed Internet

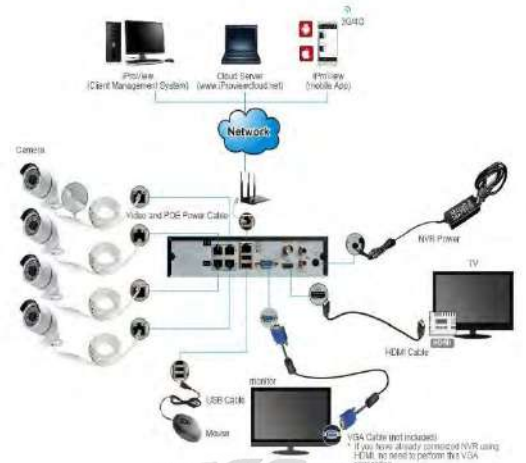
โมดูเลชัน (Modulation)

รูปที่ 6 การเชื่อมต่อของ Router และอุปกรณ์ต่างๆแบบไร้สาย



รูปที่ 7 การเชื่อมต่อของระบบ IoT กับระบบแสงสว่าง

สถานีฐาน (Base Station)



รูปที่ 8 การเชื่อมต่อของอุปกรณ์ทั้งหมด

กล้องวงจรปิดไร้สาย

รูปที่ 9 เส้นทางการเชื่อมต่อของอุปกรณ์ IoT และการมอนิเตอร์

สายกล้อง CCTV และข้อมูล (Data) ข่าวสารต่างๆ ตลอดจนสัญญาณภาพจาก**<u>กล้องวงจรปิด</u>** สามารถถูกส่งแบบไร้สายหรือที่เรียกว่า กล้องวงจรปิดไร้สาย โดยเปลี่ยนจากสัญญาณวิทยุและสัญญาณโทรทัศน์เป็น

เสียงและภาพ หรือเป็นข้อมูลด้านคอมพิวเตอร์ ข้อมูลจะถูกส่งโดยผสมไปกับคลื่นวิทยุ ซึ่งคลื่นวิทยุเป็นแค่ส่วนหนึ่งในสเปกตรัมคลื่นแม่เหล็กไฟฟ้าในช่วงที่เรียกว่า ความถี่วิทยุ (Radio frequency) ข้อมูลทุกชนิดสามารถถูกส่งโดยการใช้ความถี่วิทยุ ความถี่วิทยุมีมากมายไม่ใช่มีแค่คลื่นวิทยุเอเอ็มหรือเอฟเอ็มที่เรารู้จักเท่านั้น

โมดูเลชัน (Modulation) ข้อมูลจะถูกผสมไปกับคลื่นความถี่วิทยุโดยกระบวนที่เรียกว่า โมดุเลชั่น เมื่อได้รับคลื่นสัญญาณจะผ่านกระบวนการ ดีโมดูเลชัน (Demodulation) เพื่อแยกเอาข้อมูลออกมา

เซลล์ (Cells) โทรศัพท์เซลลูลาร์ หรือโทรศัพท์แบบพกพาหรือโทรศัพท์มือถือที่เรารู้จัก มีแนวความคิดมาจากเซลล์นี่เอง ซึ่งเป็นการแบ่งพื้นที่ออกเป็นส่วนย่อยๆ เมื่อโทรศัพท์เซลลูลาร์รับ-ส่งสัญญาณจะมีการติดต่อสื่อสารภายในเซลล์นั้นๆ เองเท่านั้น ซึ่งระยะทางการสื่อสารก็จะไม่ไกลมาก ข่าวสารต่างๆจะถูกส่งภายในเซลล์นั้นไปที่เป้าหมายภายในเซลล์นั้น

สถานีฐาน (Base Station) ภายในแต่ละเซลล์จะมีสถานีฐานซึ่งจะส่งและรับสัญญาณสื่อสารทั้งรับและส่งไปยังโทรศัพท์มือถือ ภายในเซลล์นั้นๆ

อุปกรณ์ส่งสัญญาณละอุปกรณ์รับสัญญาณ คลื่นความถี่วิทยุพร้อมด้วยข้อมูลข่าวสารจะถูกส่งโดยอุปกรณ์ส่งสัญญาณ (Transmitters) และรับโดยอุปกรณ์ที่เรียกว่าอุปกรณ์รับสัญญาณ (Receivers)

แฮนด์ออฟ (Handoff) เมื่อต้องมีเปลี่ยนสถานีฐานจากหนึ่งไปอีกฐานหนึ่งแฮนด์ออฟจะเป็นตัวช่วยให้การสื่อสารของ*กล้องวงจรปิด* **CCTV** ได้อย่างต่อเนื่องไม่ขาดตอน

เทคโนโลยีการสื่อสารไร้สายในชีวิตประจำวันของเรา ยกตัวอย่างเช่น

โทรศัพท์มือถือ(Cell Phone) นี่คืออุปกรณ์ที่ทุกๆคนนึกถึงเมื่อกล่าวถึงเทคโนโลยีแบบไร้สาย

ระบบเครือข่ายแบบไร้สาย(Wireless Network) บ้านหลายๆบ้านทุกวันนี้ไม่ใช่มีแค่คอมพิวเตอร์เครื่องเดียว ระบบเครือข่ายโดยเทคโนโลยีแบบไร้สายสามารถเชื่อมตอกันได้ และสามารถที่จะใช้อินเทอร์เน็ตความเร็วสูงผ่านทางเคเบิลโมเด็มได้ด้วย

กล้องวงจรปิดไร้สาย(Wireless security camera) ตอนนี้ได้กลายเป็นนิยมมากขึ้นในยุคปัจจุบันของผู้บริโภคที่ต้องการระบบรักษาความปลอดภัยที่มีประสิทธิภาพ ซึ่งเหมาะสำหรับโรงงานขนาดใหญ่ หรือโรงงานขนาดเล็ก โรงแรม โรงเรียน ตึกอาคารต่างๆ รวมไปถึงบ้านพักอาศัย หอพัก เป็นต้น เนื่องจากไม่จำเป็นต้องใช้สายสัญญาณภาพ

สถานประกอบการ

1527/1ถ.เจริญนคร แขวงบางลำภูล่าง เขตคลองสาน กรุงเทพ 10600

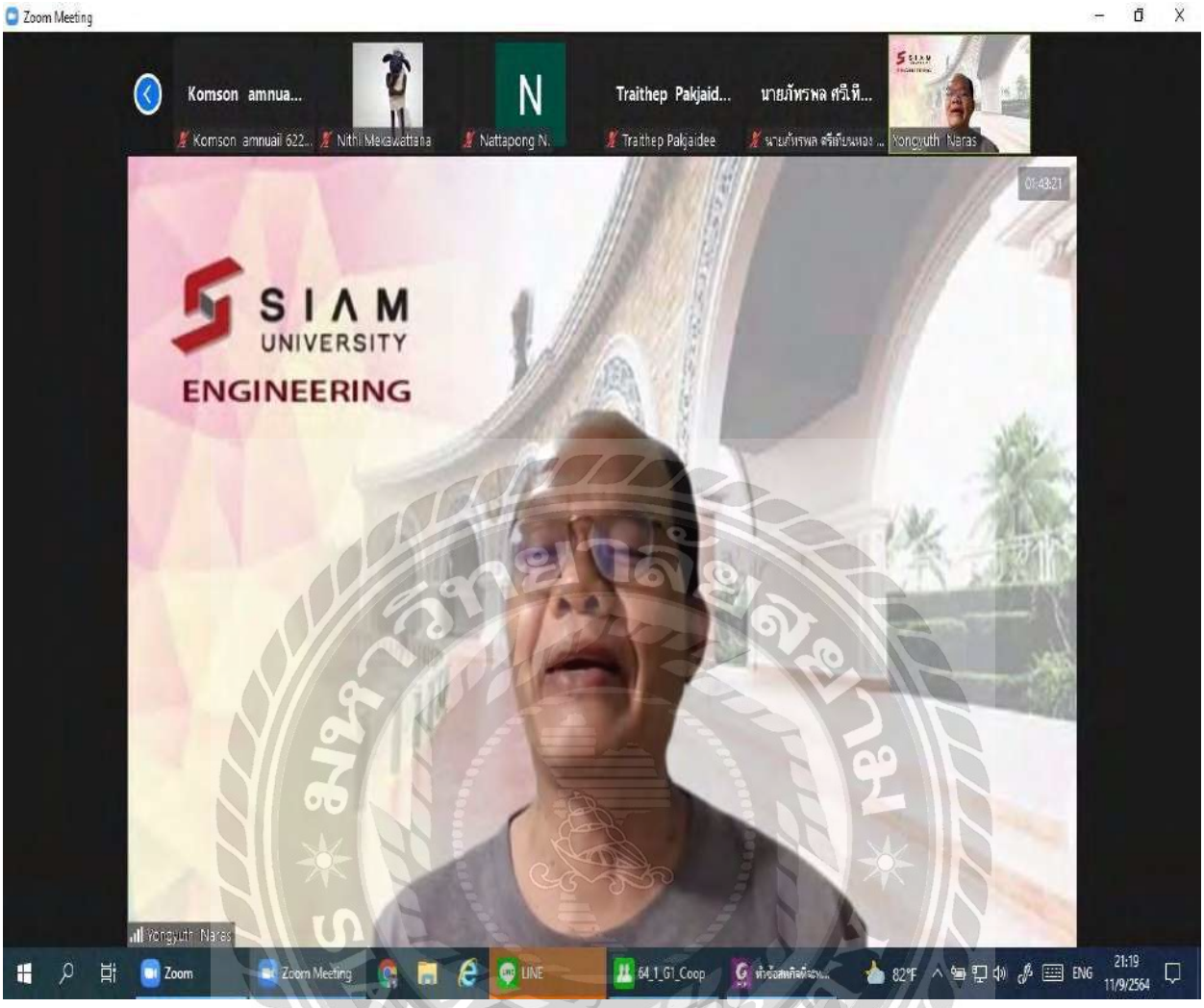
HOTLINE : 093-242-4951,02-437-0100,02-439-2790-9,02-439-2800-1

อาจารย์นิเทศสหกิจศึกษา	ผศ.ดร.ยงยุทธ	นาราษฎร์
	ผศ.วิภาวัลย์	นาคทรัพย์
	ผศ.พกิจ	สุวัตถิ์
นักศึกษาสหกิจศึกษา	นายจรัญ	รักญาติ 6223200025
คณะวิศวกรรมศาสตร์	สาขาวิชาวิศวกรรมไฟฟ้า	

นิเทศงานสหกิจศึกษา ผ่านโปรแกรม Zoom เนื่องด้วยสถานการณ์ COVID 19

ภาคผนวก ข
การนิเทศงานสหกิจศึกษา

การนิเทศงานสหกิจศึกษาผ่านโปรแกรม Zoom เนื่องด้วยสถานการณ์ COVID

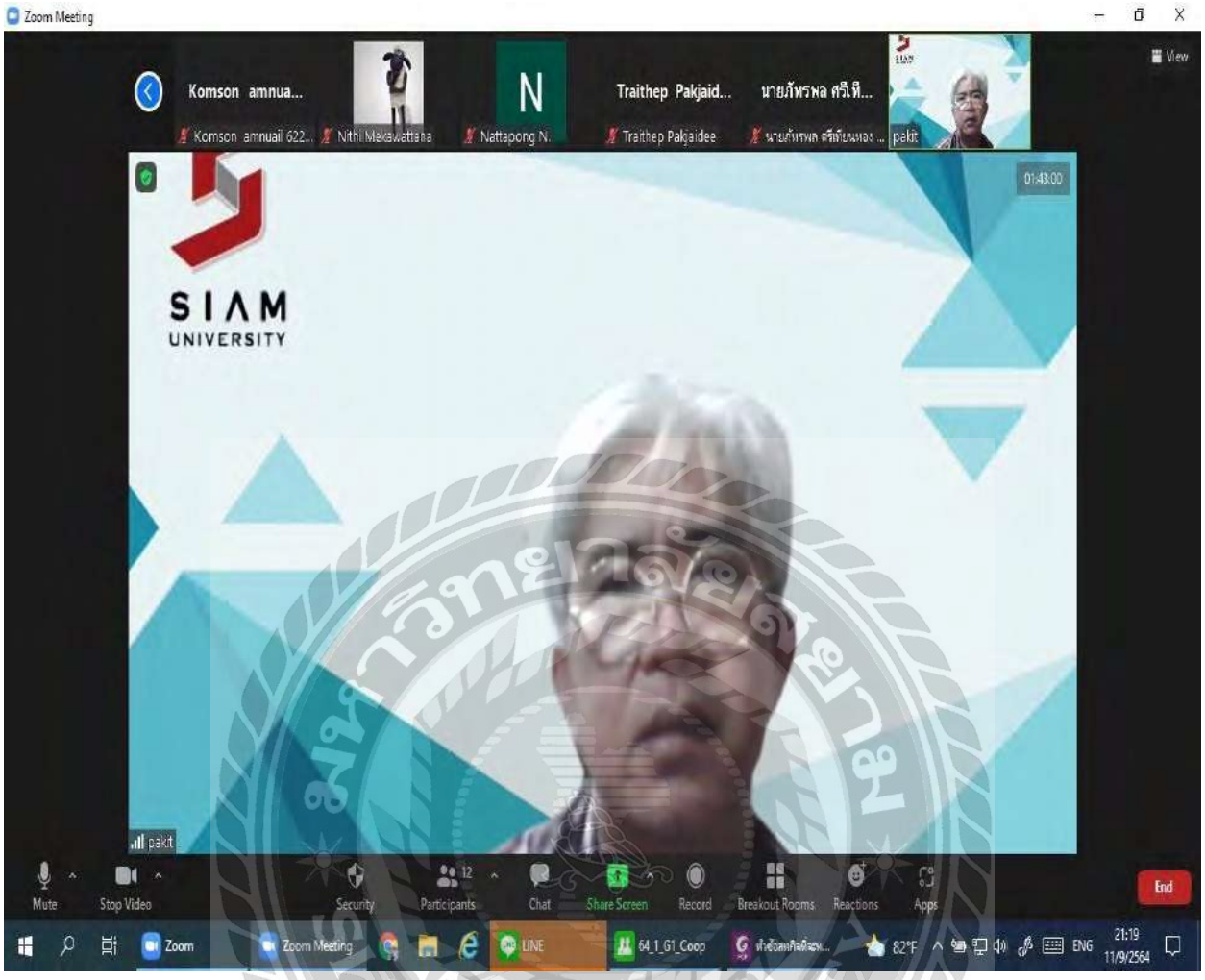


รูปที่ 1

การนิเทศงานสหกิจศึกษาผ่านโปรแกรม Zoom เนื่องด้วยสถานการณ์ COVID

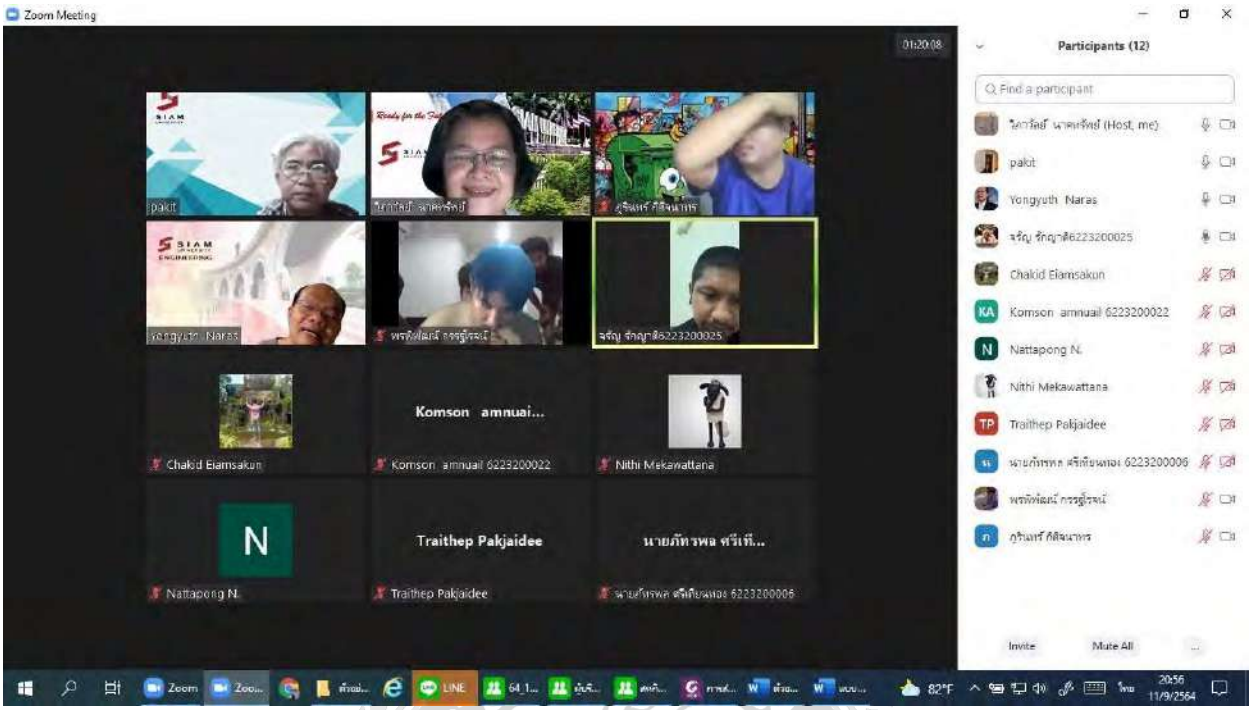
รูปที่ 2

การนิเทศงานสหกิจศึกษาผ่านโปรแกรม Zoom เนื่องด้วยสถานการณ์ COVID



รูปที่ 3

การนิเทศงานสหกิจศึกษาผ่านโปรแกรม Zoom เนื่องด้วยสถานการณ์ COVID


Zoom Meeting
Participants (12)
Find a participant
pakit
Yongyuth Naras
Chakid Elamsakun
Komson amnuail 6223200022
Nattapong N.
Nithi Mekawattana
Traithep Pakjaidee
Invite
Mute All
Komson amnuai...
Traithep Pakjaidee
Chakid Elamsakun
Komson amnuail 6223200022
Nithi Mekawattana
Nattapong N.
Traithep Pakjaidee

รูปที่ 4

ภาคผนวก ค

การสอบโครงการสหกิจศึกษา

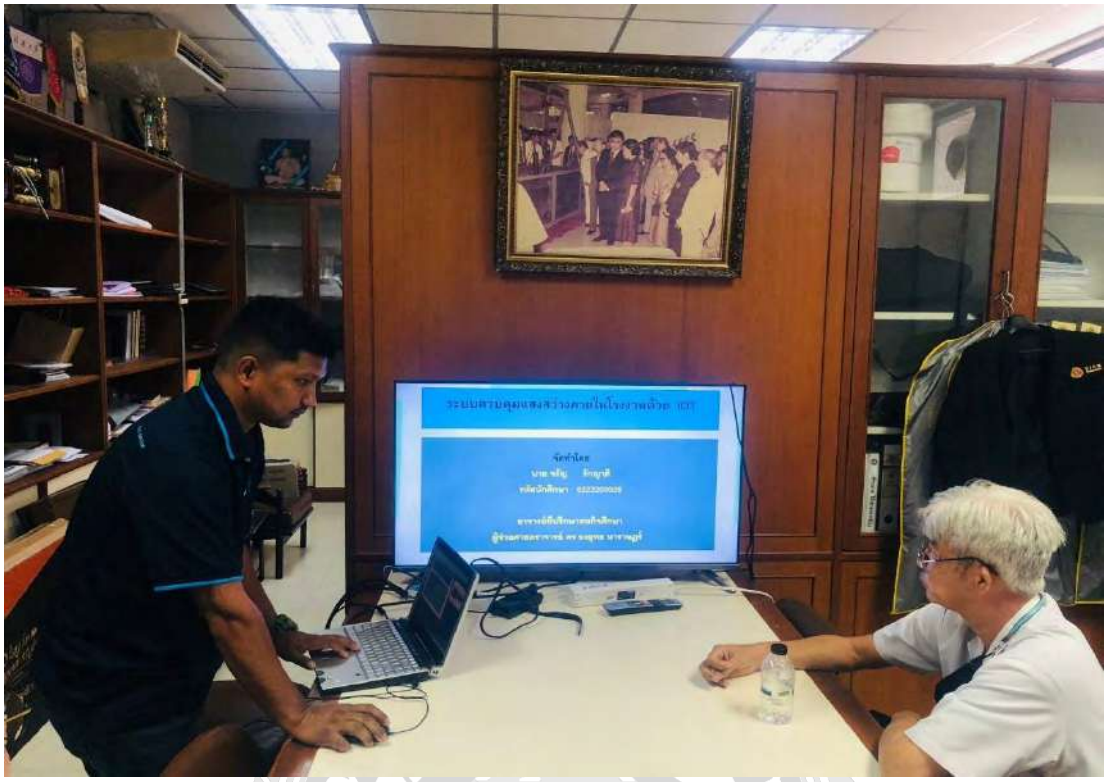
การสอบโครงการสหกิจศึกษา

รูปที่ 1

การสอบโครงการสหกิจศึกษา

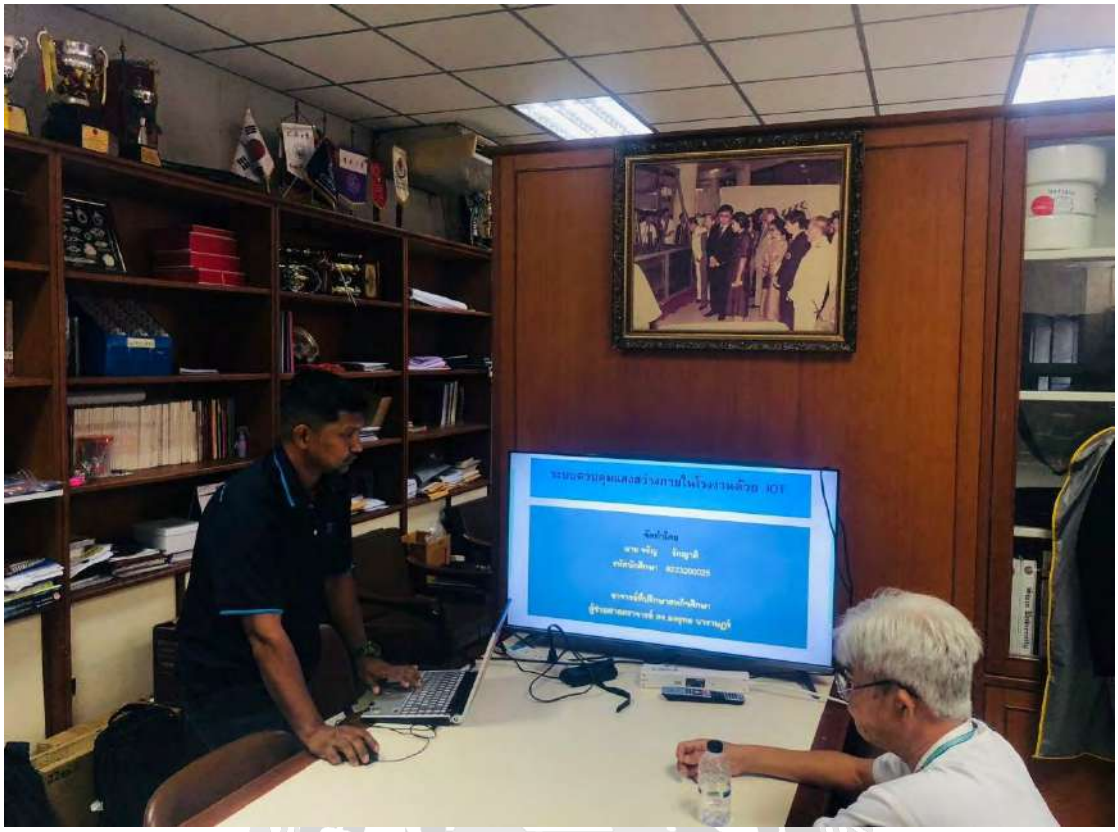
รูปที่ 2

การสอบโครงการสหกิจศึกษา



รูปที่ 3

การสอบโครงการสหกิจศึกษา



รูปที่ 4

ภาคผนวก ง

การตรวจสอบการลอกเลียนวรรณกรรมทางวิชาการโดยการใช้โปรแกรมอักขราวิสุทธิ์

การตรวจสอบการลอกเลียนวรรณกรรมทางวิชาการโดยการใช้โปรแกรมอักขราวิสุทธิ์

Plagiarism Checking Report

Created on 2024-05-23 17:07:05 at 17:07 PM

Submission Information

ID	SUBMISSION DATE	SUBMITTED BY	ORGANIZATION	FILENAME	STATUS	SIMILARITY INDEX
3754633	May 23, 2024 at 17:04 PM	wipavan.nar@siam.edu	มหาวิทยาลัยสยาม	จรัญ6223200025.pdf	Completed	1.39%

Match Overview

NO.	TITLE	AUTHOR(S)	SOURCE
1	http://huso.pn.psu.ac.th/coop/Data/Coop-Form/Student/08_Report.pdf	huso.pn.psu.ac.th	huso.pn.psu.ac.th_nutch
2	ระบบบ้านอัจฉริยะควบคุมด้วยเทคโนโลยีเครือข่ายไร้สาย เซ็นเซอร์ และแอนดรอยด์ แอปพลิเคชันภายใต้แนวคิดอินเทอร์เน็ตเพื่อทุกสิ่ง,Smart home system controlling by wireless network technology sensors and android application with the concept of internet of things.	ศิริวรรณ เอี่ยมบัณฑิต	มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
3	http://dusithost.dusit.ac.th/~juthawut_cha/download/Ch10_MAL2015.pdf	dusithost.dusit.ac.th	dusithost.dusit.ac.th_nutch
4	http://coopcenter.lpru.ac.th/web/public/uploads/pdf/PDF_2018-10-22_0022.pdf	coopcenter.lpru.ac.th	coopcenter.lpru.ac.th_nutch

รูปที่ 1

ประวัติผู้จัดทำ

ชื่อ-นามสกุล		นายจรัญ รักญาติ
รหัสนักศึกษา		6223200025
วัน/เดือน/ปีเกิด		4 มีนาคม 2530
ที่อยู่ปัจจุบัน		57/427 อาคารดีไอวี่ริเวอร์ คอนโดฯ ถ.ราษฎร์บูรณะ เขตราษฎร์บูรณะ แขวงบางปะกอก กรุงเทพฯ 10140
E-mail :		david.may.cay@ gmail.com
ประวัติการศึกษา	ปวส	วิทยาลัยเทคโนโลยีสยาม (สยามเทค)
	สาขา	ช่างไฟฟ้าและอิเล็กทรอนิกส์
	ปริญญาตรี	มหาวิทยาลัยสยาม
	สาขา	วิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์

ช่องทางการติดต่อ

FACEBOOK	:	Runways Inrunways
LINE	:	farcfar
YOUTUBE	:	FarZee Ch1
TIKTOK	:	myways03
TEL	:	063-874-5661, 081-890-4190, 099619-4799