ระบบตู้จ่ายน้ำมันบริการตนเอง
SELF-SERVICE GASOLINE STATION

โดย
นายยสินทร ธนบุญสมบัติ
นายวีรภัทร บัวชื่น

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมโทรคมนาคม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2560

ระบบตู้จ่ายน้ำมันบริการตนเอง
SELF-SERVICE GASOLINE STATION

โดย

นายยสินทร ธนบุญสมบัติ 57011036

นายวีรภัทร บัวชื่น 57011200

อาจารย์ที่ปรึกษา

ผศ.ดร.กฤษณ์ วงรุจิระ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2560

ผ่านการตรวจรูปเล่มแล้ว

(.....)

อาจารย์ที่ปรึกษา

17/05/61

KMITL วิศวกรรมโทรคมนาคม Telecommunications Engineering

ผ่านการตรวจชิ้นงานแล้ว

(.....)

กรรมการผู้ตรวจชิ้นงาน

17/05/61

KMITL วิศวกรรมโทรคมนาคม Telecommunications Engineering

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2560

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบตู้จ่ายน้ำมันบริการตนเอง

SELF-SERVICE GASOLINE STATION

ผู้จัดทำ

1. นายยสินทร ธนบุญสมบัติ 57011036
2. นายวีรภัทร บัวชื่น 57011200

..... **อาจารย์ที่ปรึกษา**

(ผศ.ดร.กฤษณ์ วงรุจิระ)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

โครงงานระบบตู้จ่ายน้ำมันบริการตนเอง (SELF-SERVICE GASOLINE STATION) นี้มีความก้าวหน้าตามแผนที่วางไว้ เนื่องจากได้รับความอนุเคราะห์อย่างดียิ่งจาก ผศ.ดร. กฤษณ์ วงรุจิระ อาจารย์ที่ปรึกษาที่ได้ให้คำแนะนำ ความรู้ และความเข้าใจตลอดระยะเวลาการดำเนินงาน ขอขอบพระคุณท่านในความหวังดีและห่วงใยที่มีให้แก่ผู้จัดทำเป็นอย่างยิ่ง

ขอขอบพระคุณคณาอาจารย์ประจำสาขาวิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุกท่าน ที่ได้อบรมสั่งสอน และให้ความรู้แก่ผู้จัดทำ

ขอขอบพระคุณผู้มีส่วนเกี่ยวข้องทุกท่านที่คอยช่วยเหลือ ให้คำแนะนำ และให้กำลังใจแก่ผู้จัดทำเสมอมา จนกระทั่งโครงงานดำเนินมาถึงเป้าหมายที่วางไว้ หากมีข้อบกพร่องประการใด ผู้จัดทำยินดีขอน้อมรับคำติชมจากทุกท่านที่ได้เข้ามาศึกษา เพื่อเป็นประโยชน์ในการพัฒนาต่อไป

นายยสินทร ธนบุญสมบัติ
นายวีรภัทร บัวชื่น
ผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบตู้จ่ายน้ำมันบริการตนเอง
SELF SERVICE GASOLINE STATION

โดย นายยสินทร ธนบุญสมบัติ 57011036
นายวีรภัทร บัวชื่น 57011200

อาจารย์ที่ปรึกษา ผศ.ดร.กฤษณ์ วงรุจิระ

บทคัดย่อ

ปริญญานิพนธ์นี้เป็นการออกแบบและสร้างระบบตู้จ่ายน้ำมันบริการตนเองแบบหยอดเหรียญ โดยในปัจจุบันยังไม่มีปั๊มน้ำมันที่บริการตนเองอย่างเป็นทางการ ผู้จัดทำจึงมีความสนใจโดยมีกลุ่มเป้าหมายเป็นบุคลากรในสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ให้เป็นสวัสดิการของบุคลากร และส่งเสริมการพัฒนาหน้าต่างแสดงผล โครงงานมีส่วนประกอบดังนี้ ส่วนติดต่อกับลูกค้าผ่านจอแสดงผลแบบสัมผัส โดยโปรแกรมคิวต์ใช้ในการออกแบบและโต้ตอบกับผู้ใช้ เพื่อสั่งไมโครคอนโทรลเลอร์และราสเบอร์รี่พาย ด้วยจอแสดงผลแบบสัมผัสผู้ใช้จะสามารถเลือกปริมาณน้ำมันที่จะเติมได้ตามต้องการ จากนั้นทำการหยอดเหรียญตามจำนวน โดยที่เครื่องมีเก็บข้อมูลเพื่อแยกระหว่างบุคลากรภายในและภายนอกสถาบันด้วยอุปกรณ์สื่อสารไร้สายระยะสั้นและบัตรประจำตัวบุคลากร(บัตรนักศึกษา) ต่อมาเป็นส่วนไมโครคอนโทรลเลอร์รับข้อมูลจากจอแสดงผลสัมผัสแล้วทำการส่งไปควบคุมอุปกรณ์จ่ายน้ำมันตามจำนวนที่ต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ABSTRACT

This project is about designing and creating a kiosk of self-service gasoline station. Currently, there are less self-service gas stations. To fill this requirement in a market, we create a kiosk with touch screen panel to communicate with a user. The QT-program is used to design and interact with a user, micro-controller and raspberry pi. With touch screen panel, user will be able to select the amount of gasoline to fill in his/her tank. After select a menu, the bank note receiver and coin accepter are used to collect the amount of money that buyer put money in. Then, the pump will be started to fill in customer tank and stop automatically after a counting flow meter met amount of buyer token. Other payments method with NFC tag and reader are also purpose in this project.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันตู้เติมน้ำมันบริการตนเองแบบหยอดเหรียญถูกนำมาใช้มากขึ้นในหลายพื้นที่ แต่ในพื้นที่สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังยังไม่มีตู้เติมน้ำมันบริการตนเองที่สะดวกในการใช้บริการของนักศึกษา ผู้จัดทำจึงได้ออกแบบและสร้างระบบตู้เติมน้ำมันบริการตนเอง(SELF-SERVICE GASOLINE STATION) ขึ้นมาโดยมีพื้นฐานมาจากตู้แบบหยอดเหรียญ แล้วพัฒนาเพิ่มระบบการยืนยันตัวตนด้วยบัตร โดยผู้ใช้บริการสามารถควบคุมได้ที่จอแสดงผล เพื่อเพิ่มความสะดวกยิ่งขึ้น

1.2 วัตถุประสงค์

1) ออกแบบและสร้างระบบตู้จ่ายน้ำมันบริการตนเอง
2) เพื่อศึกษาการใช้งาน RaspberryPi และการออกแบบ Graphic User Interface
3) พัฒนาระบบจอมอนิเตอร์ควบคุมการสั่งการ
4) สามารถควบคุมการจ่ายน้ำมันและคำนวณได้
5) ส่งเสริมระบบการบริการตนเอง เพื่อลดต้นทุนงานบริการ

1.3 ขอบเขตของปริญญานิพนธ์

1) ทดลองระบบการทำงานด้วยน้ำและลม
2) สามารับคำสั่งจากผู้ใช้งานผ่านทางจอแสดงผลแบบสัมผัสได้
3) ปรับปรุงราคาผ่านจอแสดงผลได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2

ทฤษฎีและหลักการที่เกี่ยวข้อง

ในบทนี้กล่าวถึงคุณลักษณะ หลักการ วิธีใช้งาน และคุณสมบัติต่างๆของอุปกรณ์ในปริญญานิพนธ์

2.1 บอร์ด Arduino Mega 2560 R3

บอร์ด Arduino Mega 2560 R3 เป็นบอร์ดไมโครคอนโทรลเลอร์ตระกูล AVR บอร์ด Arduino Mega 2560 R3 ถูกออกแบบมาสำหรับงานที่ใช้สื่อสารรับและส่งข้อมูล ผ่านทางอินพุต เอาต์พุตหลายๆ อุปกรณ์ ทั้งนี้บอร์ด Arduino Mega 2560 R3 ยังมีหน่วยความจำแบบแฟลชมาก ทำให้สามารถเขียนรองรับโค้ดโปรแกรมมากกว่า มีอินพุตแบบแอนะล็อก และดิจิทัล มีความจำ แฟลช 256 KB ไมโครคอนโทรเลอร์ที่ใช้คือ ATmega2560 Clock 16 เมกะเฮิร์ท ระดับแรงดันทำงานของพอร์ต 5 โวลต์

รูปที่ 2.1 Arduino Mega 2560 R3 [1]

2.1.1 คุณสมบัติทางเทคนิคของบอร์ด Arduino Mega 2560 R3

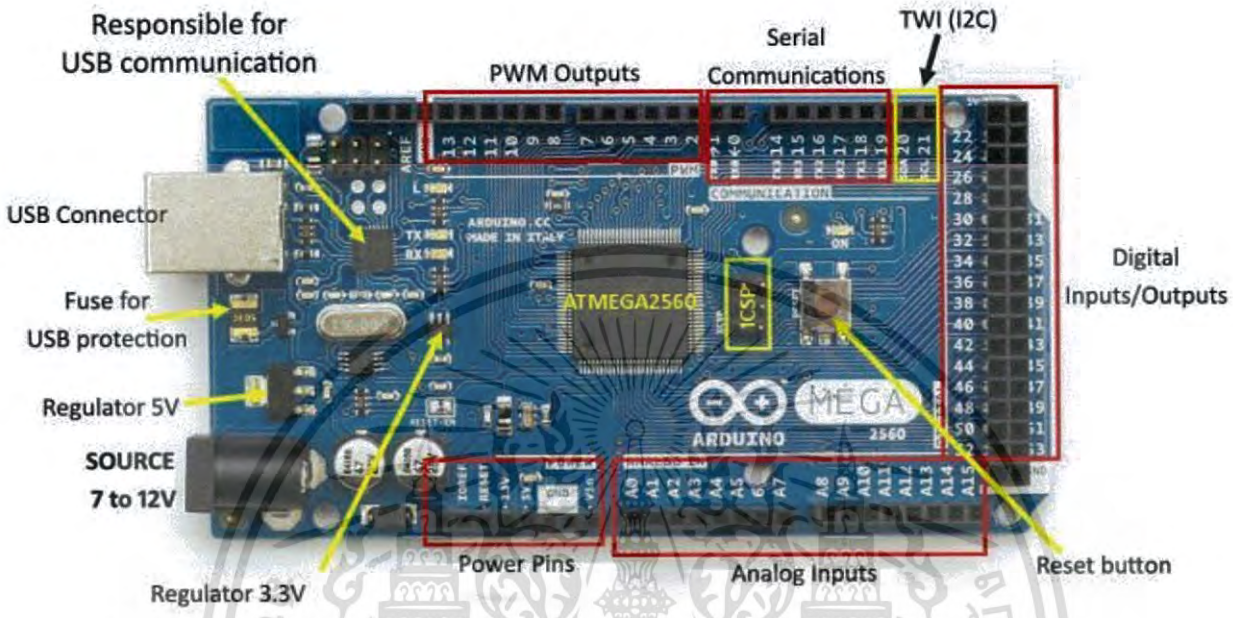
2.1.1.1.มีระบบควบคุมกลาง ATmega2560 และหน่วยความจำในตัวเดียวกัน

2.1.1.2.มีหน่วยความจำ SRAM 8 KB และ EEPROM 4 KB

2.1.1.3.มีอินพุตแบบแอนะล็อคและดิจิทัล 16 และ 54 ช่องตามลำดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.2 ส่วนประกอบของบอร์ด Arduino Mega 2560 R3



รูปที่ 2.2 Arduino Mega 2560 Rev3 [1]

2.1.2.1 USB Connector (ใช้เชื่อมต่อบอร์ดกับคอมพิวเตอร์ ในการอัพโหลดโปรแกรมและแสดงผล)

2.1.2.2 Source 7 to 12V DC (แหล่งจ่ายภายนอกแรงดัน 7 – 12V)

2.1.2.3 ICSP (In circuit serial programing ใช้เชื่อมต่อแบบ SPI)

2.1.2.4 Serial communication (ใช้เชื่อมต่อแบบ UART มีทั้งหมด 4 พอร์ต)

2.1.2.5 Power pin (เป็นแหล่งจ่ายไฟขาออก)

2.1.2.6 Analog Input (ใช้รับ/ส่งสัญญาณแอนะล็อก)

2.1.2.7 Digital Inputs/Outputs (พอร์ตรับ/ส่ง ข้อมูลแบบดิจิทัล)

2.1.2.8 Reset button (ปุ่มรีเซ็ตวงจร)

2.1.8.9 ATMEGA2560 (ชิปไอซีไมโครคอนโทรลเลอร์)

2.1.2.10 PWM Outputs (ใช้ควบคุมและเขียนค่าแอนะล็อกด้วยพอร์ตดิจิทัล)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.3 การเริ่มต้นใช้งานบอร์ด Arduino Mega2560 Rev3

2.1.3.1 ดาวน์โหลดและติดตั้งซอฟท์แวร์ Arduino IDE

2.1.3.2 เขียนโปรแกรมที่ต้องการบนซอฟท์แวร์ Arduino IDE

2.1.3.3 ประมวลผลและอัพโหลดโปรแกรมลงบอร์ดผ่านทาง USB port

2.1.3.4 แสดงผลโดย Serial Monitor บนหน้าต่างซอฟต์แวร์ Arduino IDE

2.2 Raspberry Pi

Raspberry Pi คือ บอร์ดคอมพิวเตอร์ขนาดเล็ก ที่ถูกพัฒนาขึ้นโดย Raspberry Pi Foundation มีคุณสมบัติเด่น คือ ติดต่อ และ ควบคุมอุปกรณ์อิเล็กทรอนิกส์ได้ มีความสามารถในการใช้งานทั่วไป เช่น ทำงานเอกสาร ดูหนัง ฟังเพลง เชื่อมต่ออินเทอร์เน็ตผ่าน Web Browser หรือทำ Web Server

รูปที่ 2.3 Raspberry Pi [2]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.1 คุณสมบัติทางเทคนิคของบอร์ด Raspberry Pi

2.2.1.1 หน่วยประมวลกลาง (CPU) คือ 64-bit quad-core ARM Cortex-A53 1.2 GHz Broadcom BCM2837

2.2.1.2 1 GB LPDDR2 memory

2.2.1.3 หน่วยประมวลผลกราฟฟิค (GPU) คือ Dual core Videocore IV® Multimedia co-processor

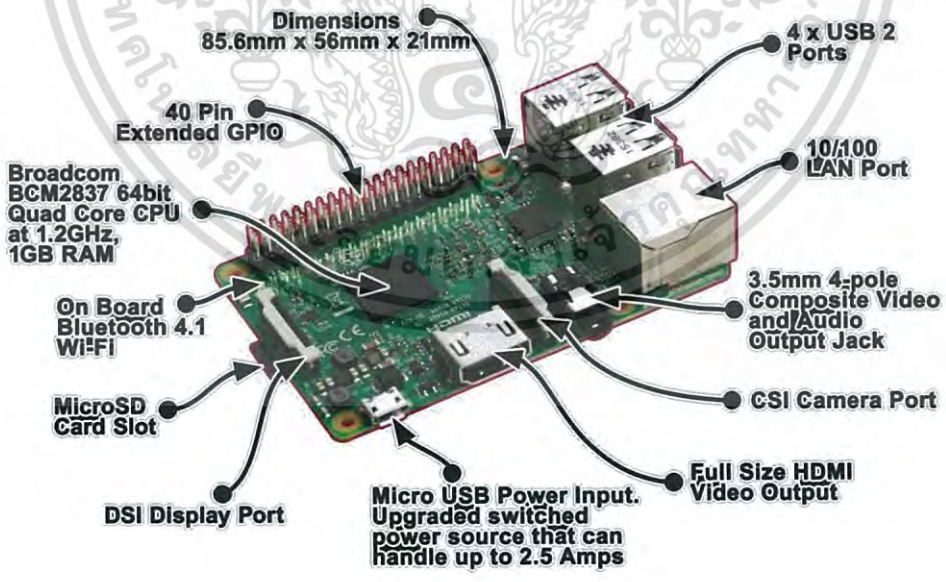
2.2.1.4 พอร์ตเชื่อมต่อ USB 2.0 จำนวน 4 พอร์ต, HDMI 1 พอร์ต, Ethernet 1 พอร์ต, GPIO 40 พิน, ช่อง Micro SDcard ที่ติดตั้งระบบปฏิบัติการไว้

2.2.1.5 รองรับ ARM GNU/Linux distributions และ Windows 10 IoT เวอร์ชั่นล่าสุด.

2.2.1.6 Power supply ขนาด 2.5 A

2.2.1.7 ขนาดบอร์ด 85 x 56 x 17 mm

2.2.2 ส่วนประกอบของบอร์ด Raspberry Pi



รูปที่ 2.4 ส่วนประกอบของบอร์ด Raspberry Pi 3 model b [2]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.2.1 40 Pins Extended GPIO (ใช้รับ/ส่งข้อมูลแบบดิจิทัล)

2.2.2.2 Bluetooth 4.1 Wi-Fi (ใช้เชื่อมต่อแบบ Bluetooth และ Wi-Fi)

2.2.2.3 Broadcom BCM2837 64 bit Quad Core CPU 1.2 GHz (หน่วยประมวลผลกลาง)

2.2.2.4 Micro SDcard slot (ช่องเสียบ SDcard)

2.2.2.5 Full size HDMI Video Output (ช่องเอาท์พุตภาพและเสียง)

2.2.2.6 CSI Camera Port (ใช้เชื่อมต่อกับกล้อง)

2.2.2.7 Audio Output Jack (เอาท์พุตเสียง)

2.2.2.8 LAN Port (ใช้เชื่อมต่ออินเทอร์เน็ตแบบ LAN)

2.2.2.9 USB Port (ใช้เชื่อมต่อกับอุปกรณ์อื่นๆ)

2.2.3 การเริ่มต้นใช้งาน Raspberry Pi 3

2.2.3.1 ติดตั้งระบบปฏิบัติการลง Micro SD card โดยดาวน์โหลดไฟล์รูปภาพจาก http://www.raspberrypi.org/downloads/

2.2.3.2 ใส่ SD card เข้าที่ช่องใส่ Raspberry Pi

2.2.3.3 เชื่อมต่ออุปกรณ์ต่างๆเข้ากับ Raspberry Pi เช่น เม้าส์ คีย์บอร์ด เป็นต้น

2.2.3.4 ทำการตั้งค่าเบื้องต้นต่างๆ เช่น เวลา แป้นพิมพ์ แล้วทำการ reboot

2.2.3.5 ทำการ update ระบบปฏิบัติเมื่อมีเวอร์ชั่นใหม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3 เครื่องหยอดและตรวจจับเหรียญ (Vending Machine)

รูป 2.5 เครื่องหยอดและตรวจจับเหรียญ

เครื่องหยอดและตรวจจับเหรียญหลายประเภทรุ่น UCAE สามารถใช้ระบุเหรียญที่หยอดลงไป โดยเหรียญแต่ละแบบจะมีชุดคำสั่ง HEX เฉพาะตัวเมื่อตรวจจับได้ ไม่ว่าจะเป็นเหรียญ 1 บาท (แบบใหม่/เก่า) เหรียญ 5 บาท รวมถึงเหรียญ 10 บาท

หลักการทำงานคือ เมื่อจ่ายไฟ 12 Vdc เข้าเครื่องหยอดเหรียญตัวหน่วยประมวลผลของเครื่องจะเริ่มทำงาน เมื่อมีการหยอดเหรียญหน่วยประมวลผลจะทำการเทียบค่าของเหรียญที่หยอดกับเหรียญตัวอย่าง ถ้าค่าตรงกันจะทำการส่งค่าไปยัง Coin signal และ Counter signal การทำงานของการสื่อสาร UCA ถูกแบ่งเป็น 3 ขั้นตอนดังนี้

2.3.1 การตรวจสอบความพร้อมของตัวรับเหรียญ (UCA)

2.3.1.1 การสั่งทำงาน ตัวรับเหรียญทำงาน (ENABLE)

ทำการส่งชุดคำสั่ง 90 05 01 03 99 ไปยังตัวรับเหรียญ จะได้รับการตอบกลับมาด้วยชุดรหัส 90 05 50 03 E8 ซึ่งหมายความว่า เครื่องรับเหรียญพร้อมทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.1.2 การสั่งหยุดทำงาน ตัวรับเหรียญหยุดทำงาน (DISABLE)

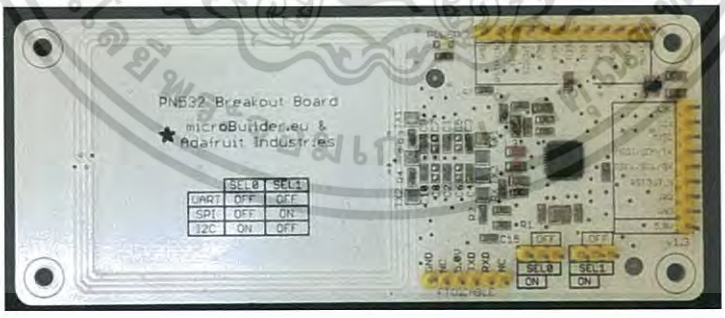
ทำการส่งชุดคำสั่ง 90 05 02 03 9A ไปยังตัวรับเหรียญ จะได้รับการตอบกลับมาด้วยชุดรหัส 90 05 50 03 E8 ซึ่งหมายความว่า เครื่องรับเหรียญหยุดทำงาน จนกว่าจะมีการส่งชุดคำสั่ง ENABLE อีกครั้ง

2.3.2 การตรวจสอบค่าของเหรียญ

โดยการตรวจสอบค่าของเหรียญ เครื่องหยอดเหรียญจะทำการส่งข้อมูลมาเป็นชุดรหัส ซึ่งเป็นค่าเฉพาะของแต่ละเหรียญ ดังนี้

1 บาทเก่า	90 06 12 01 03 AC
1 บาทใหม่	90 06 12 06 03 B1
2 บาทเก่า	90 06 12 02 03 AD
2 บาทใหม่	90 06 12 05 03 B0
5 บาท	90 06 12 03 03 AE
10 บาท	90 06 12 04 03 AF

2.4 PN532 NFC module



รูปที่ 2.6 PN532 NFC module

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

NFC (Near Field Communication) คือเทคโนโลยีที่ทำให้เราสามารถส่งผ่านข้อมูลได้ผ่านการเชื่อมต่อแบบไร้สาย ด้วยหลักการรับคำสั่งผ่านตัวนำไฟฟ้าผ่านอากาศด้วยคลื่นวิทยุ หรือที่เรียกว่า RFID (Radio-Frequency Identification) ที่จะกระตุ้นการทำงานด้วยการเข้าใกล้กัน

2.4.1 คุณสมบัติทางเทคนิคของ PN532 NFC controller

2.4.1.1 ขนาด 51mm x 117.7mm x 1.1mm

2.4.1.2 ใช้ชิปเซ็ต I2C 7-bit address 0x48

2.4.1.3 ใช้งานที่ย่านความถี่ 13.56 MHz

2.4.1.4 สามารถอ่านแท็ก RFID/NFC ได้

2.4.2 การเริ่มต้นใช้งาน PN532 NFC controller

2.4.2.1 เลือกโหมดการเชื่อมต่อตามตารางที่อยู่บนด้านซ้ายของบอร์ด

2.4.2.2 ทำการเชื่อมต่อกับขาต่างๆบนบอร์ดตามโหมดการใช้

2.4.2.3 แตะแท็ก NFC/RFID บนบอร์ด

2.5 ซอฟต์แวร์ Arduino IDE

ซอฟต์แวร์ Arduino IDE คือ เครื่องมือการเขียนโปรแกรมที่มีใช้งานได้กับ Arduino ได้ทุกรุ่น โดยภายในจะมีเครื่องมือที่จะเป็นสำหรับติดต่อ Arduino เช่น การค้นหา Arduino ที่ติดต่อกับเครื่องคอมพิวเตอร์ การเลือกรุ่น Arduinoที่ต่ออยู่เพื่อนตรวจสอบว่าขนาดของโปรแกรมที่เขียนหรือ Library ต่างๆ รองรับกับ Arduino รุ่นนั้นๆ ไหม อีกทั้งยังมีโปรแกรมติดต่อผ่าน Serial port โดยตรงสำหรับคอมพิวเตอร์

2.5.1 ระบบปฏิบัติการที่รองรับ

2.5.1.1 Windows

2.5.1.2 Mac OS X

2.5.1.3 Linux 32 , 64 bit

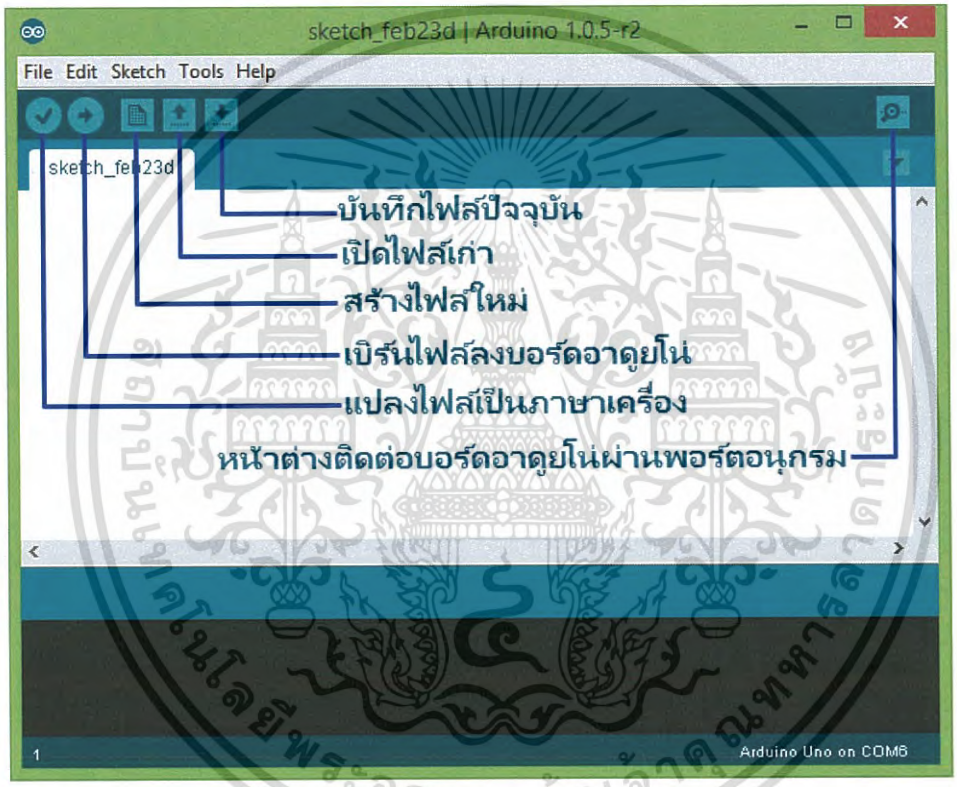
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.2 การเริ่มต้นใช้งาน Arduino IDE

2.5.2.1 ดาวน์โหลดซอฟต์แวร์ได้ที่ http://arduino.cc/en/Main/Software (เลือกให้เหมาะสมกับระบบปฏิบัติการที่ใช้)

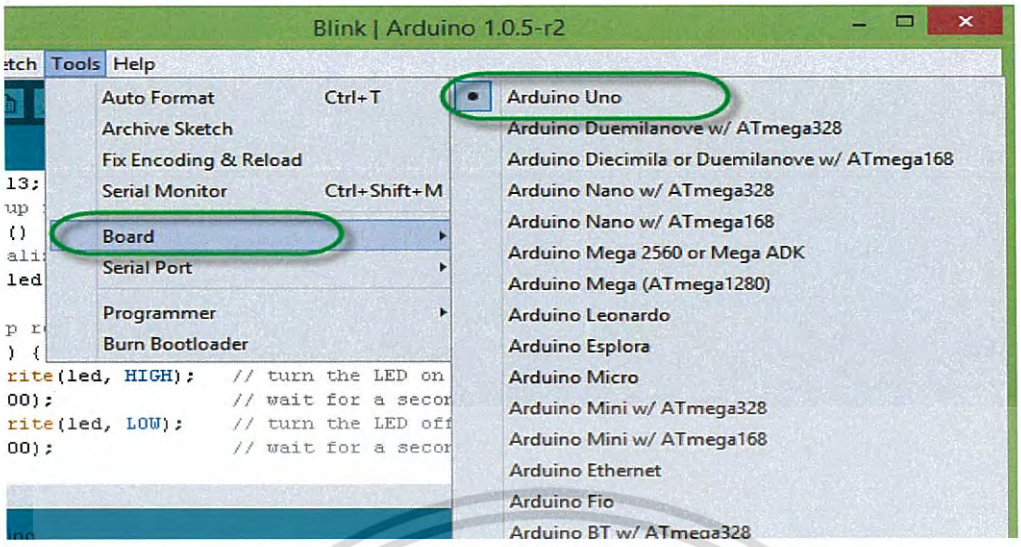
2.5.2.2 ติดตั้งซอฟต์แวร์ตามขั้นตอน

2.5.2.3 ทำการตั้งค่าเบื้องต้น

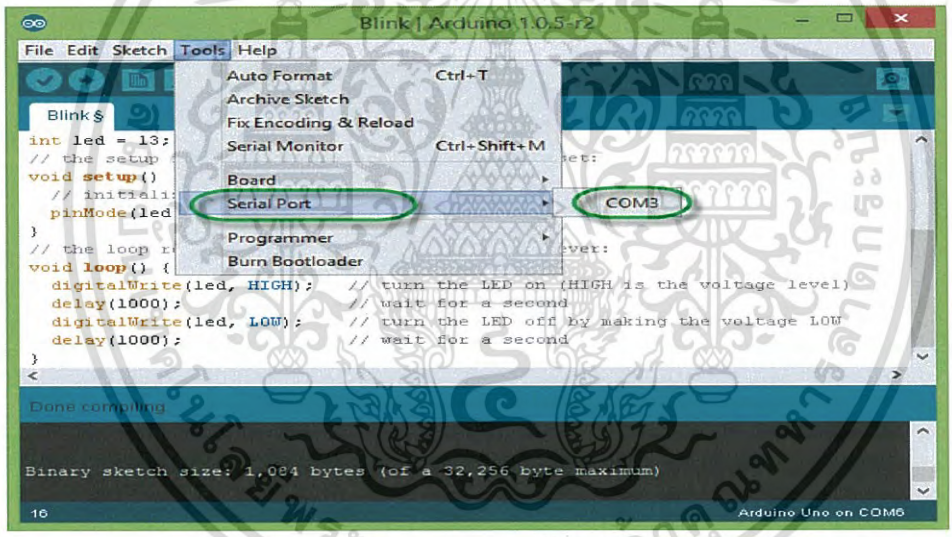


รูปที่ 2.7 หน้าต่างซอฟต์แวร์ Arduino IDE และไอคอนที่ใช้งานบ่อย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

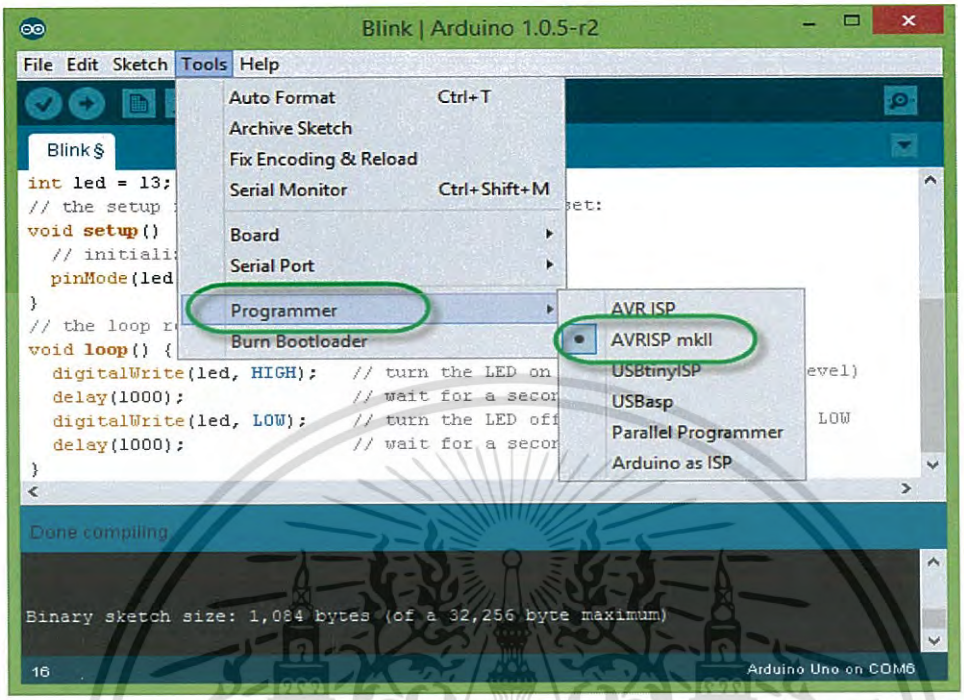


รูปที่ 2.8 การตั้งค่ารุ่นของบอร์ด Arduino ที่ใช้



รูปที่ 2.9 การเลือกพอร์ตในการเชื่อมต่อกับคอมพิวเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.10 การเลือกชนิดของโปรแกรม

2.5.2.4 เขียนโปรแกรมและทำการประมวลผลขึ้นบอร์ด Arduino

2.6 ภาษาซีพลัสพลัส (C++)

ภาษาซีพลัสพลัส เป็นภาษาคอมพิวเตอร์เพื่อวัตถุประสงค์ทั่วไป ซึ่งสามารถเขียนโปรแกรมได้ทั้งแบบออบเจ็ค และการเขียนแบบปกติทั่วไป และยังมีเครื่องมืออำนวยความสะดวกในการจัดการและเข้าถึงระดับหน่วยความจำ นอกจากนี้มันยังถูกนำไปใช้ในการเขียนโปรแกรมแบบต่างๆ มากมาย เช่น โปรแกรมคอมพิวเตอร์ ระบบฝังตัว (Embedded) เว็บเซิร์ฟเวอร์ การพัฒนาเกม และแอปพลิเคชันที่ต้องการประสิทธิภาพอย่างสูง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6.1 รูปแบบของการออกแบบภาษาซีพลัสพลัส

2.6.1.1 ภาษาซีพลัสพลัสได้ถูกออกแบบมาเพื่อเป็นภาษาสำหรับการเขียนโปรแกรมทั่วไป สามารถรองรับการเขียนโปรแกรมในระดับภาษาเครื่องได้ เช่นเดียวกับภาษาซี

2.6.1.2 ภาษาซีพลัสพลัสนั้นเป็นภาษาที่มีความซับซ้อนมากกว่าภาษาซี ภาษาซีพลัสพลัสได้รับการออกแบบเพื่อเข้ากันได้กับภาษาซีในเกือบทุกกรณี

2.6.1.3 มาตรฐานของภาษาซีพลัสพลัส ถูกออกแบบมาเพื่อไม่ให้มีการเจาะจงแพลตฟอร์มคอมพิวเตอร์

2.6.1.4 ภาษาซีพลัสพลัสถูกออกแบบมาให้รองรับรูปแบบการเขียนโปรแกรมที่หลากหลาย

ตัวอย่างโคด

```
#include <iostream>
Int main ()
{
std::cout<< "hello, world\n";
return 0;
}
```

2.6.2 โครงสร้างของโปรแกรมที่เขียนด้วยภาษาซีพลัสพลัส

โครงสร้างของโปรแกรมที่ขียนด้วยภาษาซีพลัสพลัสแบ่งย่อยได้เป็น 3 ส่วนดังนี้

2.6.2.1 ส่วนเรียกใช้ไฟล์อื่นๆ ส่วนใหญ่มักจะเป็นไฟล์ที่มีนามสกุลเป็น .h

2.6.2.2 ส่วนกำหนดชื่อในโปรแกรม เป็นส่วนที่ใช้กำหนดค่าคงที่ ตัวแปร และค่าอื่นๆ ที่ต้องการ

2.6.2.3 ส่วนคำสั่ง จะประกอบด้วยคำสั่งต่างๆ หรือฟังก์ชันอื่นๆ ที่ใช้ในการทำงานของโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่าง โครงสร้างโปรแกรมที่เขียนด้วยภาษาซีพลัสพลัส

```
#include <iostream.h>                    ส่วนเรียกใช้ไฟล์อื่น
char  ch;                                ส่วนกำหนดชื่อ
int  main(void)
{
 ch = 'A';
 cout<<"Hello world";                    ส่วนคำสั่ง
 return 0;
 }
```

2.6.3 ข้อดีของภาษาซีพลัสพลัส

2.6.3.1 ภาษาซีพลัสพลัสจะมีการทำงานที่ค่อนข้างเร็วมากเมื่อเทียบกับภาษาอื่น และยังสามาถดำเนินการกับฮาร์ดแวร์ได้ โดยที่โปรแกรมภาษาบางโปรแกรมอาจจะไม่สนับสนุนคุณลักษณะนี้

2.6.3.2 ภาษาC++สามารถเขียนโปรแกรมภาษาซี ได้ทั้งหมด ใช้ง่ายกว่าภาษาซี

2.6.3.3 สามารถทำงานได้บนเครื่องคอมพิวเตอร์ต่างประเภทกัน โดยอาศัยการประมวลผลโปรแกรมใหม่

2.6.3.4. ภาษาซีพลัสพลัสมีความเป็น Object Oriented Programming และยังเป็น Structure Programming ซึ่งเหมาะที่จะใช้ ศึกษาเกี่ยวกับการเขียนโปรแกรมสำหรับผู้เริ่มต้น และนอกจากนั้นถ้าหากเราจะเรียนเรื่อง Data Structure หรือทางด้านอัลกอริทึ่ม ในต่างประเทศจะนิยมใช้ภาษาซีพลัสพลัสในการสอน รวมถึงการเรียนรู้ถึงระบบการทำงานของระบบปฏิบัติการ ตำราส่วนใหญ่ก็จะใช้ภาษาซีพลัสพลัสในการสอน ซึ่งถ้าเราสามารถอ่านโค้ดภาษาซีพลัสพลัสรู้เรื่องก็จะทำให้เราเรียนรู้เกี่ยวกับการเป็นโปรแกรมเมอร์ได้ง่ายขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6.4 ข้อเสียของภาษาซีพลัสพลัส

2.6.4.1 ภาษาซีพลัสพลัสเวลาสร้าง function แล้วต้องสร้างไว้ตรงข้างบนไม่อย่างนั้นก็จะมองไม่เห็น

2.6.4.2 เป็นภาษาที่เรียนรู้ยาก

2.6.4.3 การตรวจสอบโปรแกรมทำได้ยาก

2.6.4.4 ไม่เหมาะกับการเขียนโปรแกรมที่เกี่ยวข้องกับการออกรายงานที่มีรูปแบบซับซ้อนมากๆ

2.7 การสื่อสารข้อมูล

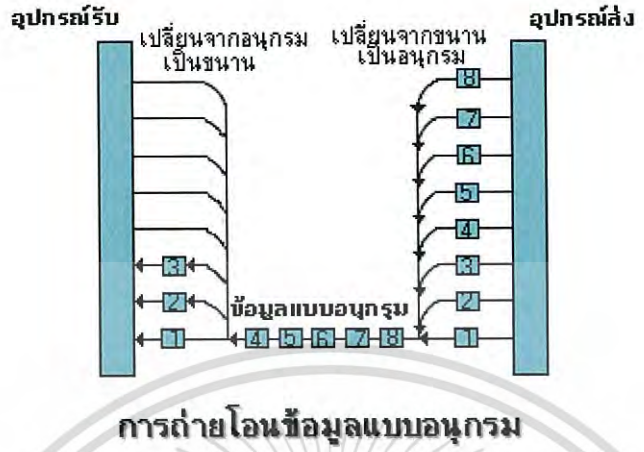
กระบวนการถ่ายโอนหรือแลกเปลี่ยนข้อมูลระหว่างผู้รับและผู้ส่งโดยผ่าน ช่องทางสื่อสาร เช่น อุปกรณ์อิเล็กทรอนิกส์หรือคอมพิวเตอร์เป็นตัวกลางในการส่งข้อมูลเพื่อให้ผู้ส่งและผู้รับเกิดความเข้าใจซึ่งกันและกัน

2.7.1 การสื่อสารข้อมูลแบบอนุกรม

ในการถ่ายโอนข้อมูลแบบอนุกรม ข้อมูลจะถูกส่งออกมาทีละบิตระหว่างจุดส่งและจุดรับ การส่งข้อมูลแบบนี้จะช้ากว่าแบบขนาน การถ่ายโอนข้อมูลแบบอนุกรมต้องการตัวกลางสำหรับการสื่อสารเพียงช่องเดียวหรือสายเพียงคู่เดียว ค่าใช้จ่ายจะถูกกว่าแบบขนานสำหรับการส่งระยะทางไกลๆ โดยเฉพาะเมื่อเรามีระบบการสื่อสารทางโทรศัพท์ไว้ใช้งานอยู่แล้ว ย่อมจะเป็นการประหยัดกว่าที่จะทำการติดต่อสื่อสารทีละ 8 ช่อง เพื่อการถ่ายโอนข้อมูลแบบขนาน

การถ่ายโอนข้อมูลแบบอนุกรมจะเริ่มโดยข้อมูลจากจุดส่งจะถูกเปลี่ยนให้เป็นสัญญาณอนุกรมเสียก่อน แล้วคอยทยอยส่งออกทีละบิตไปยังจุดรับ และที่จุดรับจะต้องมีกลไกในการเปลี่ยนข้อมูลที่ส่งมาทีละบิต ให้เป็นสัญญาณแบบขนานซึ่งลงตัวพอดี เช่น บิตที่ 1 ลงที่บัสข้อมูลที่ส่งมาทีละบิต ให้เป็นสัญญาณแบบขนานซึ่งลงตัวพอดี เช่น บิตที่ 1 ลงที่บัสข้อมูลเส้นที่ 1 ดังแสดงในรูปที่ 2.9

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



การถ่ายโอนข้อมูลแบบอนุกรม

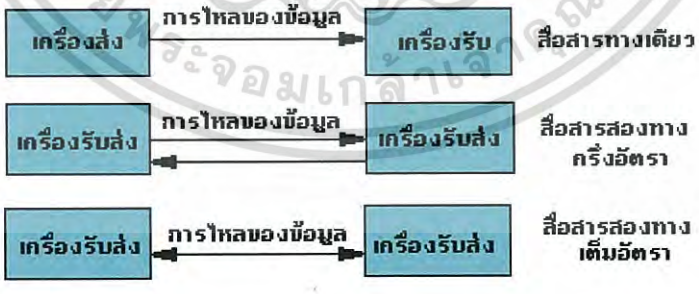
รูปที่ 2.11 การถ่ายโอนข้อมูลแบบอนุกรม

การติดต่อแบบอนุกรมอาจจะแบ่งตามรูปแบบการรับ-ส่ง ได้ 3 แบบ คือ

2.7.1.1 สื่อสารทางเดียว (simplex) ข้อมูลส่งได้ทางเดียวเท่านั้น บางครั้งก็เรียกว่าการส่งทิศทางเดียว (unidirectional data bus)

2.7.1.2 สื่อสารสองทางครึ่งอัตรา (half duplex) ข้อมูลสามารถส่งได้ทั้งสองสถานี แต่จะต้องผลัดกันส่งและผลัดกันรับ จะส่งและรับพร้อมกันไม่ได้

2.7.1.3 สื่อสารสองทางเต็มอัตรา (full duplex) ทั้งสองสถานีสามารถรับและส่งได้ในเวลาเดียวกัน



รูปแบบของการติดต่อสื่อสารข้อมูลแบบอนุกรม

รูปที่ 2.12 รูปแบบการติดต่อสื่อสารข้อมูลแบบอนุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความเร็วของการถ่ายโอนข้อมูลแบบอนุกรม หน่วยวัดเป็นบิตต่อวินาที (bps) หน่วยที่บรรยายถึงการเปลี่ยนแปลงของสัญญาณใน 1 วินาที เรียกว่าอัตราบอด (baud rate) ซึ่งเมื่อนำมาคูณกับจำนวนบิตใน 1 บอด จะได้อัตราบิต (bit rate) ซึ่งแสดงถึงการเปลี่ยนแปลงของสัญญาณ 1 ครั้ง ถ้าเขียนในรูปของสมการคณิตศาสตร์ก็จะได้

อัตราบิต (bit rate) = อัตราบอด (baud rate) x (จำนวนบิตใน 1 บอด) ...(2.1)

2.8 U-900A Flow Sensor

U-900A Flow Sensor เป็นอุปกรณ์ที่สามารถวัดอัตราการไหลได้ เมื่อนำมาใช้ร่วมกับ Arduino สามารถเขียนโปรแกรมที่สามารถบอกปริมาตรรวมได้

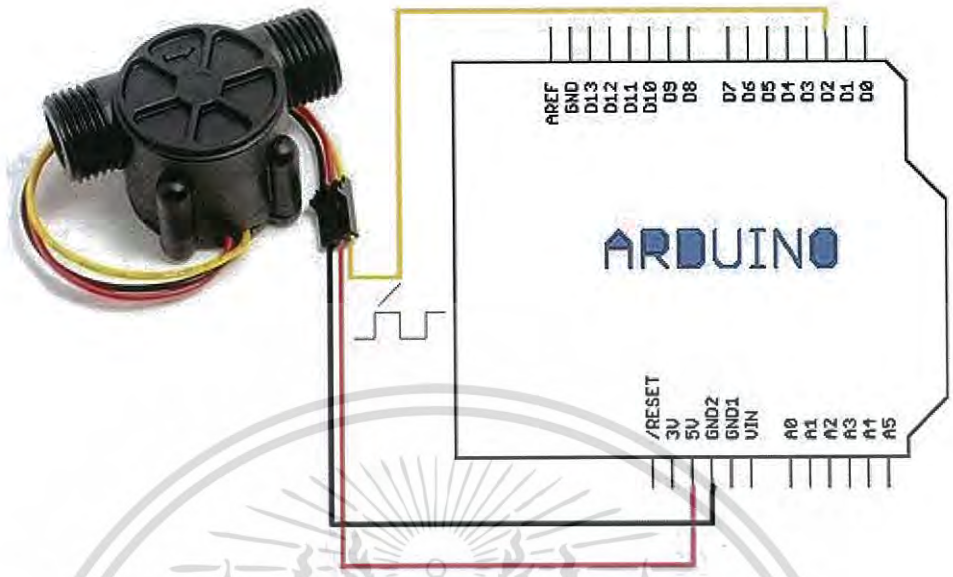
2.8.1 คุณสมบัติทางเทคนิคของ U-900A Flow Sensor

2.8.1.1 ทำงานได้เมื่อมีไฟเลี้ยง 5V DC
2.8.1.2 ทนความดันได้ไม่เกิน 1.2 Mpa
2.8.1.3 ช่วงการวัดอัตรการไหล 0.5 – 30 L/Min
2.8.1.4 สามารถใช้ได้กับน้ำมันทุกประเภท
2.8.1.5 ขนาดท่อ 1/2 นิ้ว

2.8.2 การเริ่มต้นใช้งาน U-900A Flow Sensor

2.8.2.1 ทำการต่อขา 5V GND และ TX เข้ากับบอร์ด Arduino ดังรูปที่ 2.13 และใช้งานร่วมกับซอฟต์แวร์ Arduino IDE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.13 การเชื่อมต่อขาระหว่างบอร์ด Arduino และ YF-S201 Water Flow Sensor [3]

2.8.2.2 ทดสอบปล่อยน้ำไหลผ่านตัวเซนเซอร์ ตามทิศทางที่ระบุไว้บนเซนเซอร์

2.8.3 หลักการทำงานของ U-900A flow sensor

ข้อมูลที่ได้จากสายสัญญาณ จะอยู่ในรูปของดิจิทัลพัลส์ที่ duty cycle คงที่ (50% ± 10%) ความถี่สัญญาณที่ออกมาจะแปรเปลี่ยนตามความเร็วการไหลของน้ำ โดยตัวเซนเซอร์ปล่อยสัญญาณพัลส์ออกมา 1 ลูกเมื่อใบพัดภายในตัวมิเตอร์หมุนไป 1 รอบ ดังนั้น หลักการวัดอัตราการไหลน้ำจึงใช้การนับ จำนวนพัลส์ที่ถูกส่งออกมาใน 1 หน่วยเวลา สำหรับรุ่น U-900A ที่อัตราการไหล 1 L/min สัญญาณที่ออกมามีความถี่ 7.5 Hz (ข้อมูลจาก datasheet) ดังนั้นเราใช้ค่านี้ในการคำนวณอัตราการไหลเฉลี่ยออกมา

2.8.4 การคำนวณอัตราการไหลและปริมาตร

$$Q = \text{Frequency} / \text{Coefficient} \qquad (2.2)$$

โดยให้ Q แทนอัตราการไหลในหน่วย (L/min)

Frequency แทนจำนวน pulse ที่นับได้ใน 1 วินาที

coefficient แทนค่าคงที่ของความถี่ที่ปล่อยออกมาเทียบกับอัตราการไหล (ในที่นี้คือ 7.5 Hz per L/min) ขอเรียกแทนว่า calibration0. factor

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับการหา Frequency นั้น จะใช้จำนวน pulse ที่นับได้ใน 1 วินาที ในการหา (หน่วย pulse/sec) แต่จริงแล้ว

สามารถกำหนด stepTime การนับ pulse เองได้ (ขึ้นกับความละเอียดที่ต้องการ) เวลาจริงที่บอร์ดนับ pulse ได้ (delTime) จะคาดเคลื่อนจาก stepTime ที่เราตั้งไว้เล็กน้อย ตามการประมวลของบอร์ด

ดังนั้น จำนวน pulse ที่อ่านได้จะต้องนำมาเทียบบัญญัติไตรยางศ์ให้ได้จำนวน pulse เทียบเท่าที่นับได้ใน 1 วินาที (จะได้หน่วย pulse/sec พอดี) ดังนี้

Frequency = 1000 * pulseCount / delTime (2.3)

สำหรับตัวอย่างนี้ พบว่า stepTime ที่เหมาะสม (จากการทดสอบ) อยู่ที่ 1000 – 2000 ms (1-2 วินาที) ต่อการนับ pulse แต่ละครั้ง หากอ่านค่าถี่ขึ้น (stepTime น้อยกว่า 1000 ms) ค่าที่อ่านได้จะแกว่งแบบคงที่ และหาก stepTime มากกว่า 2000 ms ค่าอัตราการไหลที่ได้จะเริ่มน้อยกว่าความเป็นจริง

ขั้นตอนต่อไปทำการแปลงหลักการข้างต้นเป็นโปรแกรมเบื้องต้นสำหรับรายงานอัตราการไหล โดยต้องการให้บอร์ดรับสัญญาณ pulse ในแต่ละลูกที่เข้ามา เพื่อไปทำการนับจำนวน pulse ในเวลาที่กำหนด กระบวนการนี้จะใช้ feature interruption ของ Arduino ทำการอ่าน จากนั้นโปรแกรมจะเข้าสู่กระบวนการนับ pulse โดยคำสั่ง attachInterrupt จะสั่งให้ฟังก์ชั่น pulseCounter(); นับ pulse (ตัวแปร pulseCount) เมื่อ pulse ที่ถูกส่งเข้ามาเปลี่ยนสถานะจาก HIGH เป็น LOW (แทนด้วย FALLING)

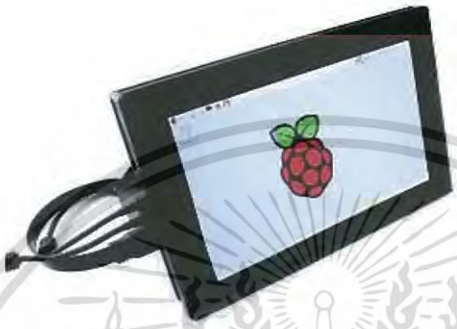
เมื่อเวลาผ่านไปทุกๆ 1 วินาที (คือค่า stepTime) โปรแกรมจะเรียกฟังก์ชั่น measureFlow(); ทำให้โปรแกรมสลับการทำงาน (detachInterrupt นั่นคือ หยุดนับ pulse) มาคำนวณพารามิเตอร์จาก pulse ที่นับได้ในช่วง stepTime [**] แล้วแสดงค่าที่คำนวณได้ผ่านทาง Serial Monitor (ฟังก์ชั่น reportResult();) จากนั้นจะรีเซ็ตตัวแปร pulseCount แล้วกลับเข้าสู่คำสั่ง attachInterrupt เพื่อทำการนับ pulse แล้วคำนวณพารามิเตอร์ออกมาในรอบถัดไป

ในระหว่างที่โปรแกรมคำนวณค่าอัตราการไหลออกมาให้เราสัญญาณพัลส์จากเซนเซอร์ที่เข้ามาจะไม่ถูกนับเพราะได้ detachInterrupt ออกมาคำนวณด้วย feature Interruption ของ Arduino นี้ ทำให้ค่าอัตราการไหลที่คำนวณได้แม่นยำขึ้น)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.9 จอ Touch Screen 10 นิ้ว

เป็นส่วนที่ใช้สื่อสารระหว่างระบบและผู้ใช้ ทำหน้าที่รับข้อมูลจากผู้ใช้และส่งต่อไปยังบอร์ด Arduino ผ่านทางบอร์ด Raspberry Pi และแสดงผล



รูปที่ 2.14 จอ touch screen 10 นิ้ว [4]

2.9.1 คุณสมบัติทางเทคนิคของ จอ Touch Screen 10 นิ้ว

2.9.1 ขนาด 10.1 นิ้ว

2.9.2 ความละเอียด 1280 x 720 pixel

2.9.3 ใช้แผงพาแนลแบบ IPS

2.9.4 รองรับการเชื่อมต่อแบบ HDMI

2.10 การเชื่อมต่อแบบ SPI

SPI หรือ Serial Peripheral Interface เป็นวิธีการสื่อสารอนุกรมแบบ Synchronous อีกรูปแบบหนึ่ง ซึ่งทำงานในรูปแบบที่ให้อุปกรณ์ตัวหนึ่งทำหน้าที่เป็น Master ในขณะที่อีกตัวหนึ่งทำหน้าที่เป็น Slave และสามารถส่งข้อมูลในโหมด Full-duplex นั่นหมายความว่า สัญญาณสามารถส่งหากันได้ระหว่าง Master และ Slave ได้อย่างต่อเนื่อง รูปแบบข้อมูลการสื่อสารหรือ Protocol ของแบบ SPI นี้ ไม่ได้มาตรฐานกำหนดตายตัว ว่าข้อมูลที่ส่งหากันต้องอยู่ในรูปแบบหรือ Format แบบไหน เป็นการคิด Protocol การสื่อสารกันเอาเอง หรือดูจาก Datasheet ของอุปกรณ์ ยกตัวอย่างอุปกรณ์ที่ใช้การสื่อสารแบบ SPI ได้แก่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

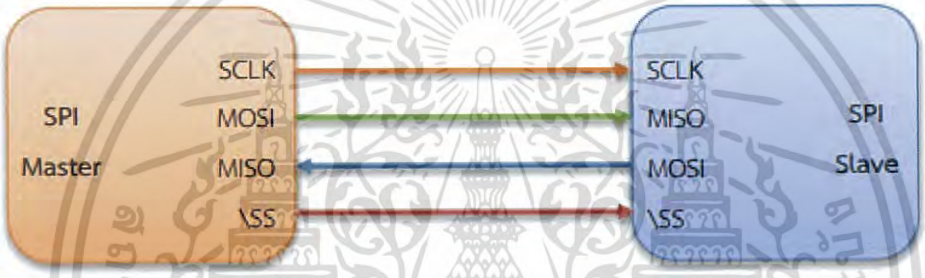
2.10.1 โมดูลแปลงสัญญาณอนาล็อกเป็นดิจิตอล และโมดูลแปลงสัญญาณดิจิตอลเป็นอนาล็อก

2.10.2 การติดต่อกับหน่วยความจำ EEPROM และ FLASH

2.10.3 โมดูลนาฬิกาดิจิตอล หรือ Real Time Clock : RTC

2.10.4 เซ็นเซอร์วัดอุณหภูมิ และความดัน

อุปกรณ์อื่น ๆ เช่น signal mixer , Potentiometer , LCD controller , USART , CAN controller , USB controller , Amplifier



รูปที่ 2.15 แสดงการเชื่อมต่อการสื่อสารแบบ SPI ระหว่างอุปกรณ์ Master – Slave โดยมสายสัญญาณ 4 เส้น หรือ Four Wire ประกอบด้วย

2.10.5 SCLK (Serial Clock) ใช้ส่งสัญญาณนาฬิกาจากอุปกรณ์ Master ไปยังอุปกรณ์ Slave เพื่อกำหนดจังหวะการรับส่งข้อมูล

2.10.6 MOSI (Master Out Slave In) ใช้ส่งข้อมูลจากอุปกรณ์ Master ไปยังอุปกรณ์ Slave

2.10.7 MISO (Master In Slave Out) ใช้รับข้อมูลจากอุปกรณ์ Slave

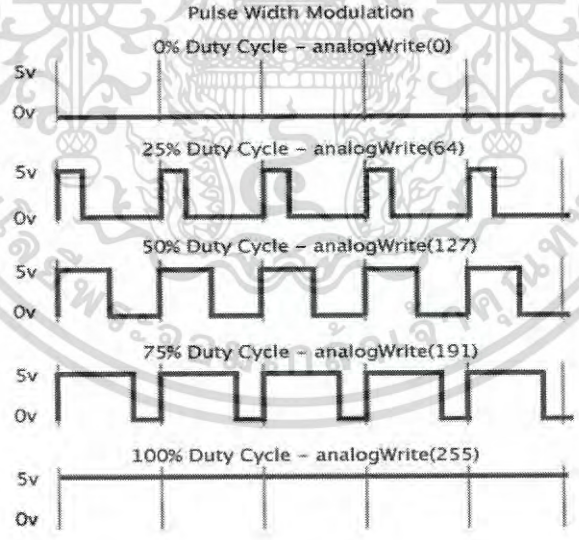
2.10.8 SS (Slave Select) หรือ ขา CS (Chip Select) ใช้ส่งสัญญาณ Low ไปยังอุปกรณ์ Slave ที่ต้องการรับส่งข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.11 Pulse Width Modulation (PWM) บน Arduino

PWM หมายถึง Pulse Width Modulation เป็นเทคนิคที่ Arduino ใช้ในการควบคุมวงจรและเขียนค่าแบบแอนาล็อก(Analog) ด้วยพอร์ตดิจิทัล(Digital) โดยปกติแล้วพอร์ตดิจิตัลจะสามารถมีได้แค่ 2 สถานะ คือ HIGH (5 โวล์ท) กับ LOW (0 โวล์ท)เท่านั้น จึงทำให้สร้างค่าสัญญาณลอจิคได้เพียง เปิดหรือปิด (1 หรือ 0) เท่านั้นซึ่งการใช้เทคนิค PWM นั้น จะเป็นการทำให้พอร์ตดิจิทัล สามารถเขียนค่าได้มากกว่า HIGH หรือ LOW โดยทำให้สามารถเขียนค่าเป็นแบบแอนาล็อกได้ (0-255) โดยวิธีการนั้นจะใช้การปรับสถานะของสัญญาณลอจิก HIGH / LOW สลับกันไปมาด้วยคาบเวลาหนึ่งๆ โดยค่าที่ได้นั้นจะขึ้นอยู่กับสัดส่วนเวลาของสัญญาณในช่วงเวลาที่มีสถานะเป็น HIGH กับช่วงเวลาที่เป็น LOW โดย ช่วงเวลาทั้งหมดที่สัญญาณมีสถานะเป็น HIGH นั้นเราจะเรียกว่าเป็น "ความกว้าง Pulse (Pulse Width)"

โดยสัญญาณพัลส์ เมื่อเทียบ % ของช่วงเวลาที่เป็น HIGH (หรือก็คือ % ของ Pulse Width) กับ % ของคาบเวลา (Period) ของพัลส์ลูกนั้นๆ เราจะเรียกว่า Duty Cycle ดังรูปที่ 2.14



รูปที่ 2.16 Pulse Width Modulation [5]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.12 การใช้งาน Interrupt บน Arduino

อินเตอร์รัพท์ (Interrupt) คือการขัดจังหวะการทำงานของโปรแกรมปกติ เมื่อเกิดเหตุการณ์บางอย่างขึ้น ทำให้ซีพียูไปทำงานที่กำหนดไว้เมื่อเกิดอินเตอร์รัพท์

2.12.1 ชนิดของอินเตอร์รัพท์

แบ่งตามชนิดของการเกิดได้ดังนี้

2.12.1.1อินเตอร์รัพท์จากภายนอก เช่น การเปลี่ยนสถานะลอจิกของพอร์ตใดพอร์ตหนึ่ง

2.12.1.2 อินเตอร์รัพท์จากภายใน เช่น อินเตอร์รัพท์ที่เกิดจากทามเมอร์

2.12.1.3 การควบคุมอินเตอร์รัพท์

การควบคุมอินเตอร์รัพท์คือการควบคุมว่าจะให้ซีพียูตอบสนองต่ออินเตอร์รัพท์หรือไม่ แบ่งได้ดังนี้

2.12.1.5 Disable Interrupt คือการควบคุมให้ซีพียูไม่ตอบสนองกับอินเตอร์รัพท์ เมื่อเกิดการอินเตอร์รัพท์ขึ้นซีพียูจะปล่อยผ่านอินเตอร์รัพท์นั้น

2.12.1.4 Enable Interrupt คือการควบคุมให้ซีพียูตอบสนองต่ออินเตอร์รัพท์ไปตามปกติ

การควบคุมอินเตอร์รัพท์จะใช้ในกรณีที่ต้องการให้ซีพียูกระทำคำสั่งที่ไม่สามารถหยุดการทำงานได้ เช่น การนับเวลา หากมีการอินเตอร์รัพท์เกิดขึ้นจะทำให้การนับเวลาคลาดเคลื่อนได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.12.2 การใช้งานอินเตอร์รัพท์ใน Arduino

ใน Arduino การใช้งานอินเตอร์รัพท์จากภายนอกใช้เพียงแค่การสร้างฟังก์ชั่นรอรับแล้วจึงใช้คำสั่งที่กำหนดว่าจะให้เกิดอินเตอร์รัพท์เมื่อไร แต่หากเป็นการอินเตอร์รัพท์จากภายในจค่อนข้างยุ่งยากมากๆ ดังนั้นในบทความนี้จึงจะกล่าวถึงการใช้อินเตอร์รัพท์จากภายนอกเท่านั้น

2.12.3.1 พอร์ตที่สามารถใช้งานอินเตอร์รัพท์ได้

ในบอร์ด Arduino จะมีพอร์ตที่สามารถใช้อินเตอร์รัพท์ได้แบบจำกัด โดยพอร์ตที่สามารถใช้อินเตอร์รัพท์ได้ในบอร์ด Arduino แต่ละรุ่นก็จะแตกต่างกัน ซึ่งสามารถดูได้ตามด้านล่างนี้

Board	int.0	int.1	int.2	int.3	int.4	int.5
Uno, Ethernet	2	3				
Mega2560	2	3	21	20	19	18
Leonardo	3	2	0	1	7	
Due			(see below)			

ในการสร้างอินเตอร์รัพท์ จะใช้ลำดับขาอินเตอร์รัพท์มาใส่ในฟังก์ชั่น เช่น ในตาราง บอร์ด Arduino Uno มีพอร์ต 2 เป็นขาอินเตอร์รัพท์ลำดับที่ 0 ดังนั้นเมื่อนำไปใช้ในคำสั่งต้องนำหมายเลข 0 ไปใส่ ส่วนใน Arduino Due สามารถใช้งานอินเตอร์รัพท์ได้ทุกขา หากนำไปใส่ในคำสั่ง ก็สามารถนำหมายเลขขาไปใส่ได้เลย

2.12.4 คำสั่งกำหนดใช้อินเตอร์รัพท์

คำสั่ง attachInterrupt() เป็นคำสั่งกำหนด และสร้างอินเตอร์รัพท์ โดยมีรูปแบบการใช้งานดังนี้รูปแบบคำสั่ง attachInterrupt(interrupt, ISR, mode)

2.12.4.1 interrupt คือลำดับขาอินเตอร์รัพท์ตามที่ได้ดูไปในตารางที่ผ่านมา ใน Arduino Due สามารถใส่หมายเลขขาลงไปได้เลย

2.12.4.2 ISR คือชื่อฟังก์ชั่นที่จะไปถูกเรียกขึ้นมาเมื่อเกิดอินเตอร์รัพท์

2.12.4.3 mode รูปแบบที่จะให้เกิดอินเตอร์รัพท์ มีทั้งหมด 5 รูปแบบดังนี้

- LOW จะเกิดอินเตอร์รัพท์ต่อเมื่อพอร์ตที่กำหนดไว้มีสถานะเป็น LOW

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- CHANGE จะเกิดอินเตอร์รัพท์เมื่อพอร์ตที่กำหนดไว้มีการเปลี่ยนสถานะ เช่น จากสถานะ HIGH เป็น LOW หรือจาก LOW เป็น HIGH

- RISING จะเกิดอินเตอร์รัพท์เมื่อพอร์ตที่กำหนดไว้มีการเปลี่ยนสถานะจาก LOW เป็น HIGH

- FALLING จะเกิดอินเตอร์รัพท์เมื่อพอร์ตที่กำหนดไว้มีการเปลี่ยนสถานะจาก HIGH เป็น LOW

- HIGH จะเกิดอินเตอร์รัพท์ต่อเมื่อพอร์ตที่กำหนดไว้มีสถานะเป็น HIGH

2.12.4.4 คำสั่งควบคุมอินเตอร์รัพท์

มีด้วยกัน 2 คำสั่ง คือ

- noInterrupt() เป็นคำสั่งที่ทำให้ซีพียูอยู่ในสถานะ Disable Interrupt
- Interrupt() เป็นคำสั่งที่ทำให้ซีพียูอยู่ในสถานะ Enable Interrupt

2.12.4.5 คำสั่งยกเลิกใช้อินเตอร์รัพท์

รูปแบบ detachInterrupt(interrupt)

2.12.4.5.1 คำสั่ง detachInterrupt() คือคำสั่งที่ใช้ในการยกเลิกการใช้งานอินเตอร์รัพท์ โดยพารามิเตอร์ interrupt ต้องใส่เป็นลำดับขาอินเตอร์รัพท์ หรือใน Arduino Due จะต้องใส่เป็นหมายเลขขาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.13 5V relay module

เป็นอุปกรณ์ที่ใช้ควบคุมการเปิด/ปิด flow sensor และ pump

รูปที่ 2.17 5V relay module [7]

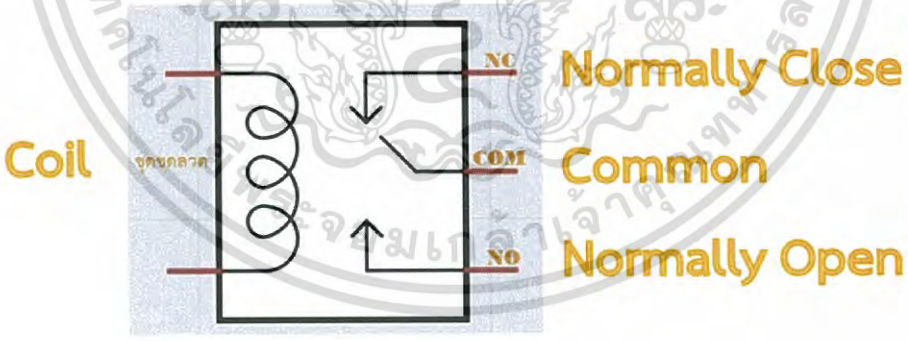
2.13.1 คุณสมบัติทางเทคนิคของ 5V relay module

2.13.1.1 มี output 1 channel

2.13.1.2 ทำงานที่แรงดัน 5 VDC

2.13.1.3 ทำงานแบบ Active High

2.13.1.4 Max Control Capacity:10A@250VAC



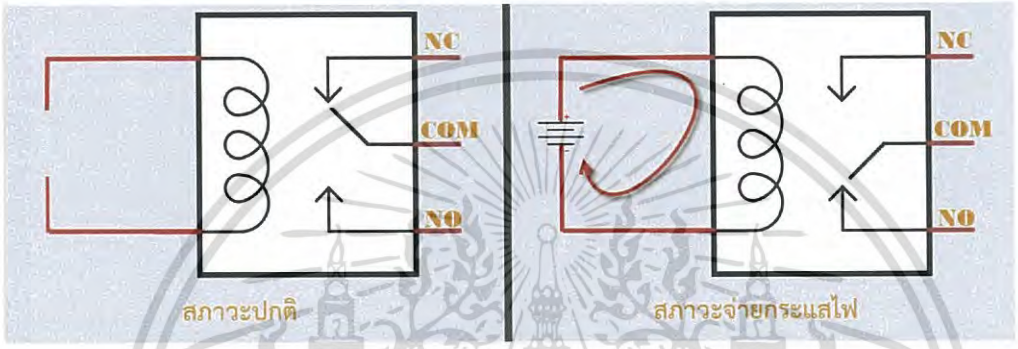
รูปที่ 2.18 สัญลักษณ์ในวงจรไฟฟ้าของรีเลย์ [7]

2.13.2 หลักการทำงานของ relay

ภายใน Relay จะประกอบไปด้วยขดลวดและหน้าสัมผัส NC(Normally Close) เป็นหน้าสัมผัสปกติปิดโดยในสภาวะปกติหน้าสัมผัสนี้จะต่อเข้ากับขาCOM (Common) และจะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลอยหรือไม่สัมผัสกันเมื่อมีกระแสไฟฟ้าไหลผ่านขดลวดหน้าสัมผัส NO(Normally Open) เป็นหน้าสัมผัสปกติเปิดโดยในสภาวะปกติจะลอยอยู่ไม่ถูกต่อกับขาCOM (Common) แต่จะเชื่อมต่อกันเมื่อมีกระแสไฟไหลผ่านขดลวดขา COM(Common) เป็นขาที่ถูกใช้งานร่วมกันระหว่าง NC และ NO ขึ้นอยู่กับว่าขณะนั้นมีกระแสไฟฟ้าไหลผ่านขดลวดหรือไม่ หน้าสัมผัสใน Relay 1 ตัวอาจมีมากกว่า 1 ชุดขึ้นอยู่กับผู้ผลิตและลักษณะของงานที่ถูกนำไปใช้



รูปที่ 2.19 การจ่ายไฟทั้ง 2 สถานะ [7]

หลักการทำงานของ Relay นั้น ในส่วนของขดลวดเมื่อมีกระแสไฟฟ้าไหลผ่าน จะทำให้ขดลวดเกิดการเหนี่ยวนำและทำหน้าที่เสมือนแม่เหล็กไฟฟ้าส่งผลให้ขา COM ที่เชื่อมต่ออยู่กับหน้าสัมผัส NC (ในสภาวะที่ยังไม่เกิดการเหนี่ยวนำ) ย้ายกลับเชื่อมต่อกับหน้าสัมผัส NO แทน และปล่อยให้ขา NC ลอย เมื่อมองที่ขา NC กับ COM และ NO กับ COM แล้วจะเห็นว่ามีการทำงานติด-ดับลักษณะคล้ายการทำงานของสวิชต์ เราสามารถอาศัยคุณสมบัตินี้ไปประยุกต์ใช้งานได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.14 Qt Creator

รูปที่ 2.20 โปรแกรมคิวต์ [8]

Qt (อ่านว่า คิวต์) เป็นเครื่องมือในการสร้างแอปพลิเคชั่น และ GUI ซึ่งสามารถทำงานบน Desktop PC , Smart Phone และ Embedded System สามารถทำงานได้หลายระบบปฏิบัติการ(OS) หรือ เรียกว่า Cross-platform แปลความได้ว่า เมื่อเรามีโปรแกรมที่ทำงานบน OS หนึ่ง เราไม่จำเป็นต้องเขียนโปรแกรมใหม่ สามารถนำโปรแกรมไป Compile เพื่อให้สามารถทำงานบน OS อื่นได้โดยไม่ต้องแก้โปรแกรมเลย เมื่อโปรแกรมทำงานลักษณะจะเปลี่ยนไปตามสิ่งแวดล้อมของ OS นั้น ๆ โดยอัตโนมัติการเขียน GUI ให้กับแอพพลิเคชันหรือระบบต่าง ๆ ในปัจจุบัน มีทางเลือกหลายทางในการพัฒนา มีเครื่องมือหลายตัวให้เลือกใช้ เช่น VC# และ VB .NET บน WIN CE เป็นต้น แต่สำหรับ Qt แล้ว ถือว่าเป็นอีกทางเลือกหนึ่งที่น่าสนใจ เพราะ สามารถเลือกใช้ API, Library ต่าง ๆ ซึ่งมีมากมายได้เช่นกัน (เปรียบเทียบได้กับ MSDN จาก Microsoft แต่เป็น Opensource) ไม่จำเป็นต้องเริ่มตั้งต้นใหม่ ซึ่งเสียเวลาโดยเปล่าประโยชน์ Qt จะมี API และ Library ต่าง ๆ ที่เขียนด้วยภาษา C++ ทำให้สามารถพัฒนาแอพพลิเคชั่นแบบ GUI และ ยังสนับสนุนการพัฒนาทั้ง C++, Java, Python, Perl, Pascal และ PHP ทั้งนี้ก็ขึ้นอยู่กับผู้พัฒนาว่าจะเลือกใช้ภาษาใดในการพัฒนาความสามารถที่นอกเหนือจากส่วนต่อประสานงานกราฟฟิคกับผู้ใช้ เช่น การติดต่อกับฐานข้อมูลSQL การอ่านข้อมูลXML การบริหารThreadด้านเครือข่าย และการจัดการไฟล์

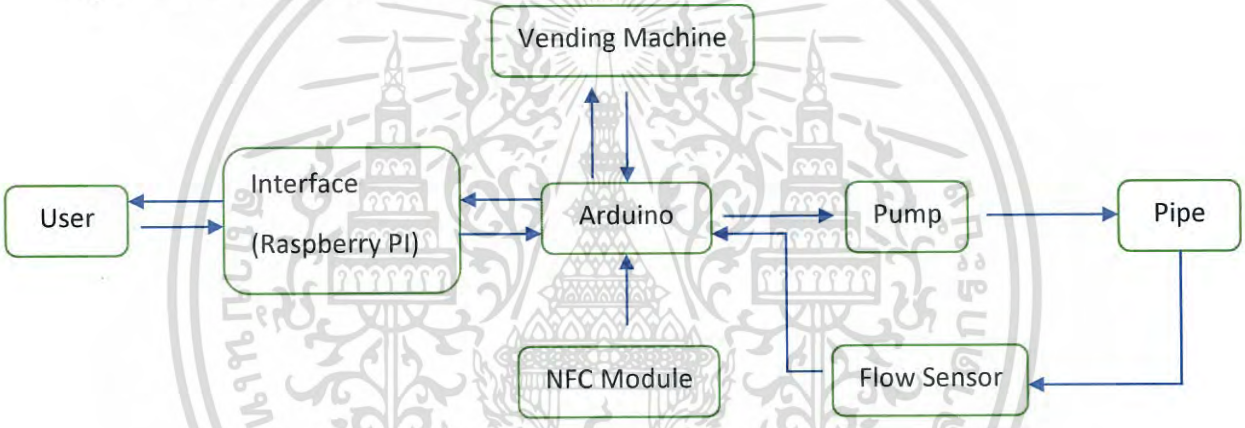
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

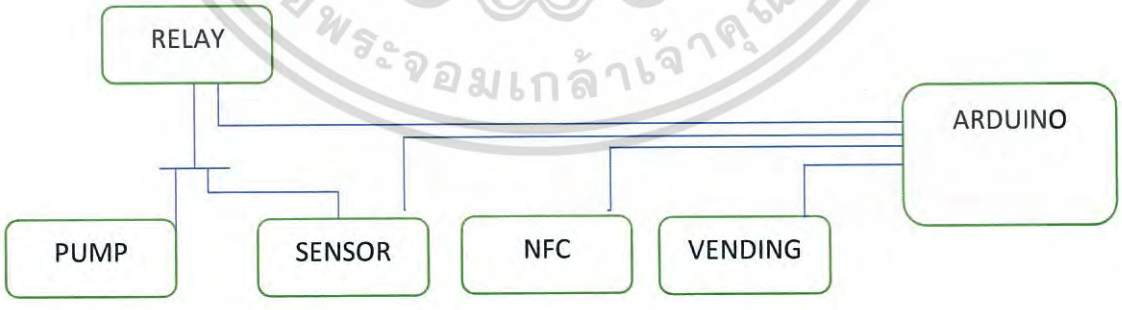
การออกแบบและการจัดทำปริญญานิพนธ์

3.1 การออกแบบ

การออกแบบระบบจ่ายน้ำมันอัตโนมัติ มีจุดประสงค์เพื่อเพิ่มความสะดวกสบายให้กับผู้ใช้น้ำมัน รวมไปถึงการพัฒนาระบบตู้จ่ายน้ำมันที่มีอยู่ในปัจจุบัน บล็อกไดอะแกรมระบบตู้จ่ายน้ำมันดังรูปที่ 3.1 และส่วนควบคุมการทำงานดังรูปที่ 3.2



รูปที่ 3.1 บล็อกไดอะแกรมของระบบ



รูปที่ 3.2 ส่วนวงจรควบคุมการทำงานและส่งข้อมูล

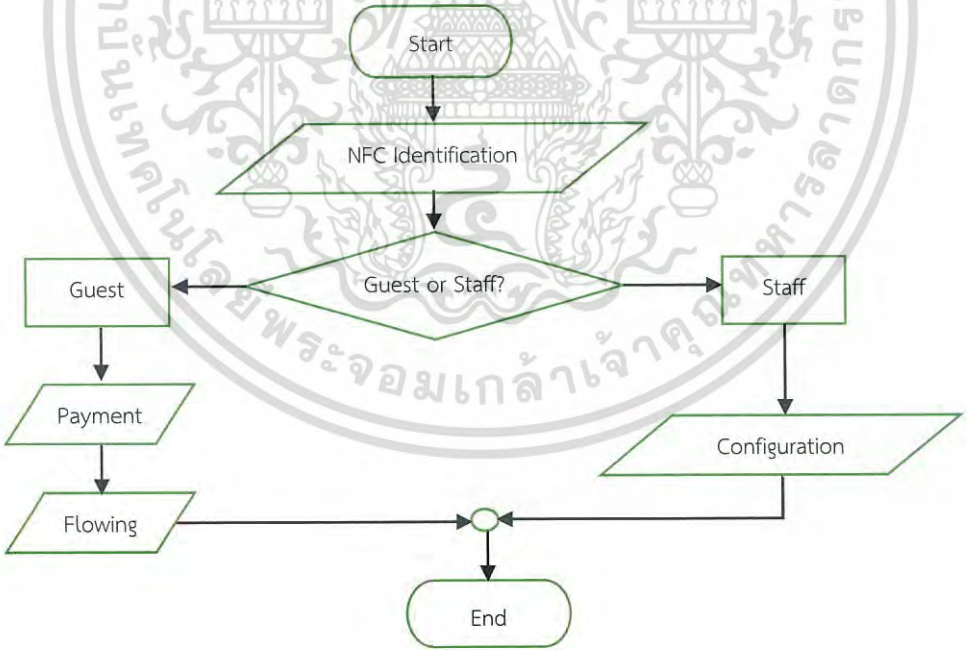
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.1 การออกแบบส่วนติดต่อและแสดงผลกับผู้ใช้งาน

ส่วนติดต่อและแสดงผลกับผู้ใช้งานออกแบบโดย Qt creator บนตัวควบคุมหลักคือ RaspberryPi ซึ่งเชื่อมต่ออยู่กับจอ Touch Screen เพื่อส่งคำสั่งไปยังบอร์ด Arduino

3.1.2 การระบุตัวตนผ่าน NFC module

การสื่อสารระหว่าง NFC module กับ บอร์ด Arduino เป็นการสื่อสารแบบ SPI โดยการเชื่อมต่อขา SCK MOSI MISO SCL GND และ ไฟเลี้ยง 5V ของบอร์ด Arduino กับขา SCK MOSI MISO SCL GND และ ไฟเลี้ยง 5V ของ NFC module โดย NFC Module จะทำการส่งชุดรหัสของบัตรแต่ละใบไปยังบอร์ด Arduino เพื่อใช้ในการระบุตัวตนของผู้ใช้ ซึ่งมีขั้นตอนดังรูปที่ 3.3



รูปที่ 3.3 ผังขั้นตอนการระบุตัวตน

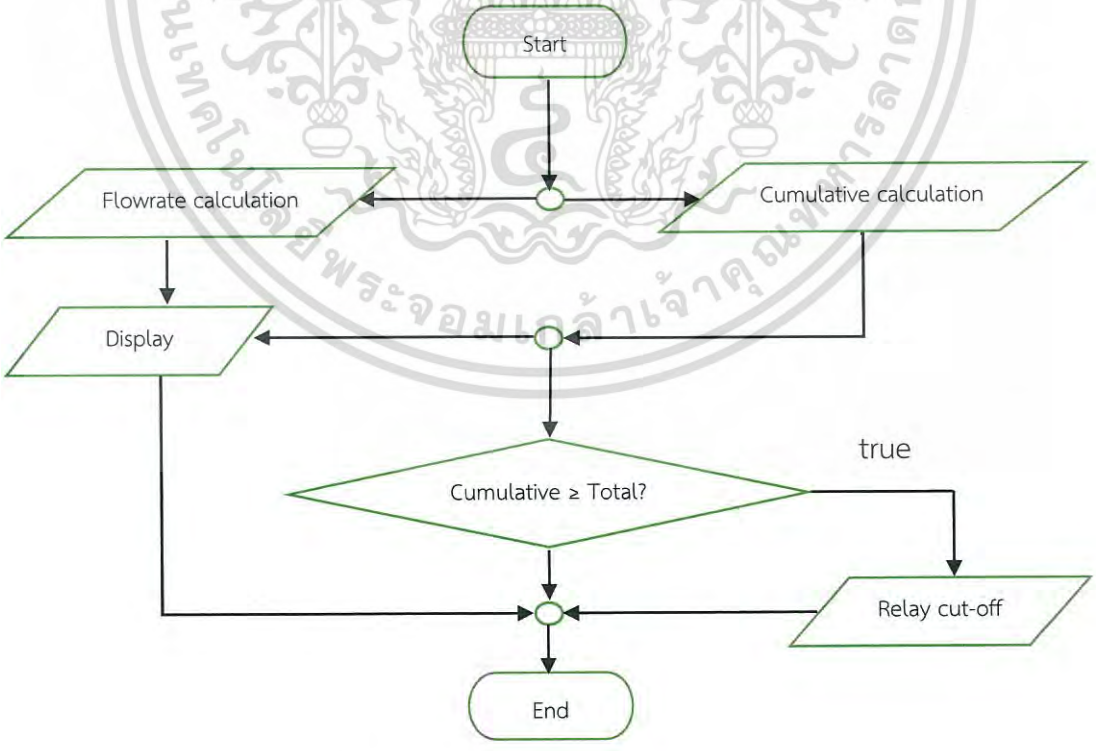
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.3 การออกแบบการสื่อสารระหว่างบอร์ด Arduino กับ เครื่องหยอดเหรียญ

การสื่อสารระหว่างเครื่องหยอดเหรียญกับบอร์ด Arduino เป็นการสื่อสารแบบอนุกรมโดยใช้พิน RX TX และ GND เชื่อมต่อกับบอร์ด Arduino โดยตรง ทำหน้าที่รับคำสั่งเปิด/ปิดช่องรับเหรียญ และส่งชุดรหัสของเหรียญแต่ละเหรียญที่รับเข้ามา เพื่อนำไปประมวลผลหาค่าของเหรียญ คำนวณยอดสุทธิ และแสดงผลบนจอ Touch Screen ในลำดับต่อไป

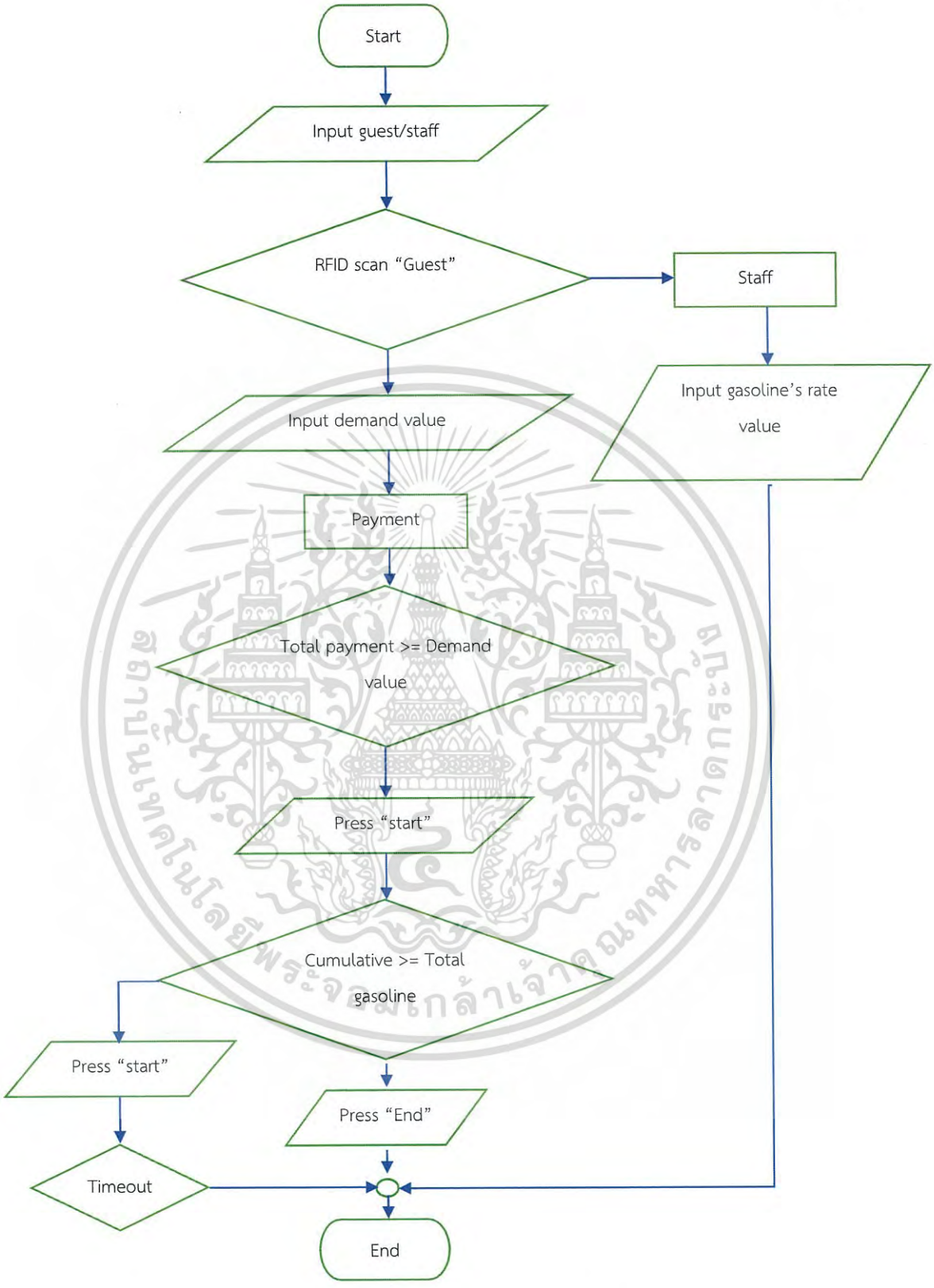
3.1.4 การควบคุมการไหล

การสื่อสารระหว่าง Flow Meter และ บอร์ด Arduino เป็นการสื่อสารแบบอนุกรมโดยส่งสัญญาณ PWM (Pulse Width Modulation) ผ่านพิน PWM บนบอร์ด Arduino เพื่อใช้ในการคำนวณอัตราการไหลและปริมาตรทั้งหมดดังรูปที่ 3.4



รูปที่ 3.4 ผังการควบคุมการไหล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.5 ผังภาพรวมการทำงานของระบบจ่ายน้ำมัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.5 ภาพรวมการทำงานของระบบ

รูปที่ 3.5 แสดงถึงภาพรวมการทำงานของระบบเริ่มต้นที่การระบุตัวตน และสิ้นสุดที่การควบคุมการไหล โดยมีขั้นตอนการทำงานดังนี้

3.1.5.1 ผู้ใช้ทำการระบุตัวตนผ่าน NFC module เพื่อไปสู่ขั้นตอนถัดไป หากเป็นผู้ควบคุมดูแลจะเข้าสู่การตั้งค่าราคา หากเป็นลูกค้าจะเข้าสู่ขั้นตอนการป้อนจำนวนเงินที่ต้องการเติม และเข้าสู่ขั้นตอนการชำระเงิน

3.1.5.2 ผู้ใช้ทำการชำระเงินผ่านเครื่องหยอดเหรียญ จากนั้นบอร์ด Arduino จะทำการอ่านค่าเหรียญและเก็บค่า เมื่อครบตามความต้องการของลูกค้าจะเข้าสู่ขั้นตอนการเติมน้ำมัน

3.1.5.3 เมื่อเสร็จสิ้นขั้นตอนการชำระเงิน ผู้ใช้สามารถกดปุ่ม Start เพื่อเริ่มจ่ายน้ำมันได้ทันที จากนั้น Flow sensor จะทำการส่งสัญญาณไปยังบอร์ด Arduino เพื่อใช้ในการคำนวณอัตราการไหลและปริมาตรทั้งหมด ผู้ใช้สามารถหยุดการจ่ายน้ำมันได้ทุกเมื่อ ระบบจะหยุดการจ่ายน้ำมันเมื่อครบตาม timeout ของระบบ หรือเมื่อปริมาตรการไหลครบตามจำนวณที่ต้องการเติม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 เครื่องมือที่ใช้ในการทดลอง

3.2.1 บอร์ด Raspberry Pi 3 Model b

3.2.2 จอ touch screen 10 นิ้ว

3.2.3 บอร์ด Arduino Mega2560 rev3

3.2.4 เครื่องหยอดเหรียญ (Vending Machine)

3.2.5 NFC module

3.2.6 ซอฟท์แวร์ Arduino IDE

3.2.7 U-900A Flow sensor

3.2.8 Pump (Blower/Water pump)

3.2.9 Relay 5 Vdc

3.2.10 สายไฟ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 การจัดเก็บผลการทดลอง

3.3.1 การออกแบบ Graphic User Interface (GUI)

3.3.1.1 ส่วนของผู้ใช้ (user) ส่งคำสั่งจำนวนเงินที่ต้องการเติมผ่านจอแสดงผล แล้วหยอดเหรียญตามจำนวนที่ต้องการเมื่อหยอดครบจะสามารถกด start เพื่อปล่อยน้ำมัน

3.3.1.2 ส่วนของพนักงานตั้งค่า (staff) ต้องมีบัตรระบุตัวตนเพื่อยืนยันตัวตนจึงจะสามารถทำการตั้งค่าราคาน้ำมันต่อลิตรได้

3.3.2 การทดสอบการอ่านค่าของเหรียญแต่ละแบบ และทำการส่งไปยังบอร์ด Arduino

ทำการเชื่อมต่อเครื่องหยอดเหรียญกับบอร์ด Arduino ผ่านทางขา RX TX โดยทดสอบคำสั่งเปิด/ปิด ช่องรับเหรียญ และการระบุชุดรหัสของเหรียญแต่ละแบบ

3.3.3 การทดสอบการทำงานของ NFC module

ทำการเชื่อมต่อ NFC module กับ บอร์ด Arduino โดยใช้การเชื่อมต่อแบบ SPI และอ่านค่าจากบัตรแต่ละใบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.4 การทดสอบการทำงานของ Relay 5 Vdc

ทำการต่อวงจรระหว่าง pump , sensor โดยใช้ relay ควบคุมการเปิด/ปิด pump และ sensor และใช้ไฟเลี้ยง 5Vdc จากบอร์ด Arduino

3.3.5 การทดสอบการทำงานของ pump

ใช้ relay ควบคุมการเปิด/ปิด pump และใช้ไฟเลี้ยง 5Vdc จากบอร์ด Arduino

3.3.6 การทดสอบการทำงานของ U-900A Flow sensor

ใช้ relay ควบคุมการเปิด/ปิด flow sensor และใช้ไฟเลี้ยง 5Vdc จากนั้นทำการส่งสัญญาณในรูปแบบ PWM ไปยัง PWM pin บนบอร์ด Arduino เพื่อนับพัลส์ และคำนวณอัตราการไหลและปริมาตรในลำดับถัดไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

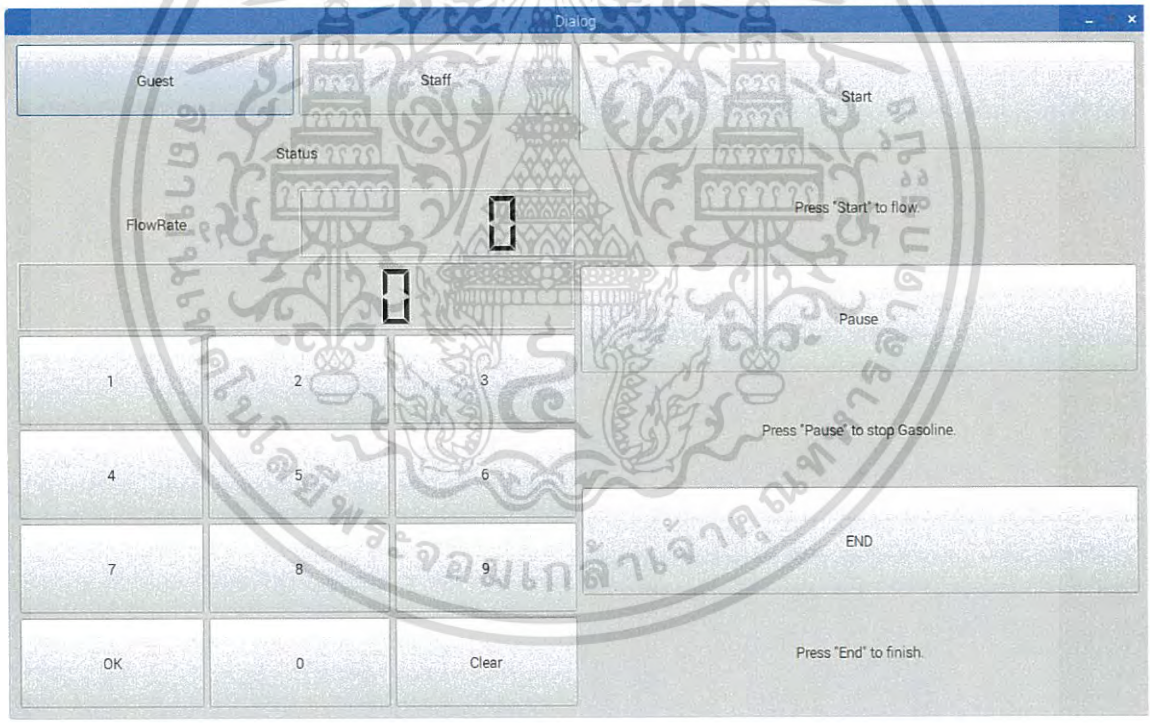
บทที่ 4

ผลการทดลอง

ในบทนี้กล่าวถึงผลการทดลองของอุปกรณ์แต่ละชิ้นในระบบ รวมถึงอธิบายภาพรวมการทำงานตามลำดับขั้นตอน โดยเริ่มจาก User Interface และสิ้นสุดที่ Flow sensor

4.1 การสร้าง User Interface

ผู้จัดทำได้สร้าง User Interface บน Qt Creator ซึ่งเป็นส่วนที่ใช้ในการสื่อสารกับผู้ใช้งานและ Arduino ซึ่งใช้ในการรับคำสั่งจากผู้ใช้ดังรูปที่ 4.1



รูปที่ 4.1 Graphic User Interface บนจอ touch screen

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 การทดสอบการทำงานของ NFC module

ทำการทดสอบการทำงานของ NFC module โดยใช้การเชื่อมต่อแบบ SPI (Serial Peripheral Interface) กับบอร์ด Arduino Mega2560 rev3 เพื่อระบุไอดีของบัตรแต่ละใบ ดังรูปที่ 4.2 เพื่อใช้ในการระบุตัวตนของผู้ใช้งานหรือผู้ควบคุมดูแล

```
UID Value: B96FECAF
185111236175
```

รูปที่ 4.2 การทดสอบ NFC module

4.3 การทดสอบการทำงานของเครื่องหยอดเหรียญ (Vending Machine)

ผู้จัดทำได้ทำการทดสอบการทำงานของเครื่องหยอดเหรียญ โดยใช้บอร์ด Arduino เป็นตัวสั่งการคำสั่ง เปิด/ปิด และรับข้อมูลชุดรหัสของเหรียญแต่ละเหรียญ ในการตรวจสอบสถานะของเครื่องหยอดเหรียญ โดยการส่งคำสั่ง Enable ซึ่งเป็นเลขฐาน 16 ไปยังเครื่องรับเหรียญ จะมีการตอบ Ack กลับมาในรูปแบบเลขฐาน 16 จะทำให้เครื่องหยอดเหรียญแสดงไฟสีเขียวซึ่งหมายถึงสถานะพร้อมใช้งาน(Enable)ดังรูปที่ 4.3

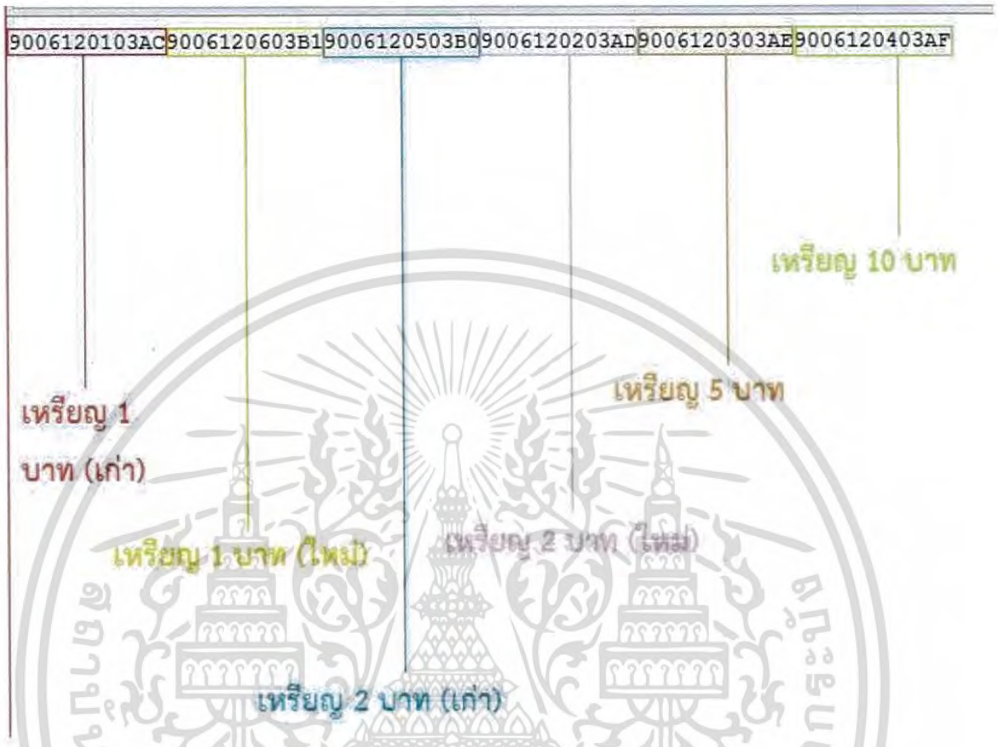
จากนั้นทำการส่งชุดคำสั่ง Disable ไปยังเครื่องหยอดเหรียญอีกครั้ง จะได้รับการ Ack กลับมาในรูปแบบเลขฐาน 16 และเครื่องหยอดเหรียญจะแสดงไฟสีแดงซึ่งหมายถึงสถานะไม่พร้อมใช้งาน ดังรูปที่ 4.4

รูปที่ 4.3 เครื่องหยอดเหรียญแสดงสถานะไฟสีเขียวพร้อมใช้งาน

รูปที่ 4.4 เครื่องหยอดเหรียญแสดงสถานะไฟสีแดงไม่พร้อมใช้งาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลำดับถัดมาได้ทดสอบการอ่านค่าแต่ละเหรียญ ได้แก่เหรียญ 1 บาท(เก่า) 1บาท(ใหม่) 2 บาท(เก่า) 2 บาท(ใหม่) 5 บาท และ 10 บาทตามลำดับได้ผลดังรูปที่ 4.5



รูปที่ 4.5 ค่าของเหรียญแต่ละเหรียญที่อ่านได้

4.4 การทดสอบการทำงานของ Flow Sensor

ทำการทดสอบโดยเชื่อมต่อเซนเซอร์กับบอร์ด Arduino Mega2560 r3 ดังรูปที่ 4.6 จากนั้น เปิดซอฟท์แวร์ Arduino IDE ที่ได้เขียนโค้ดในการคำนวณอัตราการไหลและปริมาตรการไหล ทั้งหมดไว้ เพื่อสังเกตอัตราการไหลและปริมาตรทั้งหมดเมื่อมีการไหลผ่าน Flow sensor

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 4.6 การเชื่อมต่อ Flow sensor กับ Arduino Mega2560 rev3

4.4.1 การคำนวณอัตราการไหลและปริมาตร

ใช้ Arduino รับค่าจากเซนเซอร์แล้วทำการนับพัลส์เพื่อนำมาคำนวณอัตราการไหลตามสมการที่ 2.2 ผ่านฟังก์ชัน pulseCounter() และ measureFlow() ดังรูปที่ 4.7 เพื่อใช้ในการคำนวณปริมาณการไหลทั้งหมด จากนั้นทำการแสดงผลผ่านฟังก์ชัน reportresulf() ดังรูปที่ 4.8 ซึ่งมีการแสดงผลอัตราการไหลในหน่วย L/min(litre/minute) และ mL/s(milliliter/second) และปริมาตรการไหลทั้งหมดในหน่วย L(litre) ดังรูปที่ 4.9

ในส่วนของสมการที่ 2.2 : Q = Frequency / Coefficient

ซึ่งค่า Coefficient ของ Flow sensor ที่ใช้คือ 7.5 Hz per L/min ซึ่งหมายความว่าหากนับลูกคลื่นได้ 7.5 ลูกใน 1 วินาที(ตลอดระยะเวลา 1 นาที) จะแสดงอัตราการไหล 7.5 L/min ซึ่งจุดประสงค์หลักในการทดลองนี้คือการควบคุมปริมาตรการไหล จึงไม่สามารถควบคุมการไหลให้คงที่ได้ตลอดระยะเวลา 1 นาที ทำให้อัตราการไหลเปลี่ยนแปลงตลอดเวลาและไม่สามารถแสดงการคำนวณอัตราการไหลที่แน่นอนได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

mega_sensor

```
#define sensorInterrupt 0
#define sensorPin 2
const float coeff = 7.5;
#define stepTime 1000
volatile uint16_t pulseCount;
float flowRate;
float flowRateMLS;
float flowCumVol;
unsigned long previousTime;
unsigned long currentTime;
unsigned long delTime;
void setup(){
  Serial.begin(9600);
  Serial.setTimeout(0);
  pinMode(sensorPin, INPUT);
  digitalWrite(sensorPin, HIGH);
  //Serial.println("Flow meter start");
  delay(1000);
  previousTime = 0;
  flowRate = 0;
  flowRateMLS = 0;
  pulseCount = 0;
  attachInterrupt(sensorInterrupt,pulseCounter, FALLING);
}
void loop(){
  currentTime = millis();
  delTime = currentTime-previousTime;
  serialReadCommand();
  if(delTime >= stepTime){
    measureFlow();
  }
}
void measureFlow(){
  detachInterrupt(sensorInterrupt);
  flowRate = (1000 * pulseCount / delTime) / coeff;
  flowRateMLS = flowRate*1000/60;
  flowCumVol += (flowRate/60)*(delTime/1000);
  previousTime =  currentTime;
  reportResulf();
  pulseCount = 0;
  attachInterrupt(sensorInterrupt,pulseCounter, FALLING);
```

กำหนด pin สำหรับอ่านค่า

กำหนดค่า calibration factor (coefficient) ในที่นี้คือ 7.5 Hz/L/min) (จาก datasheet) คือค่าคงที่ของความถี่ที่

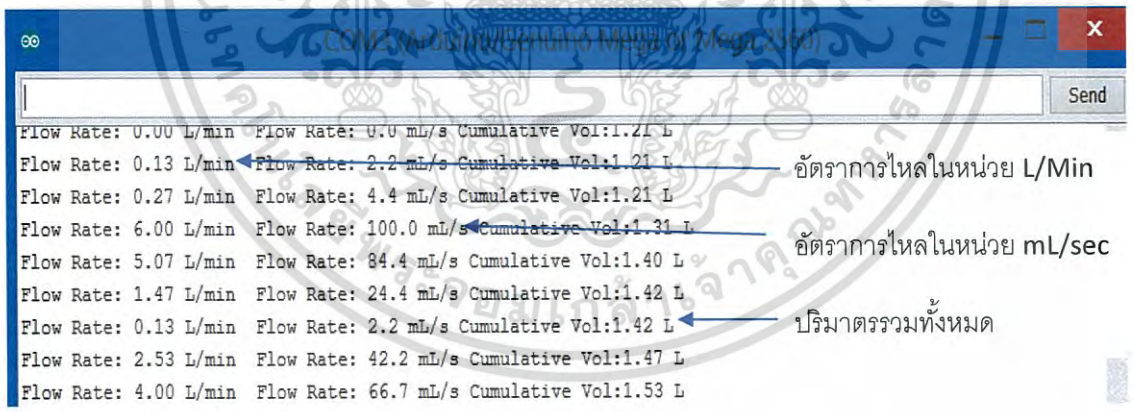
รูปที่ 4.7 โปรแกรมส่วนต้นกำหนดตัวแปร พินต่างๆ และคำนวณพัลส์สัญญาณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

mega_sensor

```
  detachInterrupt(sensorInterrupt);
  flowRate = (1000 * pulseCount / delTime) / coeff;
  flowRateMLS = flowRate*1000/60;
  flowCumVol += (flowRate/60)*(delTime/1000);
  previousTime =  currentTime;
  reportResulf();
  pulseCount = 0;
  attachInterrupt(sensorInterrupt,pulseCounter, FALLING);
}
void pulseCounter(){
  pulseCount++;
}
void reportResulf(){
  Serial.print("Flow Rate: "); Serial.print(flowRate,2);Serial.print(" L/min");
  Serial.print("  Flow Rate: "); Serial.print(flowRateMLS,1);Serial.print(" mL/s");
  Serial.print(" Cumulative Vol:"); Serial.print(flowCumVol,2);Serial.print(" L");
  Serial.println("  ");
  //Serial.print(flowCumVol); Serial.println(",");
}
```

รูปที่ 4.8 โปรแกรมส่วนแสดงผล สัญญาณที่ได้จากเซนเซอร์

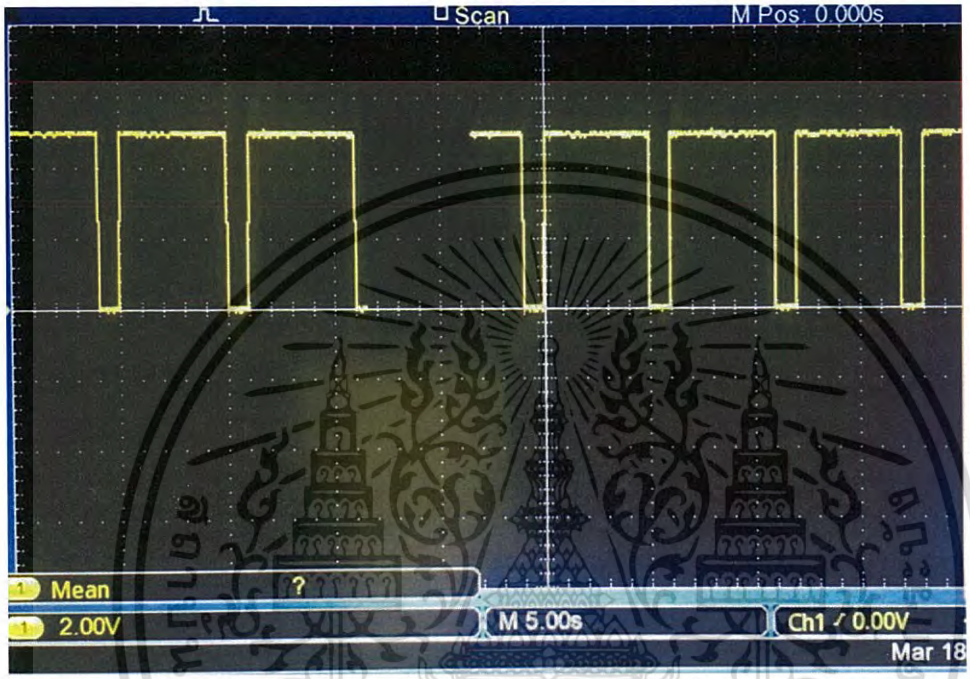


รูปที่ 4.9 การทดสอบวัดอัตราการไหลและปริมาตรโดยใช้ U-900A Flow Sensor

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5 การทดสอบการควบคุมเปิด/ปิด Pump และ Flow Sensor โดยใช้ relay

ใช้ Arduino ควบคุม relay ให้เปิด/ปิด pump และ sensor เป็นจังหวะ จากนั้นใช้ Osilloscope สังเกตขนาดของไฟเลี้ยงจากบอร์ด Arduino ที่ขา 5V



รูปที่ 4.10 ไฟเลี้ยงขนาด 5VDC จากบอร์ด Arduino

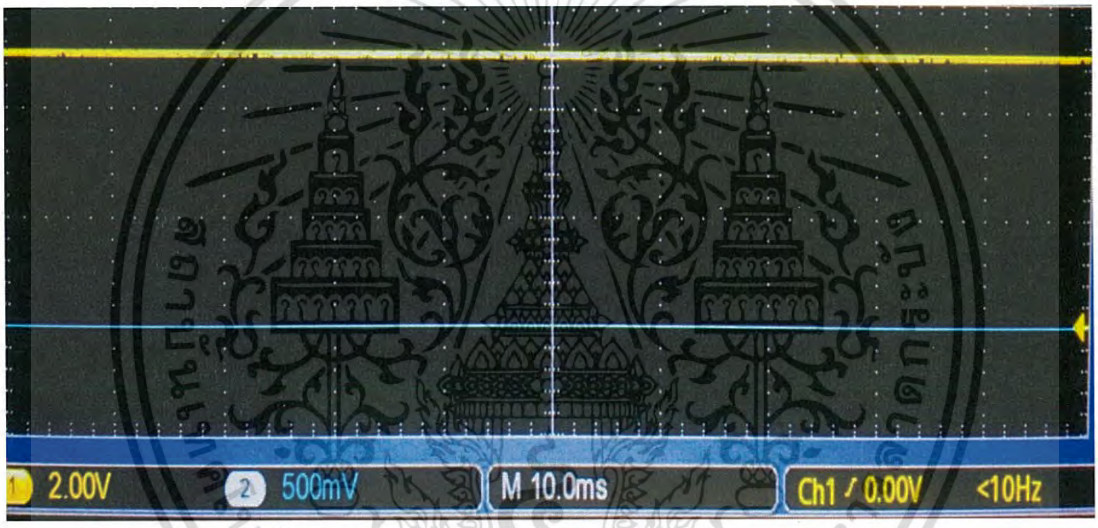
4.6 การรับค่าสัญญาณ PWM จาก Flow sensor

ในขั้นตอนนี้ได้ทำการทดสอบการไหลผ่าน Flow sensor โดยแบ่งระดับการไหลเป็น 3 ระดับ ได้แก่ ไม่มีการไหล ไหลผ่านอย่างเบา และไหลผ่านอย่างรวดเร็ว จากนั้นทำการใช้ Oscilloscope วัดค่าสัญญาณและสังเกตการเปลี่ยนแปลงของลูกคลื่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

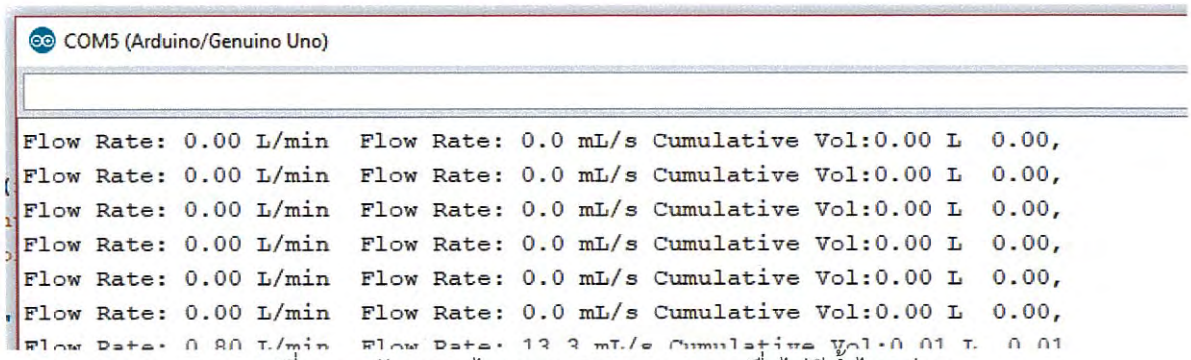
4.6.1 สัญญาณ PWM จาก Flow sensor เมื่อไม่มีการไหล

เมื่อใช้ Oscilloscope วัดค่าสัญญาณพบว่าสัญญาณคงที่ ที่ 5VDC และไม่มีพัลส์เกิดขึ้นดังรูปที่ 4.11 จากนั้นทำการป้อนสัญญาณ PWM จาก Flow sensor ไปยังขา PWM บนบอร์ด Arduino เพื่อนำไปประมวลผลโดยโค้ดที่ได้เขียนขึ้นบนซอฟต์แวร์ Arduino IDE ได้ผลการทดลองบน Oscilloscope ดังรูปที่ 4.11 และผลบน Serial monitor ดังรูปที่ 4.12



รูปที่ 4.11 สัญญาณ PWM จาก Flow sensor เมื่อไม่มีน้ำไหลผ่าน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

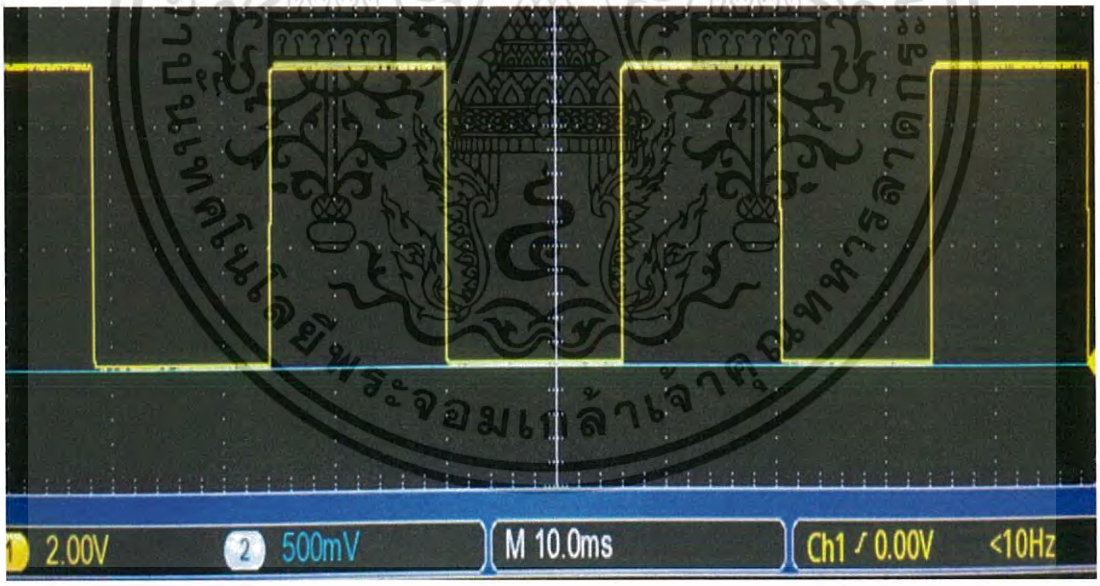


```
Flow Rate: 0.00 L/min  Flow Rate: 0.0 mL/s Cumulative Vol:0.00 L  0.00,
Flow Rate: 0.00 L/min  Flow Rate: 0.0 mL/s Cumulative Vol:0.00 L  0.00,
Flow Rate: 0.00 L/min  Flow Rate: 0.0 mL/s Cumulative Vol:0.00 L  0.00,
Flow Rate: 0.00 L/min  Flow Rate: 0.0 mL/s Cumulative Vol:0.00 L  0.00,
Flow Rate: 0.00 L/min  Flow Rate: 0.0 mL/s Cumulative Vol:0.00 L  0.00,
Flow Rate: 0.00 L/min  Flow Rate: 0.0 mL/s Cumulative Vol:0.00 L  0.00,
Flow Rate: 0.80 L/min  Flow Rate: 13.3 mL/s Cumulative Vol:0.01 L  0.01
```

รูปที่ 4.12 อัตราการไหลจาก Flow sensor เมื่อไม่มีน้ำไหลผ่าน

4.6.2 สัญญาณ PWM จาก Flow sensor เมื่อมีการไหลอย่างช้า

เมื่อใช้ Oscilloscope วัดค่าสัญญาณพบว่ามีขนาดคงที่ ที่ 5VDC และมีพัลส์เกิดขึ้นดังรูปที่ 4.13 จากนั้นทำการป้อนสัญญาณ PWM จาก Flow sensor ไปยังขา PWM บนบอร์ด Arduino เพื่อนำไปประมวลผลโดยโค้ดที่ได้เขียนขึ้นบนซอฟต์แวร์ Arduino IDE ได้ผลการทดลองบน Oscilloscope ดังรูปที่ 4.13 และผลบน Serial monitor ดังรูปที่ 4.14



รูปที่ 4.13 สัญญาณ PWM จาก Flow sensor เมื่อน้ำไหลผ่านอย่างช้า

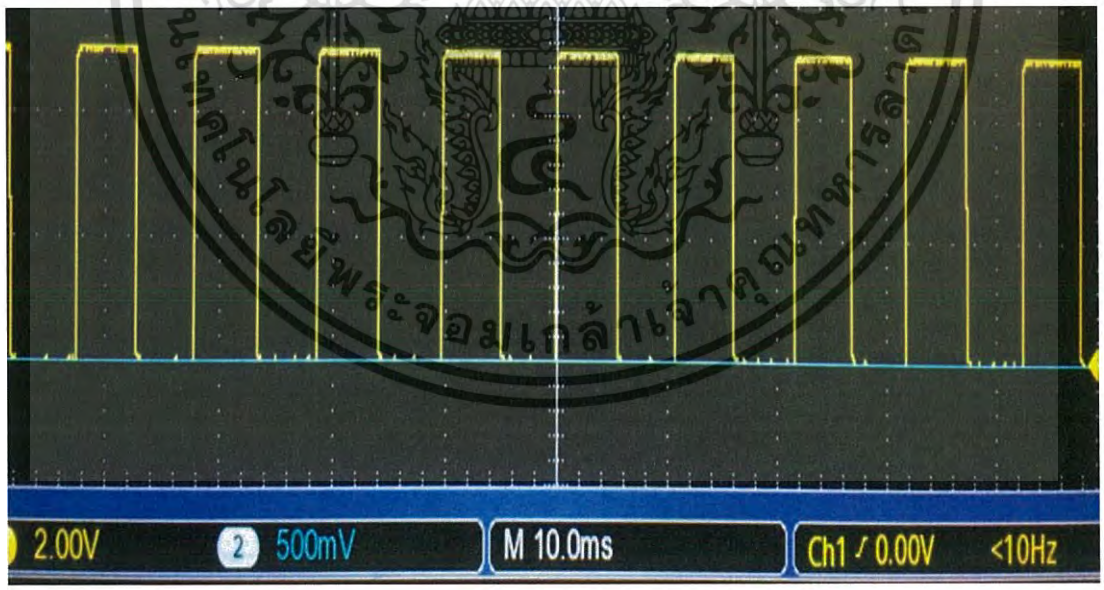
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Flow Rate: 0.00 L/min   Flow Rate: 0.0 mL/s Cumulative Vol:0.00 L   0.00,
Flow Rate: 0.80 L/min   Flow Rate: 13.3 mL/s Cumulative Vol:0.01 L   0.01,
Flow Rate: 1.33 L/min   Flow Rate: 22.2 mL/s Cumulative Vol:0.04 L   0.04,
Flow Rate: 0.00 L/min   Flow Rate: 0.0 mL/s Cumulative Vol:0.04 L   0.04,
```

รูปที่ 4.14 อัตราการไหลจาก Flow sensor เมื่อน้ำไหลผ่านอย่างช้า

4.6.2 สัญญาณ PWM จาก Flow sensor เมื่อมีการไหลอย่างรวดเร็ว

เมื่อใช้ Oscilloscope วัดค่าสัญญาณพบว่ามีขนาดคงที่ ที่ 5VDC และมีพัลส์เกิดขึ้นดังรูปที่ 4.15 และพบว่าความกว้างพัลส์ลดลงจากในกรณีที่มีการไหลอย่างช้า จากนั้นทำการป้อนสัญญาณ PWM จาก Flow sensor ไปยังขา PWM บนบอร์ด Arduino เพื่อนำไปประมวลผลโดยโค้ดที่ได้เขียนขึ้นบนซอฟต์แวร์ Arduino IDE ได้ผลการทดลองบน Oscilloscope ดังรูปที่ 4.15 และผลบน Serial monitor ดังรูปที่ 4.16



รูปที่ 4.15 สัญญาณ PWM จาก Flow sensor เมื่อน้ำไหลผ่านอย่างรวดเร็ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Flow Rate: 0.80 L/min  Flow Rate: 13.3 mL/s Cumulative Vol:1.70 L  1.70,
Flow Rate: 4.67 L/min  Flow Rate: 77.8 mL/s Cumulative Vol:1.78 L  1.78,
Flow Rate: 0.27 L/min  Flow Rate: 4.4 mL/s Cumulative Vol:1.79 L  1.79,
```

รูปที่ 4.16 อัตราการไหลจาก Flow sensor เมื่อน้ำไหลผ่านอย่างรวดเร็ว

4.7 การทดสอบทำงานโดยรวมของระบบ

4.7.1 ผลทดสอบของ Arduino

ทำการเชื่อมต่ออุปกรณ์ทั้งหมดเข้าด้วยกัน โดยใช้ Arduino Mega2560 r3 เป็นหน่วยประมวลผล รับคำสั่งจากผู้ใช้ และควบคุม ซึ่งประกอบด้วยส่วนการระบุตัวตน ส่วนการอ่านค่าเหรียญและส่วนการควบคุมการไหล ตามลำดับ ดังรูปที่ 4.17

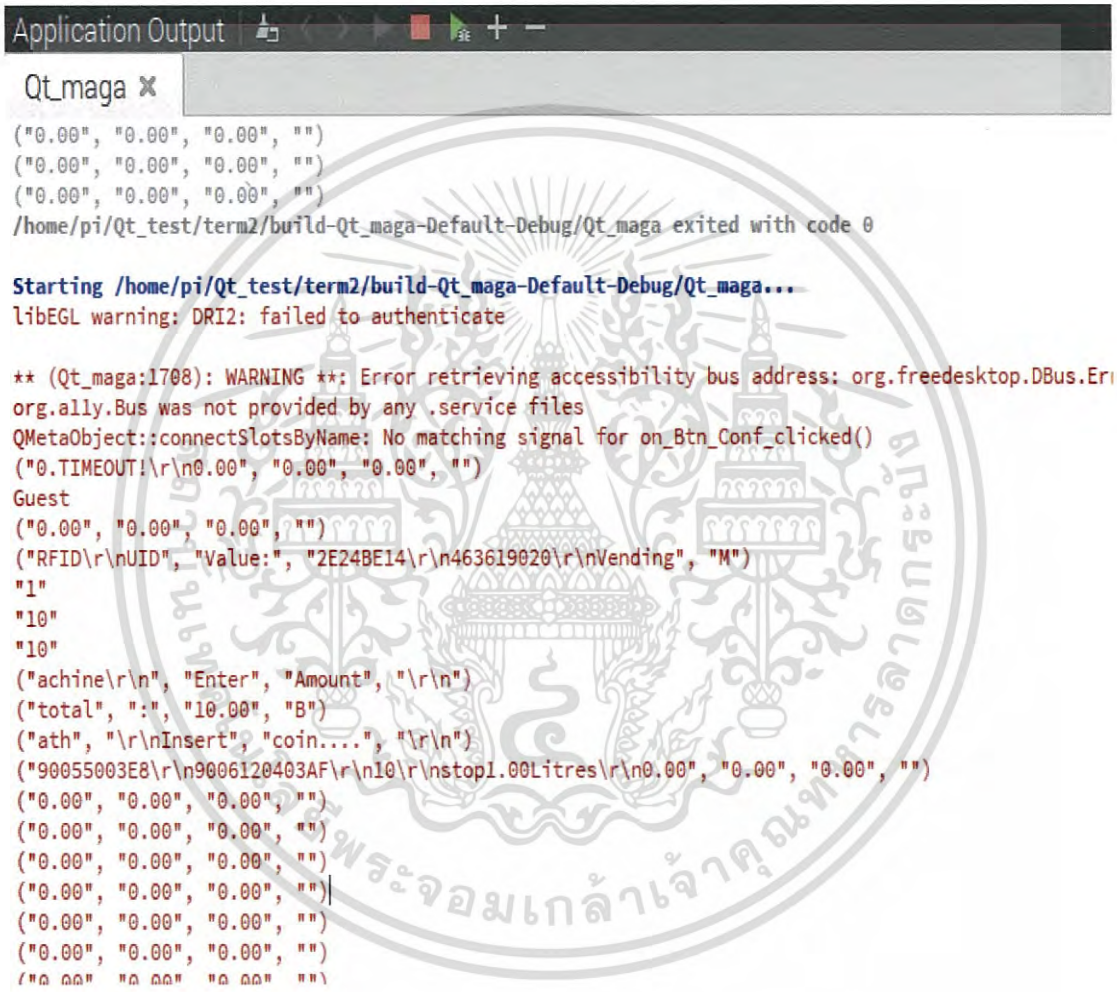
```
UID Value: 2E24BE14
463619020
Vending Machine
9006120403AF9006120403AF9006120403AF90055003E8
stop0.00Litres
0.00 0.00 0.00 0.00 0.00 0.00 0.00 RFID
UID Value: 2E24BE14
463619020
Vending Machine
90055003E890055003E8
stop0.00Litres
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 stop
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

รูปที่ 4.17 ผลการทดสอบด้วย Arduino IDE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.7.2 ผลทดสอบของ Qt creator

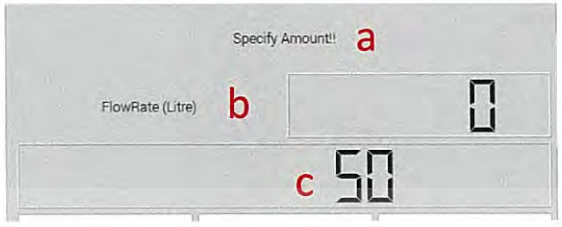
ส่วนของโปรแกรมที่ใช้รับค่าจาก Arduino Mega2560 r3 เพื่อนำไปแสดงผลบนหน้าต่างผู้ใช้ ซึ่งค่าที่ได้มีความสอดคล้องกับค่าบน Serial monitor ของซอฟต์แวร์ Arduino IDE ดังรูปที่ 4.18



รูปที่ 4.18 ผลการทดสอบด้วย Qt creator

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.7.3 ผลการทดสอบจอมอนิเตอร์เมื่อมีการทำงาน



รูปที่ 4.19 ส่วนแสดงผลบนจอมอนิเตอร์

a แสดงถึงสถานะการทำงาน ซึ่งประกอบด้วย Status!! ,Specify Amount!! ,Scan Card!! ,Refilling ,Finished Thank you

b แสดงถึงปริมาตรรวมการไหล

c แสดงถึงยอดเงินที่ผู้ใช้ต้องการเติม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

สรุปผลและข้อเสนอแนะ

5.1 สรุปผล

ปริญญานิพนธ์นี้เป็นการออกแบบและสร้างระบบตู้จ่ายน้ำมันริการตนเอง (Self-service Gasoline Station) โดยใช้บอร์ด Arduino Mega2560 rev3 เป็นตัวควบคุมและสั่งการอุปกรณ์ต่างๆ สามารถแบ่งการทำงานออกเป็น 3 ส่วน ได้แก่ ส่วนคำนวณเงินแบบหยอดเหรียญ ส่วนสัญญาณของการไหล และการไหล ต่อมาเป็นส่วนจอแสดงผลที่ติดสื่อสารกับผู้ใช้บริการ โดยควบคุมการทำงานด้วยระบบจอสัมผัส ด้วยโปรแกรมคิวต์ทำงานใน Raspberry Pi การสื่อสารกับผู้ใช้นั้นมี 3 ส่วนได้แก่ ส่วนระบุตัวตนผ่าน RFID ส่วนระบุปริมาณและควบคุมการไหลจนจบการทำงาน แล้วเมื่อจบการทำงานสามารถเริ่มอีกครั้งได้ทันที

5.2 ข้อเสนอแนะ

โครงงานระบบตู้จ่ายน้ำมันบริการตนเองยังมีความไม่สมบูรณ์ในหลายส่วน และในบางส่วนสามารถพัฒนาได้อีก จะทำให้สามารถใช้งานได้หลากหลายมากขึ้น ดังนี้

1. เพิ่มช่องทางการชำระเงินด้วยธนบัตรเครดิตการ์ด
2. ใช้ปั๊มสูบที่เหมาะในการสูบน้ำมันเบนซิล
3. ควรมีหัวจ่ายที่เหมาะสมในการจ่ายน้ำมัน
4. ตู้หยอดสามารถทอนเงินได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

[1] "Arduino Mega2560 rev3"
https://www.researchgate.net/figure/Arduino-MEGA-2560_fig1_281538436
https://dspace.cvut.cz/bitstream/handle/10467/65602/F2-BP-2016-Lexmann-Robert-priloha-4-ArduinoMega2560.pdf?sequence=4

[2] "Raspberry Pi 3 model b"
https://www.arduitronics.com/product/1243/raspberry-pi-3-model-b-1gb-%E0%B8%88%E0%B8%B2%E0%B8%81-element-14

[3] " Flow sensor"
https://www.gravitechthai.com/product_detail.php?d=900

[4] " Touch Screen "
https://www.waveshare.com/10.1inch-hdmi-lcd-b-with-case.htm

[5] "Arduino กับการใช้งาน Interrupt"
https://www.ioxhop.com/article/23/arduino-%E0%B8%81%E0%B8%B1%E0%B8%9A%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B9%83%E0%B8%8A%E0%B9%89%E0%B8%87%E0%B8%B2%E0%B8%99%E0%B8%AD%E0%B8%B4%E0%B8%99%E0%B9%80%E0%B8%95%E0%B8%AD%E0%B8%A3%E0%B9%8C%E0%B8%A3%E0%B8%B1%E0%B8%9E%E0%B8%97%E0%B9%8C

[6] " Arduino PWM"
http://naringroup.blogspot.com/2016/03/arduino-pwm.html

[7] "การใช้งาน Arduino กับ relay"
http://www.thaieasyelec.com/article-wiki/review-product-article/ตัวอย่างการใช้งาน-arduino-relay-module-ควบคุมการปิดเปิดเครื่องใช้ไฟฟ้า.html

[8] " Qt Create "
https://wiki.qt.io/About_Qt

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

โค้ดควบคุมการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนการกำหนดตัวแปรและเครื่องมือที่ใช้

```
String content = "";

int total = 0;

float total_max;

int LED = 42;

int relay = 44;

#include <Wire.h>

#include <SPI.h>

#include <Adafruit_PN532.h>

#define PN532_SCK (52) //13

#define PN532_MOSI (51) //11

#define PN532_SS (53) //10

#define PN532_MISO (50) //12

//#define PN532_IRQ (2)

//#define PN532_RESET (3)

Adafruit_PN532 nfc(PN532_SCK, PN532_MISO, PN532_MOSI, PN532_SS);

#if defined(ARDUINO_ARCH_SAMD)

#define Serial SerialUSB

#endif

String uidString;

#define sensorInterrupt 0
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#define sensorPin 2

const float coeff = 7.5;

#define stepTime 1000

volatile uint16_t pulseCount;

float flowRate;

float flowRateMLS;

float flowCumVol;

unsigned long previousTime;

unsigned long currentTime;

unsigned long delTime;

float oil_rate = 10;

float total_oil;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนการตั้งค่า

```
void setup() {
  Serial.begin(9600);
  Serial1.begin(9600);
  pinMode(LED, OUTPUT);
  pinMode(relay, OUTPUT);
#ifndef ESP8266
#endif
  nfc.begin();
  uint32_t versiondata = nfc.getFirmwareVersion();
  if (! versiondata) {
    //Serial.print("Didn't find PN53x board");
    while (1);
}
  nfc.setPassiveActivationRetries(0xFF);
  nfc.SAMConfig();
  Serial.setTimeout(2000);
  pinMode(sensorPin, INPUT);
  digitalWrite(sensorPin, HIGH);
  delay(1000);
  previousTime = 0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  flowRate = 0;
  flowRateMLS = 0;
  pulseCount = 0;
  attachInterrupt(sensorInterrupt, pulseCounter, FALLING);
}
void loop() {
  if (Serial.available() > 0) {
    char input = Serial.read();
    if (input == '2') {
      member_card();
    }
    if (input == '1'){
      digitalWrite(relay ,HIGH);
    }
    if (input == '0'){
      digitalWrite(relay ,LOW);delay(10000);
 previousTime = 0; flowRate = 0; flowRateMLS = 0; pulseCount = 0; flowCumVol = 0;
    }
  }
  currentTime = millis();
  delTime = currentTime - previousTime;
```

-การสั่งเปิด/ปิดปั๊ม
(ไม่สามารถเปิดปั๊มได้หากไม่ผ่านการคำนวณปริมาณน้ำมันที่ลูปการจ่ายเงิน)
หากมีมีการปิดปั๊มก่อนครบปริมาณที่ต้องจ่ายระบบจะทำการ reset ภายใน 10 วินาที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  if (delTime >= stepTime) {
    measureFlow();
  }
}
void measureFlow() {
  detachInterrupt(sensorInterrupt);
  flowRate = (1000 * pulseCount / delTime) / coeff;
  flowRateMLS = flowRate * 1000 / 60;
  flowCumVol += (flowRate / 60) * (delTime / 1000);
  previousTime =  currentTime;
  reportResulf();
  pulseCount = 0;
  attachInterrupt(sensorInterrupt, pulseCounter, FALLING);
}
void pulseCounter() {
  pulseCount++;
}

 void reportResulf() {
 Serial.print(flowCumVol); Serial.print(",");
```

ลูปการแสดงผลจาก flow sensor เมื่อปริมาณน้ำมันที่ไหลผ่าน sensor ครบตามจำนวน ปั๊มจะถูกปิดโดยอัตโนมัติ และทำการ reset เพื่อใช้อ่านค่ารอบต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 if (flowCumVol > total_oil) {
   Serial.println("stop");
   digitalWrite(relay, LOW);
   previousTime = 0; flowRate = 0; flowRateMLS = 0; pulseCount = 0; flowCumVol =
0;
 }
}
int money_coin( ) {
  byte enable[] = {0x90, 0x05, 0x01, 0x03, 0x99};
  Serial1.write(enable, sizeof(enable));
  if (!Serial.available() > 0) {
    Serial.println(" Enter Amount ");
    String total_value = "";
    while (!Serial.available());
    while (Serial.available()) {
      char a = Serial.read();
      total_value.concat(a);
      delay(10);
    }
    total_max = total_value.toInt();
    Serial.print("total : ");
```

ลูปการจ่ายเงินด้วยเหรียญ โดยรับค่าที่ได้จากเครื่องหยอดเหรียญแล้วเก็บเป็น content เพื่อใช้เทียบค่าเหรียญ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  Serial.print(total_max);
  Serial.println(" Bath ");
  Serial.println("Insert coin.... ");
}
while (1) {
  if (Serial1.available()) {
    content = "";
    while (Serial1.available()) {
    byte coinIn = Serial1.read();
    content.concat(String(coinIn < 0x10 ? "0" : ""));
    content.concat(String(coinIn, HEX));
    delay(10);
    } content.toUpperCase();
    Serial.println(content);
if (content == "9006120603B1" || content == "9006120103AC") {
      content = "";
      total += 1;
      Serial.println(total);
    } else if (content == "9006120503B2" || content == "9006120203AD" || content ==
            "9006120503B0") {
      content = "";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  total += 2;
  Serial.println(total);
} else if (content == "9006120303AE") {
  content = "";
  total += 5;
  Serial.println(total);
} else if (content == "9006120403AF") {
  content = "";
  total += 10;
  Serial.println(total);
} if (total >= total_max) {
  Serial.print("stop");
  byte disable[] = {0x90, 0x05, 0x02, 0x03, 0x9A};
  Serial1.write(disable, sizeof(disable));
  digitalWrite(LED, HIGH);
  delay(500);
  total = 0;
  total_oil = (total_max / oil_rate);
  Serial.print(total_oil , 2); Serial.println("Litres");
  reportResulf();
  break;
```

การคำนวณปริมาณน้ำมันที่จ่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }
   }
  }
}
int member_card() {
  Serial.println("RFID");
  boolean success;
  uint8_t uid[] = { 0, 0, 0, 0, 0, 0, 0 };
  uint8_t uidLength;
  success = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A, &uid[0], &uidLength);
  if (success) {
    Serial.print("UID Value: ");
    for (uint8_t i = 0; i < uidLength; i++)
    {
      //Serial.print(" 0x");
      Serial.print(uid[i], HEX);
    }
    Serial.println("");
    delay(1000);
    uidString = String(uid[0]) + String(uid[1]) + String(uid[2]) + String(uid[3]);
    Serial.print(uidString);
```

ลูปการระบุตัวตนของบัตรเพื่อแยกระหว่างพนักงานและลูกค้า โดยนำค่าที่ได้จัดเก็บเป็น string เพื่อใช้เทียบค่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  Serial.println("");
  delay(500);
  if (uidString == "4202247210") {
    Serial.println("config");
      rate();
    }
    else  {
      money_coin();
    }
 }
}
int rate() {
  Serial.println("setting");
  if (!Serial.available() > 0) {
    Serial.println("Enter Oil rate");
    String total_value = "";
    while (!Serial.available());
    while (Serial.available()) {
      char a = Serial.read();
      total_value.concat(a);
      delay(10); }
```

ไอดีจากบัตรพนักงานที่ใช้ในการตั้งค่าราคาน้ำมัน

หากเป็นไอดีลูกค้าก็จะเข้าไปยังลูปการจ่ายเงินด้วยเหรียญ

ใช้กำหนดราคาน้ำมัน สามารถเปลี่ยนแปลงได้ผ่านหน้าจอ Interface

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    oil_rate = total_value.toInt();
    Serial.print(oil_rate);
    Serial.println(" Bath ");
  }return;
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนของการออกแบบหน้าจอแสดงผลแบบสัมผัส

ส่วนของการเชื่อมด้วย serialport

```
#-------------------------------------------------

# Project created by QtCreator 2018-02-13T21:58:54

#-------------------------------------------------

QT       += core gui serialport

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

TARGET = Qt_maga

TEMPLATE = app

# The following define makes your compiler emit warnings if you use

# any feature of Qt which as been marked as deprecated (the exact warnings

# depend on your compiler). Please consult the documentation of the

# deprecated API in order to know how to port your code away from it.

DEFINES += QT_DEPRECATED_WARNINGS

# You can also make your code fail to compile if you use deprecated APIs.

# In order to do so, uncomment the following line.

# You can also select to disable deprecated APIs only up to a certain version of Qt.

#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000

# disables all the APIs deprecated before Qt 6.0.0

SOURCES += main.cpp\

        dialog.cpp \
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    numberkeybtn.cpp \
    confdialog.cpp \
    numpadconf.cpp
HEADERS  += dialog.h \
    numberkeybtn.h \
    confdialog.h \
    numpadconf.h
FORMS    += dialog.ui \
    confdialog.ui
#QMAKE_CXXFLAGS += -std=gnu++14
```

ส่วนของการกำหนดตัวแปร

```
#ifndef DIALOG_H
#define DIALOG_H
#include <QDialog>
#include <QSerialPort>
#include <QSerialPortInfo>
#include <QObject>
#include <QPushButton>
//#include "confdialog.h"
namespace Ui {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
class Dialog;

}


class Dialog : public QDialog

{

    Q_OBJECT


public:

    explicit Dialog(QWidget *parent = 0);

    ~Dialog();

private slots:

    void on_Btn_Guest_clicked();

    void on_Btn_Member_clicked();

    void on_Btn_OK_clicked();

    void on_Btn_Clear_clicked();

    void onKeyWasClicked(const QString &number);

    void readSerial();

    void updateFlowRate(QString);

    void on_Btn_Conf_clicked();

    void on_Btn_Pause_clicked();

    //void nnew(QString serv);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    void on_Btn_Strat_clicked();

private:
    void connectKey();
    Ui::Dialog *ui;
    QSerialPort *arduino;
    QString arduino_port_name;
    QString money;
    QString serialBuffer;
    QByteArray serialData;
    QByteArray coin;
    bool arduino_is_available;
    static const quint16 arduino_mega_vendor_id = 9025;
    static const quint16 arduino_mega_product_id = 66;
};

#endif // DIALOG_H
```

ส่วนของฟังค์ชันการทำงาน

```
#include "dialog.h"
#include "ui_dialog.h"
#include <QSerialPort>
#include <QSerialPortInfo>
#include <QDebug>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#include <QMessageBox>
#include <QtWidgets>
#include <QObject>
#include "confdialog.h"
#include "ui_confdialog.h"
#define PASSWORDCONFIG 123456
Dialog::Dialog(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::Dialog)
{
    ui->setupUi(this);
    ui->Btn_Clear->setMinimumSize(100,100);
    ui->Btn_OK->setMinimumSize(100,100);
    connectKey();
    serialBuffer ="";
    arduino_is_available = false;
    arduino_port_name = "";
    arduino = new QSerialPort(this);
    /*qDebug() <<"Number of available port:
"<<QSerialPortInfo::availablePorts().length();
    foreach (const QSerialPortInfo &serialPortInfo, QSerialPortInfo::availablePorts()) {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    qDebug() <<"Has vendor ID: "<< serialPortInfo.hasVendorIdentifier();
    if(serialPortInfo.hasVendorIdentifier()){
        qDebug() <<"Vendor ID: "<< serialPortInfo.vendorIdentifier();
    }
    qDebug() <<"Has Product ID: "<< serialPortInfo.hasProductIdentifier();
    if(serialPortInfo.hasProductIdentifier()){
        qDebug() <<"Product ID: "<< serialPortInfo.productIdentifier();
    }
}*/
/*----------------------*/
foreach (const QSerialPortInfo &serialPortInfo, QSerialPortInfo::availablePorts()){
    if(serialPortInfo.hasVendorIdentifier() && serialPortInfo.hasProductIdentifier()){
        if(serialPortInfo.vendorIdentifier() == arduino_mega_vendor_id){
            if(serialPortInfo.productIdentifier() == arduino_mega_product_id){
                arduino_port_name = serialPortInfo.portName();
                arduino_is_available = true;
            }
        }
    }
}
/*-------------------------*/
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(arduino_is_available){
        //open and configure the serialport
        arduino->setPortName(arduino_port_name);
        arduino->open(QSerialPort::ReadWrite);
        arduino->setBaudRate(QSerialPort::Baud9600);
        arduino->setDataBits(QSerialPort::Data8);
        arduino->setParity(QSerialPort::NoParity);
        arduino->setStopBits(QSerialPort::OneStop);
        arduino->setFlowControl(QSerialPort::NoFlowControl);
        QObject::connect(arduino, SIGNAL(readyRead()),this, SLOT(readSerial()));
        //QObject::connect(arduino, SIGNAL(readyRead()),this, SLOT(readSerial00());
    }else{
        //give error message if not available
        QMessageBox::warning(this, "Port error", "Couldn't find the Arduino!");
    }
}
Dialog::~Dialog()
{
    delete ui;
}
void Dialog::readSerial()
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
//    serialData = arduino->readAll();
//    serialBuffer += QString::fromStdString(serialData.toStdString());
//    qDebug() << serialBuffer;
    QStringList bufferSplit = serialBuffer.split((" "));
    if(bufferSplit.length() <=3 ){
        serialData = arduino->readAll();//
        serialBuffer += QString::fromStdString(serialData.toStdString());
    }else{
        //bufferSplit[1] is a good value
        qDebug() << bufferSplit;
        Dialog::updateFlowRate(bufferSplit[1]);
        serialBuffer = "";
    }
}
void Dialog::on_Btn_Guest_clicked()
{
    qDebug("Guest");
    ui->Status_label->setText(QString("Specify Amount!!"));
    if(arduino->isWritable()){
        arduino->write("2");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }else{
        qDebug() << "Cloudn't write to serial!";
    }
}
void Dialog::on_Btn_Member_clicked()
{
    qDebug("Staff");
    ui->Status_label->setText(QString("Scan Card!!"));
    if(arduino->isWritable()){
        arduino->write("2");
    }else{
        qDebug() << "Cloudn't write to serial!";
    }
}
void Dialog::onKeyWasClicked(const QString &number)
{
    ui->Value_lcd->display
    (QString::number(ui->Value_lcd->value())+number);
    money += number;
    qDebug() << money;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void Dialog::on_Btn_OK_clicked()
{
    //setMinimumSize(100,100);
    arduino->write(money.toUtf8());
    qDebug() << money;
    money = "";
}
void Dialog::connectKey()
{
    connect(ui->Btn_0,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
    connect(ui->Btn_1,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
    connect(ui->Btn_2,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
    connect(ui->Btn_3,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
    connect(ui->Btn_4,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
    connect(ui->Btn_5,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    connect(ui->Btn_6,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
    connect(ui->Btn_7,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
    connect(ui->Btn_8,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
    connect(ui->Btn_9,&NumberKeyBtn::keyWasClicked,
            this, &Dialog::onKeyWasClicked);
}
void Dialog::on_Btn_Clear_clicked()
{
    //setMinimumSize(100,100);
    ui->Value_lcd->display(0);
    money = "";
    qDebug("Clear");
}
void Dialog::updateFlowRate(QString sensor_reading)
{
    // update the value displayed on the lcdNumber
    ui->lcdNumber->display(sensor_reading);
    const double SensorReading = sensor_reading.toDouble();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(SensorReading == 0.00){
        ui->Status_label->setText("Status!!");
    }else{return;}
}
void Dialog::on_Btn_Pause_clicked()
{
    qDebug("Pause");
    ui->Status_label->setText(QString("Stop"));
    if(arduino->isWritable()){
        arduino->write("0");
    }else{
        qDebug() << "Cloudn't write to serial!";
    }
}
/*
void Dialog::on_Btn_END_clicked()
{
    qDebug("END");
}
*/
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//void nnew::setDestination(QString serv)
//{
//    serv="";
//}
void Dialog::on_Btn_Conf_clicked()
{
//    const double password = money.toDouble();
//    if(password == PASSWORDCONFIG)
//    {
//        arduino->write("3");
//        qDebug("ok");
//        ui->Status_label->setText("Price per Litre");
//    }else{
//        qDebug("no");
//        return;
//    }
    //reject();
    ConfDialog confDeialog;
    confDeialog.setModal(true);
    confDeialog.exec();
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void Dialog::on_Btn_Strat_clicked()
{
    qDebug("start");
    ui->Status_label->setText(QString("Start"));
    if(arduino->isWritable()){
        arduino->write("1");
    }else{
        qDebug() << "Cloudn't write to serial!";
    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้