



รายงาน วิชา 240-319 Embedded System Module

หัวข้อ Piggy Bank Saving Your Life

โดย

นายจักรวาล คำมนี รหัสนักศึกษา 6510110061

นายชนาริป สุขพรสววรค์ รหัสนักศึกษา 6510110091

นายนฤเบศร ไสพรณรงค์ รหัสนักศึกษา 6510110230

Section 01

เสนอ

รศ.ดร.ทวีศักดิ์ เรืองพีระกุล

รศ.ดร.ปัญญา ไชยการ

ผศ.ดร.วชรินทร์ แก้วอภิชัย

สาขาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

ภาคเรียนที่ 1 ปีการศึกษา 2567

คำนำ

ในปัจจุบัน เทคโนโลยีได้เข้ามามีบทบาทสำคัญในทุกแง่มุมของการดำเนินชีวิต ไม่ว่าจะเป็นการทำงาน การศึกษา การเดินทาง หรือแม้กระทั่งการเก็บออม ความก้าวหน้าทางเทคโนโลยีไม่เพียงแต่ซ่อนเร้นความสะดวกสบาย แต่ยังเป็นแรงผลักดันในการปรับเปลี่ยนพฤติกรรมการใช้ชีวิตของมนุษย์ ในขณะที่เทคโนโลยี การเงินและ IoT กำลังเข้ามาทดแทนวิถีชีวิตดั้งเดิม หลายสิ่งที่เคยเป็นส่วนหนึ่งของชีวิตประจำวันจึงได้รับการปรับเปลี่ยนให้เข้ากับยุคสมัย รวมถึง "ระบบปุกออมสิน" ซึ่งเคยเป็นสัญลักษณ์ของการออมทรัพย์ในอดีต โครงการ "ระบบปุกออมสินอัจฉริยะ" เกิดจากการผสมผสานเทคโนโลยีสมัยใหม่เข้ากับแนวคิดการออมทรัพย์แบบดั้งเดิม โดยนำเทคโนโลยี IoT มาเสริมความสามารถให้ระบบปุกออมสินไม่เพียงแต่เก็บออมเงิน แต่ยังสามารถตรวจสอบคุณภาพอากาศในพื้นที่ที่ตั้งอยู่ได้ ด้วยการติดตั้งเซ็นเซอร์วัดค่าฝุ่น PM 1.0, PM 2.5, และ PM 10 พร้อมทั้งเชื่อมต่อข้อมูลกับแพลตฟอร์ม Blynk, ThingSpeak, Siri ของระบบปฏิบัติการ iOS และ ผ่านแอพ Line Notify เพื่ออำนวยความสะดวกในการติดตามคุณภาพอากาศแบบเรียลไทม์ โครงการนี้จึงไม่เพียงเป็นการนำอุปกรณ์ดั้งเดิมมาปรับปรุงให้ทันสมัย แต่ยังตอบสนองต่อปัญหามลภาวะทางอากาศที่ผู้คนเผชิญอยู่ ในปัจจุบัน ช่วยให้ผู้ใช้งานสามารถออมเงินและตรวจวัดคุณภาพอากาศได้ในอุปกรณ์เดียว ถือเป็นการนำเทคโนโลยีมาช่วยเสริมสร้างชีวิตที่ดีขึ้น ทั้งในด้านการเงินและสุขภาพ

คณะผู้จัดทำคาดหวังเป็นอย่างยิ่งว่า รายงานฉบับนี้จะเป็นประโยชน์ต่อผู้ที่สนใจในการพัฒนาอุปกรณ์ Internet of Things เพื่อใช้ในบ้าน และนำไปพัฒนาต่อยอดในเชิงงาน และผู้ที่สนใจเกี่ยวกับอุปกรณ์ Internet of Things สำหรับใช้ในบ้านเรือน หากรายงานฉบับนี้ขาดตกบกพร่องในส่วนใด ๆ ไป ขออภัยไว้ ณ ที่นี่ด้วย

ผู้จัดทำ

สารบัญ

เรื่อง	หน้า
คำนำ	ก
สารบัญ	ข
สารบัญรูปภาพ	ง
สารบัญตาราง	ช
บทที่ 1 บทนำ	
ที่มาและความสำคัญของปัญหา	1
วัตถุประสงค์	2
สมมติฐาน	2
บทที่ 2 เอกสารและงานวิจัยที่เกี่ยวข้อง	
เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 3	3
เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 9	4
เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 13	5
เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 14	6
เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 15	7
เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 17	11
เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 18	15
เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บท Odroid	16
บทที่ 3 วิธีการดำเนินงาน	
การออกแบบ และรายละเอียด	17
วัสดุอุปกรณ์	17
การออกแบบวงจร	18
Schematic Diagram	18
Wiring	19
วงจรสำหรับการวัดค่าฝุ่น	19
วงจรสำหรับการแยกเหรียญ	20
ขั้นตอนการดำเนินงาน	21
รายละเอียดเกี่ยวกับโค้ดของแต่ละอุปกรณ์ภายใน	32
Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003	32
NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk	32

สารบัญ

เรื่อง	หน้า
Odroid C4 ที่เชื่อมต่อไปยัง ThingSpeak และ Line Notify	33
Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อกับ Keypad	33
สรุปภาพรวมการทำงานของโค้ด	36
State Machine	37
Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003	37
NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk	37
Odroid C4 ที่เชื่อมต่อไปยัง ThingSpeak และ Line Notify	38
Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อกับ Keypad	38
บทที่ 4 ผลการทดลอง	
เพื่อศึกษาการเขียนโปรแกรม และศึกษาการทำงานของบอร์ด Arduino ในการแยกเหรียญ	39
เพื่อศึกษาหาประสิทธิภาพของการส่งข้อมูลpm 2.5 ที่ถูกส่งไปยังอุปกรณ์ต่าง ๆ	42
เพื่อพัฒนาให้อุปกรณ์ Internet of Things สามารถใช้ในบ้าน และชีวิตประจำวันได้	50
บทที่ 5 สรุปผลการทดลอง	
เพื่อศึกษาการเขียนโปรแกรม และศึกษาการทำงานของบอร์ด Arduino ในการแยกเหรียญ	51
เพื่อศึกษาหาประสิทธิภาพของการส่งข้อมูลpm 2.5 ที่ถูกส่งไปยังอุปกรณ์ต่าง ๆ	51
เพื่อพัฒนาให้อุปกรณ์ Internet of Things สามารถใช้ในบ้าน และชีวิตประจำวันได้	52
ประโยชน์ที่คาดว่าจะได้รับ	52
ข้อเสนอแนะ	52
เอกสารอ้างอิง	53

สารบัญรูปภาพ

ภาพที่	หน้า
ภาพที่ 1 เวกเตอร์การขัดจังหวะของ ATMega328P	3
ภาพที่ 2 การตั้งค่า Interrupt ใน Arduino IDE	3
ภาพที่ 3 LCD 16x2 Pinout	4
ภาพที่ 4 Library LiquidCrystal.h	4
ภาพที่ 5 การແນະນຳການເຂື້ອມຕ່ອແບບ Two-Wire	5
ภาพที่ 6 ຮາຍລະເອີດ IC PCF8574 (1)	5
ภาพที่ 7 ຮາຍລະເອີດ IC PCF8574 (2)	6
ภาพที่ 8 ຮາຍລະເອີດ IC DS1307	6
ภาพที่ 9 ກາວະໜັບຂອງຕົວປະມາລຸລ	7
ภาพที่ 10 ຮາຍລະເອີດ Power-Save Mode	7
ภาพที่ 11 Power-Save Mode	8
ภาพที่ 12 ເຮັດວຽກ SMCR	8
ภาพที่ 13 Sleep Mode ແບບຕ່າງ ๆ	9
ภาพที่ 14 ການສ້າງໃຫ້ CPU ເຂົ້າສູ່ກາວະໜັບ	9
ภาพที่ 15 ຕົວຢ່າງຝຶກໜັນໃນການເປີດ/ປິດທາງການເຂົ້າສູ່ກາວະໜັບ (1)	10
ภาพที่ 16 ຕົວຢ່າງຝຶກໜັນໃນການເປີດ/ປິດທາງການເຂົ້າສູ່ກາວະໜັບ (2)	10
ภาพที่ 17 ຮາຍລະເອີດ EEPROM	11
ภาพที่ 18 ມີຄວາມຈຳ EEPROM	11
ภาพที่ 19 ຂາດຂອງ EEPROM ໃນ AVR ແຕ່ລະຕົວ	12
ภาพที่ 20 ຈຳນວນຄັ້ງທີ່ບັນທຶກຂໍ້ມູນລົງ EEPROM ໄດ້	12
ภาพที่ 21 ຮູບແບບການ Write ຂອງ EEPROM	13
ภาพที่ 22 ການ Write ຂອງ EEPROM ແບບ Byte Write	13
ภาพที่ 23 ການ Write ຂອງ EEPROM ແບບ Byte Write	14
ภาพที่ 24 ການ Read ຂອງ EEPROM ແບບ Random Read	14
ภาพที่ 25 NodeMCU Pinout	15
ภาพที่ 26 ແອພັລິເຄັນ Blynk	15
ภาพที่ 27 Odroid C4 Pinout	16
ภาพที่ 28 ThingSpeak Website	16
ภาพที่ 29 Diagram ຂອງຈົກສຳຮັບການວັດຄ່າຜູ້ນ	18

สารบัญรูปภาพ

ภาพที่	หน้า
ภาพที่ 30 Diagram ของวงจรสำหรับการแยกเหรียญ และนาฬิกาแสดงเวลา	18
ภาพที่ 31 Diagram ของวงจรสำหรับการวัดค่าฝุ่น	19
ภาพที่ 32 Diagram ของวงจรสำหรับการแยกเหรียญ และนาฬิกาแสดงเวลา	20
ภาพที่ 33 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003	22
ภาพที่ 34 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003	22
ภาพที่ 35 โค้ดสำหรับอุปกรณ์ NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk	23
ภาพที่ 36 โค้ดสำหรับอุปกรณ์ NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk	24
ภาพที่ 37 โค้ดสำหรับอุปกรณ์ Odroid C4 ที่เชื่อมต่อไปยัง ThingSpeak และ Line Notify	25
ภาพที่ 38 การแสดงค่า PM 1.0 ผ่านโปรแกรม Shortcut	26
ภาพที่ 39 การแสดงค่า PM 2.5 ผ่านโปรแกรม Shortcut	26
ภาพที่ 40 การแสดงค่า PM 10 ผ่านโปรแกรม Shortcut	27
ภาพที่ 41 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad	28
ภาพที่ 42 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad	29
ภาพที่ 43 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad	29
ภาพที่ 44 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad	30
ภาพที่ 45 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad	31
ภาพที่ 46 State Diagram สำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003	37
ภาพที่ 47 State Diagram สำหรับอุปกรณ์ NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk	37
ภาพที่ 48 State Diagram สำหรับอุปกรณ์ Odroid C4 ที่เชื่อมต่อไปยัง ThingSpeak และ Line Notify	38
ภาพที่ 49 Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad	38
ภาพที่ 50 ผลลัพธ์ผ่าน Serial Monitor ของอุปกรณ์ Arduino UNO R3	42
ภาพที่ 51 ผลลัพธ์ผ่าน Serial Monitor ของอุปกรณ์ NodeMCU	45
ภาพที่ 52 ผลลัพธ์ผ่าน Blynk	48

สารบัญรูปภาพ

ภาพที่	หน้า
ภาพที่ 53 ผลลัพธ์ผ่านการ print การรับและส่งข้อมูลของ Odroid C4	48
ภาพที่ 54 ผลลัพธ์ผ่าน ThingSpeak	48
ภาพที่ 55 อุปกรณ์ Piggy Bank Saving Your Life แบบประกอบสำเร็จ	50
ภาพที่ 56 อุปกรณ์ Piggy Bank Saving Your Life เทียบขนาดกับโน้ตบุ๊ค	50

สารบัญตาราง

ตารางที่	หน้า
ตารางที่ 1 ตารางในการแสดงประสิทธิภาพในการหยดเหรียญแล้วตรวจพบเหรียญ	41
ตารางที่ 2 ตารางในการแสดงผลลัพธ์ประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5	43
ตารางที่ 3 ตารางในการแสดงผลลัพธ์ประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5	44
ตารางที่ 4 ตารางอัตราประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5	45
ตารางที่ 5 ตารางในการแสดงผลลัพธ์ประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5	46
ตารางที่ 6 ตารางในการแสดงผลลัพธ์ประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5	47
ตารางที่ 7 ตารางในการแสดงประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5	47
ตารางที่ 8 ตารางในการแสดงประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5 ของ Odroid C4	49

บทที่ 1

บทนำ

ที่มาและความสำคัญของปัญหา

ปัจจุบันเทคโนโลยีมีความสำคัญต่อการดำเนินชีวิต เพราะวิถีทางการเดินทางนั้นแทรกซึมอยู่ในทุกตารางในการใช้ชีวิตของมนุษย์ เพرامนุษย์มีการพัฒนาและคิดค้นสิ่งอำนวยความสะดวกด้านความสะอาดสภากาชาดต่อการดำเนินชีวิตเป็นอันมาก เทคโนโลยีในปัจจุบันได้เข้ามาระเริ่มปัจจัยพื้นฐานการดำเนินชีวิตได้เป็นอย่างดีและทวีความสำคัญขึ้นเมื่อโลกยิ่งพัฒนามากยิ่งขึ้น รวมถึงพลังงาน ซึ่งเป็นสิ่งจำเป็นของมนุษย์ในโลกปัจจุบันเช่นกัน เนื่องจากการผลิตพลังงานค่อย ๆ เปลี่ยนแปลงเป็นการผลิตพลังงานที่ต้องอาศัยเทคโนโลยีในการผลิตมากยิ่งขึ้น ในทุกวันของการใช้ชีวิตประจำวัน เทคโนโลยีทันสมัยเหล่านี้ได้เข้ามามีบทบาทในชีวิตของเรามากขึ้น ทำให้การดำเนินชีวิตของคนในยุคนี้มีความสะอาดสภากาชาดมากกว่าในอดีต อุปกรณ์และเครื่องทุนแรงหลากหลายชนิด ถูกคิดค้นขึ้นด้วยการนำเอatechnologyอันชาญฉลาดเข้ามาใช้ อาทิ โทรศัพท์มือถือ รถยนต์ เครื่องบิน รวมถึงเครื่องใช้ไฟฟ้าที่ถูกสร้างเป็นอุปกรณ์ IoT ทุกชนิด

กระบุก odometer นับว่าเป็นสิ่งของที่นำมาเพื่อใช้ในการออมทรัพย์ไม่ว่าจะเก็บในรูปแบบของธนบัตรหรือเหรียญภาษาปัณฑิต เป็นสิ่งของที่นิยมใช้กันทุกรั้วเรือน ในทางกลับกันในสมัยปัจจุบันกระบุก odometer ได้รับความนิยมที่ลดลง เนื่องจากการเข้าถึงเทคโนโลยีทางการเงินที่ง่ายขึ้น ทำให้การเก็บออมและฝากเงินผ่านธนาคารเป็นเรื่องที่ง่าย แต่ก็มีกลุ่มคนบางส่วนที่อาจจะไม่สามารถเข้าถึงเทคโนโลยีหรือการเงินได้มากไปกว่านั้น ปัญหาทางมูลภาพทางอากาศในปัจจุบันนั้น ได้ส่งผลเสียให้กับผู้คนในปัจจุบันอย่างมาก โดยไม่สามารถรับรู้ได้ว่าพื้นที่ที่ตนอยู่อาศัยนั้นมีผลกระทบที่เป็นอันตรายต่อระบบทางเดินหายใจหรือไม่ ทำให้ก่อให้เกิดโรคที่เกี่ยวข้องกับระบบทางเดินหายใจ จึงได้ก่อให้เกิดโครงการกระบุก odometer ที่สามารถวัดค่าฝุ่นได้ ซึ่งเป็นการผสมผสานเทคโนโลยี Internet of Things (IoT) เข้ากับแนวคิดการออมทรัพย์แบบตั้งเดิม เพื่อสร้างสรรค์อุปกรณ์ที่ทันสมัยและใช้งานได้ในชีวิตประจำวัน โดยนำสิ่งที่เคยใช้ในอดีตอย่างกระบุก odometer มาปรับปรุงให้เหมาะสมกับยุคสมัย กลายเป็นอุปกรณ์ที่ไม่เพียงแต่เก็บออมเงินและบอกจำนวนเงินได้ แต่ยังสามารถตรวจสอบค่าฝุ่นในพื้นที่ที่ตั้งกระบุก odometer ได้ด้วย เช่นเซอร์ตรวจับฝุ่นทำการวัดค่า PM 1.0, PM 2.5, และ PM 10 (มีหน่วยเป็น $\mu\text{g}/\text{m}^3$) และแสดงผลผ่านแพลตฟอร์ม Blynk บนโทรศัพท์มือถือ และที่อุปกรณ์โดยตรง นอกจากนี้ยังเชื่อมต่อข้อมูลไปยังเว็บไซต์ ThingSpeak สำหรับการดูข้อมูลออนไลน์ และมีการส่งข้อมูลของคุณภาพอากาศ PM 2.5 ผ่าน Line Notify และรองรับการสั่งงานผ่านโปรแกรม Shortcut หรือคำสั่งเสียง Siri ทำให้ง่ายต่อการติดตามคุณภาพอากาศโดยไม่ต้องเปิดแอปพลิเคชันตลอดเวลา

วัตถุประสงค์

ตอนที่ 1 เพื่อศึกษาการเขียนโปรแกรม Arduino และศึกษาการทำงานของบอร์ด Arduino ในการแยกเหวี่ยง

ตอนที่ 2 เพื่อศึกษาหาประสิทธิภาพของการส่งข้อมูลประเภท PM 2.5 ที่ถูกส่งไปยังอุปกรณ์ต่าง ๆ

ตอนที่ 3 เพื่อพัฒนาให้อุปกรณ์ Internet of Things สามารถใช้ในบ้าน และชีวิตประจำวันได้

สมมติฐาน

ตอนที่ 1 หากเขียนโปรแกรม Arduino และศึกษาการทำงานของบอร์ด Arduino จนเข้าใจ จะสามารถทำให้อุปกรณ์ทำงานร่วมกันได้

ตอนที่ 2 หากชุดคำสั่งการทำงานของ Arduino, PMS5003 Sensor และ NodeMCU มีประสิทธิภาพ ที่ดี ประสิทธิภาพของการส่งข้อมูลประเภท PM 2.5 ที่ถูกส่งไปยังอุปกรณ์ต่าง ๆ จะดีตามไปด้วย

ตอนที่ 3 หากนำอุปกรณ์ Internet of Things ให้สามารถใช้ในบ้าน และชีวิตประจำวันได้ จะเป็นการต่อยอดความรู้ความสามารถในด้านความคิดสร้างสรรค์ และความรู้ความสามารถเกี่ยวกับการนำ Arduino มาใช้ประโยชน์ในชีวิตประจำวัน

บทที่ 2

เอกสารและงานวิจัยที่เกี่ยวข้อง

โครงการเรื่อง Piggy Bank Saving Your Life ทางผู้จัดทำได้ดำเนินงานโครงการโดยมีเอกสารและงานวิจัยที่เกี่ยวข้อง ดังต่อไปนี้

เอกสารประกอบการเรียน 240-319 Embedded System Module บทที่ 3 Interrupt of AVR

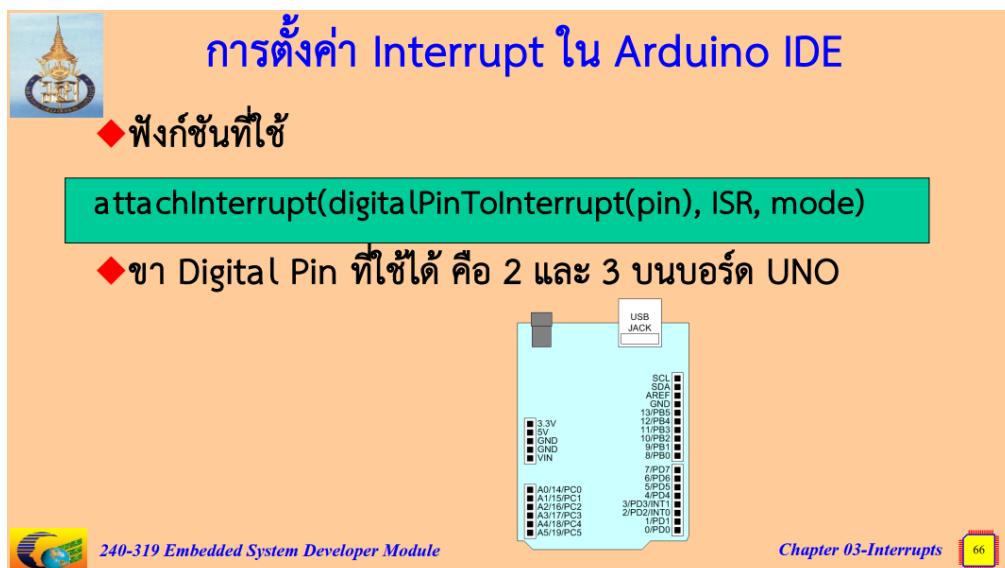


เวกเตอร์การขัดจังหวะของ ATMEGA328P

V _N	S _A	ชนิดของ การขัดจังหวะ	คำอธิบาย
1	0x0000	RESET	เกิดการรีเซ็ตตัวประมวลผล
2	0x0002	INT0	การขัดจังหวะจากภายนอกผ่านทางขาสัญญาณ INT0
3	0x0004	INT1	การขัดจังหวะจากภายนอกผ่านทางขาสัญญาณ INT1
4	0x0006	PCINT0	การขัดจังหวะแบบพินเข็นที่ขาสัญญาณ PCINT[7:0]
5	0x0008	PCINT1	การขัดจังหวะแบบพินเข็นที่ขาสัญญาณ PCINT[14:8]
6	0x000A	PCINT2	การขัดจังหวะแบบพินเข็นที่ขาสัญญาณ PCINT[23:16]
7	0x000C	WDT	การขัดจังหวะจากการรับเข้าเวลาเรือต์ต่อ ก
8	0x000E	TIMER2 COMPA	การขัดจังหวะจากการจรนับ/จับเวลาหมายเลข 2 เมื่อค่าในเรจิสเตอร์ TCNT2 เท่ากับค่าในเรจิสเตอร์ OCR2A
9	0x0010	TIMER2 COMPB	การขัดจังหวะจากการจรนับ/จับเวลาหมายเลข 2 เมื่อค่าใน TCNT2 เท่ากับค่าในเรจิสเตอร์ OCR2B
10	0x0012	TIMER2 OVF	การขัดจังหวะจากการจรนับ/จับเวลาหมายเลข 2 เมื่อเกิดการล้นของ ค่าในเรจิสเตอร์ TCNT2

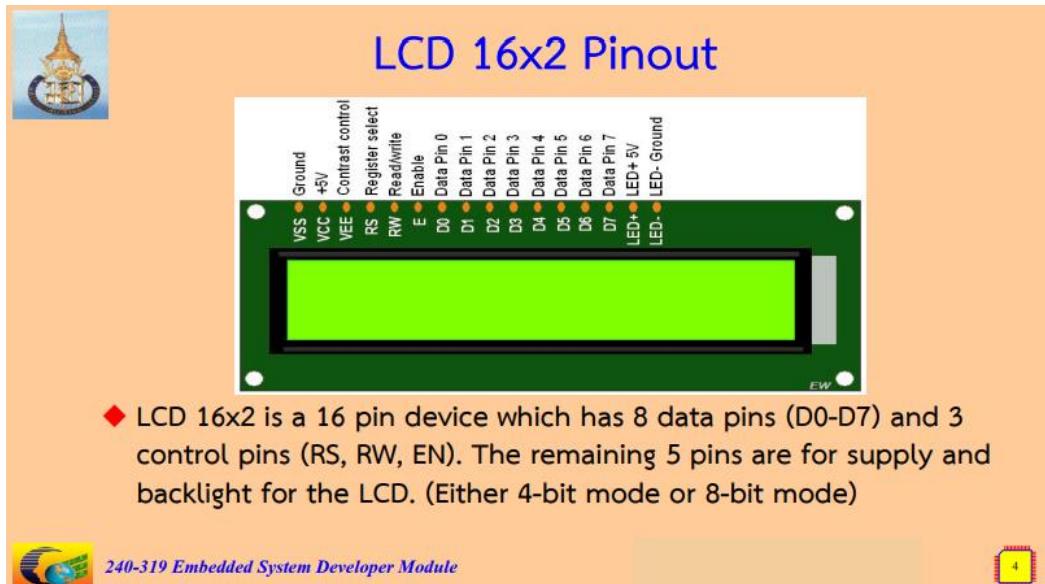
 240-319 Embedded System Developer Module Chapter 03-Interrupts 15

ภาพที่ 1 เวกเตอร์การขัดจังหวะของ ATMega328P



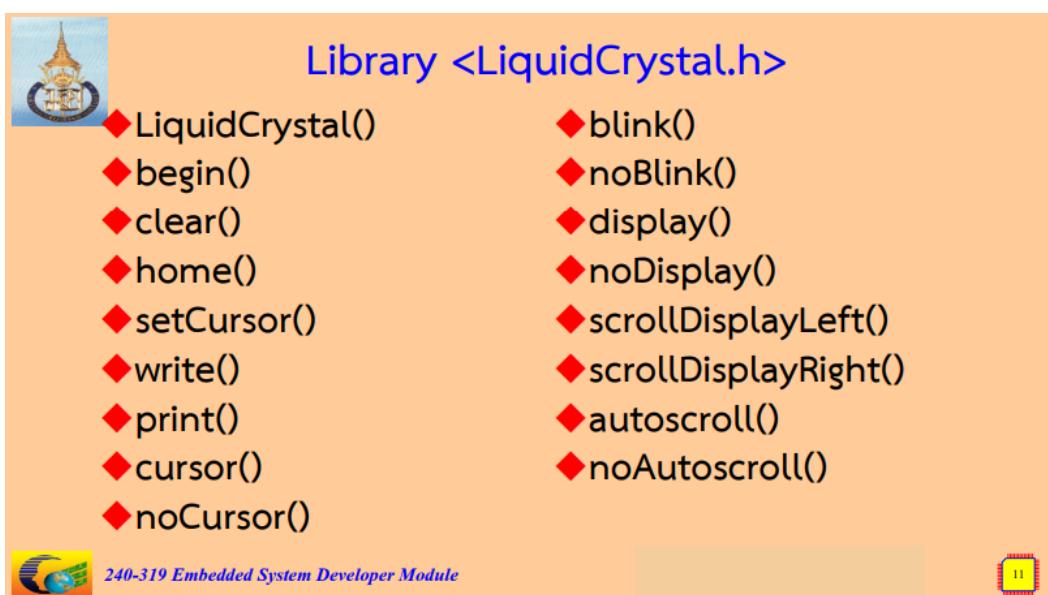
ภาพที่ 2 การตั้งค่า Interrupt ใน Arduino IDE

เอกสารประกอบการเรียน 240-319 Embedded System Module บทที่ 9 LCD



- ◆ LCD 16x2 is a 16 pin device which has 8 data pins (D0-D7) and 3 control pins (RS, RW, EN). The remaining 5 pins are for supply and backlight for the LCD. (Either 4-bit mode or 8-bit mode)

ภาพที่ 3 LCD 16x2 Pinout



ภาพที่ 4 Library LiquidCrystal.h

เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 13 I2C and PCF-6574



แนะนำการเชื่อมต่อแบบทู-ไวร์

- ◆ Two-Wire Interface : TWI
- ◆ เรียกอีกอย่างหนึ่งว่าระบบบัส TWI
- ◆ เป็นการสื่อสารอนุกรมอย่างง่ายที่มีความซับซ้อนต่ำ
- ◆ มีสายสัญญาณสองเส้น
 - ◆ SCL (ย่อมาจาก Serial Clock)
 - ◆ SDA (ย่อมาจาก Serial Data)
- ◆ สัญญาณแต่ละเส้นจะต้องถูกเชื่อมต่อด้วยตัวต้านทานตึงขึ้น
- ◆ ระบบบัสถูกออกแบบมาให้รองรับอุปกรณ์บนสายสัญญาณห้องสองได้มากสุด ถึง 128 ตัว

 240-319 Embedded System Developer Module Chapter 13 – I2C and PCF-8574 4

ภาพที่ 5 การแนะนำการเชื่อมต่อแบบ Two-Wire



ไอซีขยายพอร์ตอินพุตเอาต์พุต PCF8574

- ◆ ใช้เพิ่มขยายพอร์ตอินพุตเอาต์พุต มีพอร์ตขนาด 8 บิต ให้ใช้งานจำนวน 1 พอร์ต
- ◆ แต่ละบิตสามารถใช้เป็นอินพุตหรือเอาต์พุตได้อย่างอิสระ
- ◆ หมายเลขที่อยู่ A[6:3] เท่ากับ 0100₂ ส่วนบิต A[2:0] สามารถกำหนดเองโดยผู้ใช้งานได้โดยผ่านการป้อนค่าตறรรจากเข้าที่ขาใช้งาน A[2:0] ของตัวไอซีเอง
- ◆ สามารถส่งสัญญาณการขัดจังหวะไปที่ตัวประมวลผลได้ ส่งผลให้การใช้งานเป็นพอร์ตรับเข้ามีประสิทธิภาพสูง เนื่องจากตัวประมวลผลไม่จำเป็นต้องกราดตรวจสภาวะอินพุตอยู่ตลอดเวลา
- ◆ หากมีการเปลี่ยนสภาวะทางตรรกะที่อินพุตก็จะส่งผลให้ไอซี PCF8574 ส่งสัญญาณการขัดจังหวะไปบอกให้ตัวประมวลผลได้รับทราบ

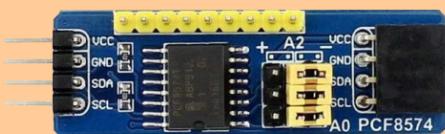
 240-319 Embedded System Developer Module Chapter 13 – I2C and PCF-8574 59

ภาพที่ 6 รายละเอียด IC PCF8574 (1)



ไอซีขยายพอร์ต PCF8574

- ◆ ใช้สำหรับเพิ่มขยายพอร์ตอินพุตเอาต์พุต
- ◆ มีพอร์ตขนาด 8 บิตให้ใช้งาน 1 พอร์ต
- ◆ หมายเลขที่อยู่ A[6:3] เท่ากับ 0100₂ บิต A[2:0] กำหนดได้เองโดยผู้ใช้
- ◆ หมายเลขตำแหน่ง 0100000-0100111 หรือ 0x20-0x27



240-319 Embedded System Developer Module

Chapter 13 – I2C and PCF-8574

60

ภาพที่ 7 รายละเอียด IC PCF8574 (2)

เอกสารประกอบการเรียน 240-319 Embedded System Module บทที่ 14 DS-1307



ไอซีนาฬิกาเวลาจري่ง DS1307

- ◆ ใช้แบตเตอรี่สำรองขนาดเล็ก (กินไฟในการทำงานต่ำกว่า 500 นาโนแอม培ร์)
- ◆ ปรับวันเวลาของปีได้ถูกต้องตามปีปกติสุริ Tin (365 วัน) และปีอธิกสุริ Tin (366 วัน)
- ◆ ไม่มีขาแอ็อดเดรส จึงสามารถตั้งค่าโดยผู้ใช้เมมอย่างไอซีตัวอื่น
- ◆ มีหน่วยความจำแรมสกิดิตภัยใน 64 ตำแหน่ง แต่ละตำแหน่งเก็บได้ 1 ไบต์
 - ◆ ตำแหน่ง 0-6 เก็บค่าวันและเวลาของระบบ
 - ◆ ตำแหน่ง 7 เก็บไปต์ควบคุมการทำงานของไอซีเอง
 - ◆ ตำแหน่ง 8-63 ใช้เป็นหน่วยความจำชนิดไม่ลับเลือน (non volatile mem)

240-319 Embedded System Developer Module

Chapter 14 – DS1307

7

ภาพที่ 8 รายละเอียด IC DS1307

เอกสารประกอบการเรียน 240-319 Embedded System Module บทที่ 15 Power management and sleep mode



ภาวะหลับของตัวประมวลผล

- ◆ สถาบันเทคโนโลยีราชมงคลรัตนโกสินทร์ ได้แนะนำการเข้าสู่ภาวะหลับ (Sleep mode) ซึ่งช่วยให้ตัวประมวลผลกินไฟลดลง
- ◆ แบบวิธีของภาวะหลับ 6 แบบ
 - ◆ แบบวิธีเดินเครื่องเปล่า (Idle mode)
 - ◆ แบบวิธีลดสัญญาณรบกวนของวงจรแปลงแอนะล็อกเป็นดิจิทัล (ADC Noise Reduction mode)
 - ◆ แบบวิธีลดพลังงาน (Power-down mode)
 - ◆ แบบวิธีประหยัดพลังงาน (Power-save mode)
 - ◆ แบบวิธีสแตนด์บาย (Standby mode)
 - ◆ แบบวิธีสแตนด์บายส่วนเพิ่ม (Extended Standby mode)

240-319 Embedded System Developer Module **Chapter 15 - Sleep modes**

4

ภาพที่ 9 ภาวะหลับของตัวประมวลผล



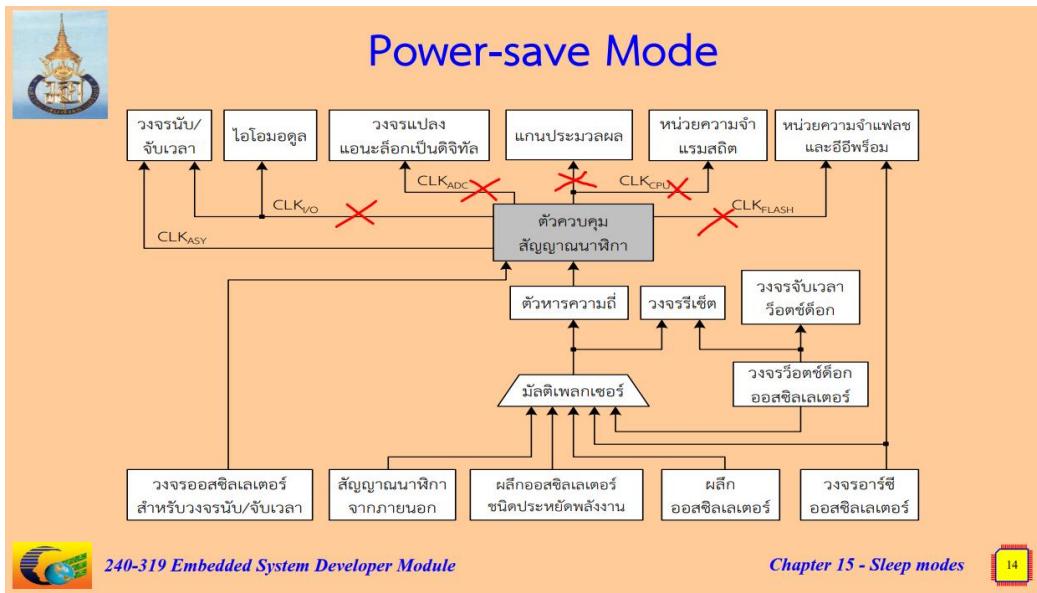
Power-save Mode

- ◆ This mode is identical to Power-down, with one exception
- ◆ If Timer/Counter2 is enabled, it will keep running during sleep.
- ◆ If Timer/Counter2 is not running, Power-down mode is recommended instead of Power-save mode.

240-319 Embedded System Developer Module **Chapter 15 - Sleep modes**

13

ภาพที่ 10 รายละเอียด Power-Save Mode



ภาพที่ 11 Power-Save Mode



ภาพที่ 12 เรจิสเตอร์ SMCR



Sleep mode select

SM2	SM1	SM0	Sleep Mode
0	0	0	Idle
0	0	1	ADC Noise Reduction
0	1	0	Power-down
0	1	1	Power-save
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Standby ⁽¹⁾
1	1	1	Extended Standby ⁽¹⁾



ภาพที่ 13 Sleep Mode แบบต่าง ๆ



การสั่งการให้ตัวประมวลผลเข้าสู่ภาวะหลับ

<ul style="list-style-type: none"> ◆ ภาษาแอสเซมบลี <ul style="list-style-type: none"> ◆ เรียกใช้คำสั่ง sleep ◆ ภาษาซี <ul style="list-style-type: none"> ◆ เรียกใช้ฟังก์ชัน sleep_cpu() ในคลังโปรแกรม <avr/sleep.h>



ภาพที่ 14 การสั่งให้ CPU เข้าสู่ภาวะหลับ



ฟังก์ชันสำหรับเปิดทาง/ปิดทางการเข้าสู่ภาวะหลับ

```

1 #define SLEEP_IDLE 0 //ค่าสำหรับตั้งให้เข้าสู่ภาวะหลับแบบเดินเครื่องเบล่า
2 #define SLEEP_ADC_REDUCE 1 //แบบบวชึลดเสียงรบกวนของวงจรแปลงแอนะล็อกเป็นดิจิทัล
3 #define SLEEP_POWER_DOWN 2 //ค่าสำหรับตั้งให้เข้าสู่ภาวะหลับแบบลดพลังงาน
4 #define SLEEP_POWER_SAVE 3 //ค่าสำหรับตั้งให้เข้าสู่ภาวะหลับแบบประหยัดพลังงาน
5 #define SLEEP_STANDBY 6 //ค่าสำหรับตั้งให้เข้าสู่ภาวะหลับแบบสแตนด์บาย
6 #define SLEEP_EXT_STANBY 7 //ค่าสำหรับตั้งให้เข้าสู่ภาวะหลับแบบสแตนด์บายส่วนเพิ่ม
7 #include <avr/sleep.h> //เรียกใช้คลังโปรแกรมสำหรับเรียกฟังก์ชัน sleep_cpu
8 void SLEEP_DISABLE (void) //ฟังก์ชันสำหรับสั่งปิดทางไว้หัวประมวลผลเข้าสู่ภาวะหลับ
9 {
10     SMCR &= 0xFE; //สั่งให้ปิต SE (ปิตค่าสูง) ใน rejister SMCR มีค่าตระกงต่ำ
11 }
12 void SLEEP_INITIALIZE(uint8_t m) //ฟังก์ชันสำหรับเปิดทางให้สู่ภาวะหลับได้
13 {
14     SMCR = (m << 1) | 0x01; //ตั้งค่าแบบบวชิริภาวะหลับ และตั้งให้ปิต SE มีค่าตระกงสูง
15 } //สั่งสุดฟังก์ชัน SLEEP_INITIALIZE

```

 240-319 Embedded System Developer Module Chapter 15 - Sleep modes 27

ภาพที่ 15 ตัวอย่างฟังก์ชันในการเปิด/ปิดทางการเข้าสู่ภาวะหลับ (1)



ตัวอย่างการใช้งาน

ฟังก์ชันสำหรับเปิดทาง/ปิดทางการเข้าสู่ภาวะหลับ

```

16 int main(void) //ฟังก์ชันหลักของโปรแกรม
17 { //เริ่มฟังก์ชันหลัก
18     .... //การทำงานอื่นๆ ก่อนจะเรียกฟังก์ชัน SLEEP_INITIALIZE
19     SLEEP_INITIALIZE(SLEEP_IDLE); //ตัวอย่างการตั้งให้ทำงานในภาวะหลับแบบเดินเครื่องเบล่า
20     sleep_cpu(); //สั่งให้ตัวประมวลผลเข้าสู่ภาวะหลับ
21     SLEEP_DISABLE(); //เมื่อตื่นจากภาวะหลับ ให้ปิดทางการเข้าสู่ภาวะหลับ
22     .... //ส่วนอื่นๆ ที่ต้องกระทำการหลังจากการตื่นจากภาวะหลับ
23 } //สั่งสุดฟังก์ชันหลัก

```

 240-319 Embedded System Developer Module Chapter 15 - Sleep modes 28

ภาพที่ 16 ตัวอย่างฟังก์ชันในการเปิด/ปิดทางการเข้าสู่ภาวะหลับ (2)

เอกสารประกอบการเรียน 240-319 Embedded System Module บทที่ 17 EEPROM and Serial EEPROM



EEPROM

- ◆ Electrically erasable programmable read-only memory (also called E²PROM : E-square PROM)
- ◆ EEPROM is usually integrated in microcontrollers
- ◆ EEPROM is NOR-type

 240-319 Embedded System Developer Module Chapter 17 - EEPROM 

ภาพที่ 17 รายละเอียด EEPROM



หน่วยความจำอีอีพร้อม

- ◆ Electrically Erasable Programmable Read-Only Memory (EEPROM)
- ◆ เป็นหน่วยความจำชนิดไม่ลบเลือน (Non-Volatile memory)
- ◆ จำนวนครั้งในการเขียนข้อมูลมีได้จำกัด
- ◆ ไม่ควรนำมาใช้ในการเก็บตัวแปรชั่วคราวก่อนและหลังการประมวลผล
- ◆ มากใช้ในการเก็บค่าโครงร่างแบบ (Configuration) ของระบบที่ผู้ใช้ต้องมีการเปลี่ยนแปลงบ้างแต่ไม่บ่อยนัก
- ◆ อีอีพร้อมของตัวประมวลผลชุด ATmega สามารถเขียนค่าได้ไม่ต่ำกว่า 100,000 ครั้ง

 240-319 Embedded System Developer Module Chapter 17 - EEPROM 

ภาพที่ 18 หน่วยความจำ EEPROM

EEPROM of the ATmega Series

- ◆ ATmega48A 256 Bytes
- ◆ ATmega168A 512 Bytes
- ◆ ATmega328P 1 KBytes

240-319 Embedded System Developer Module Chapter 17 - EEPROM 45

ภาพที่ 19 ขนาดของ EEPROM ใน AVR แต่ละตัว

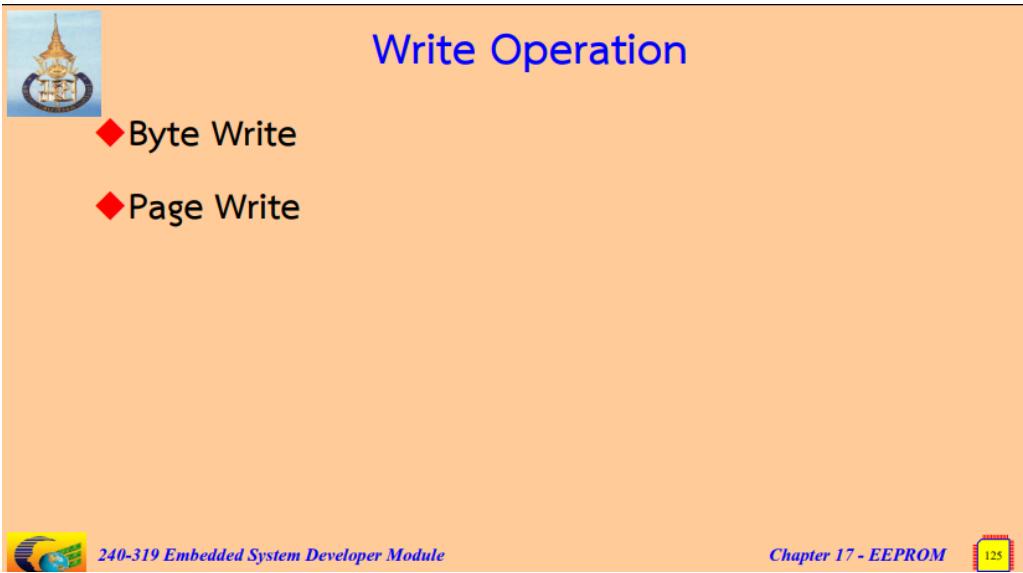
จำนวนครั้งที่บันทึกข้อมูลได้

	data	status
0	FF	FF
2	FF	FF
4	FF	FF
6	FF	FF
8	FF	FF
10	CC	00
1022		FF
1023		FF

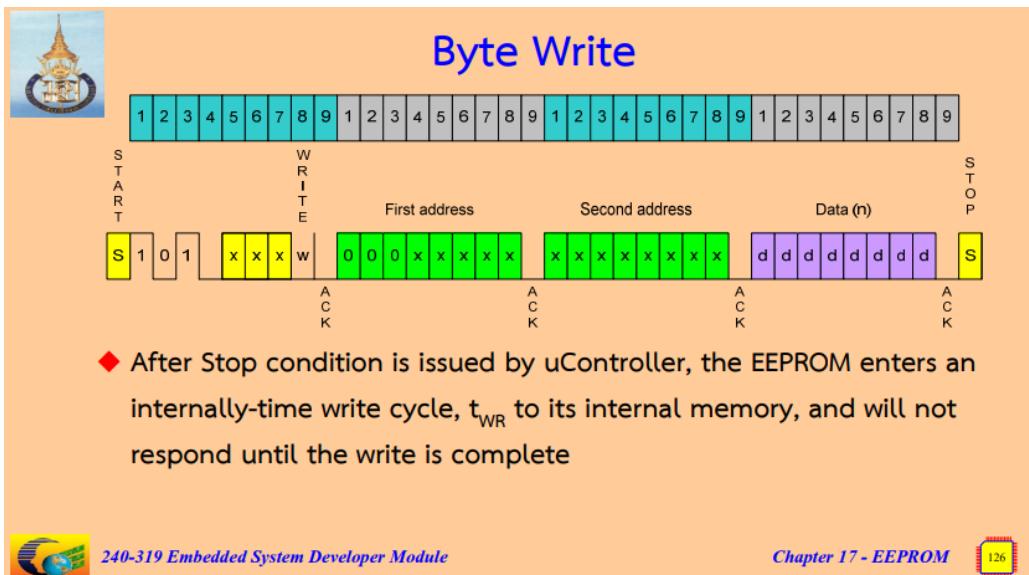
- ◆ หน่วยความจำ 1024 ตำแหน่ง
- ◆ ใช้ 2 ตำแหน่งเขียนค่าสถานะ 1 ครั้ง
- ◆ แต่ละตำแหน่งเขียนได้ 100000 ครั้ง
- ◆ ดังนั้น จึงสามารถบันทึกสถานะของเครื่องได้ 51.2 ล้านครั้ง

240-319 Embedded System Developer Module Chapter 17 - EEPROM 103

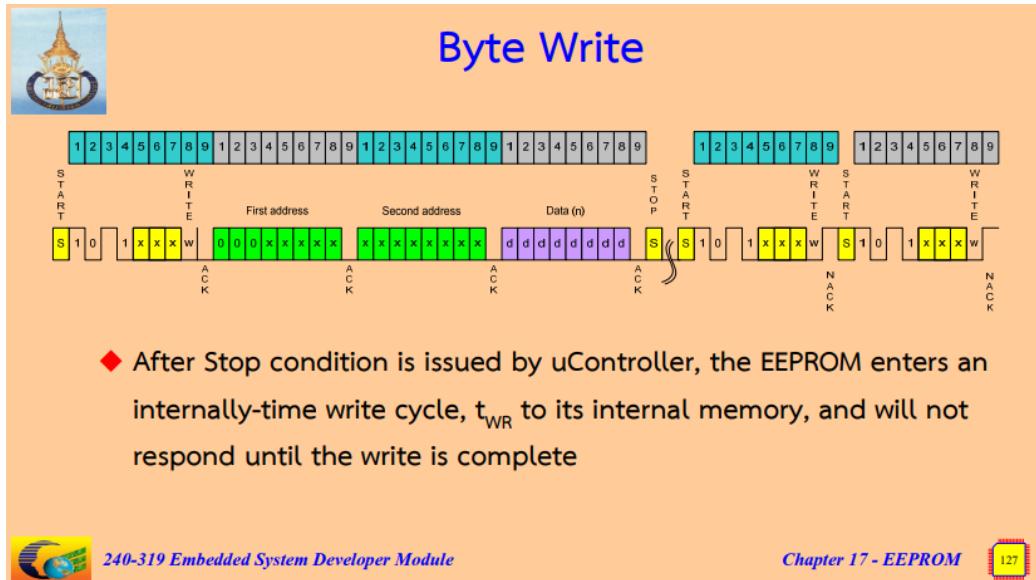
ภาพที่ 20 จำนวนครั้งที่บันทึกข้อมูลลง EEPROM ได้



ภาพที่ 21 รูปแบบการ Write ของ EEPROM



ภาพที่ 22 การ Write ของ EEPROM แบบ Byte Write

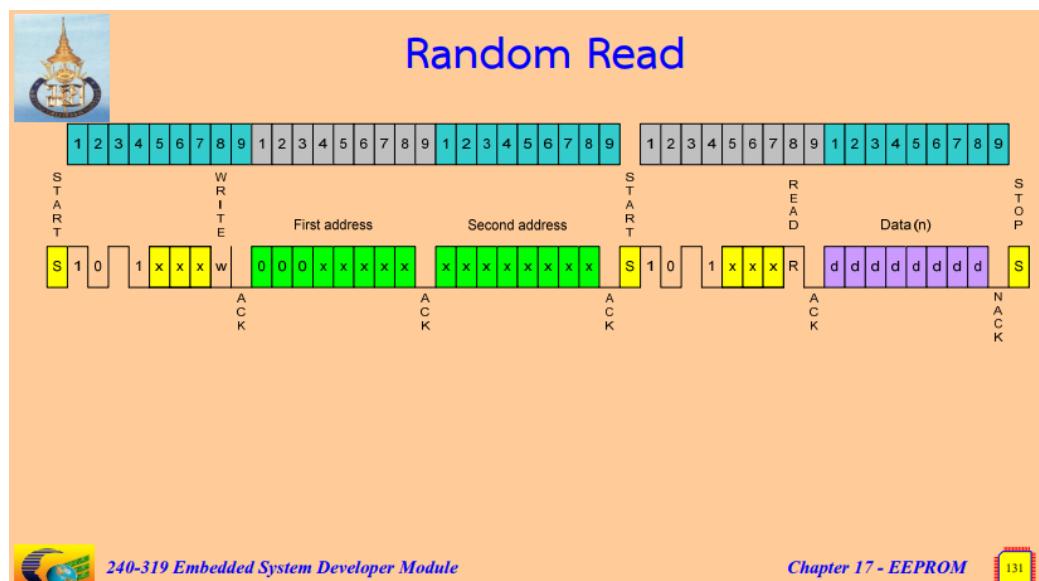


240-319 Embedded System Developer Module

Chapter 17 - EEPROM

127

ภาพที่ 23 การ Write ของ EEPROM แบบ Byte Write



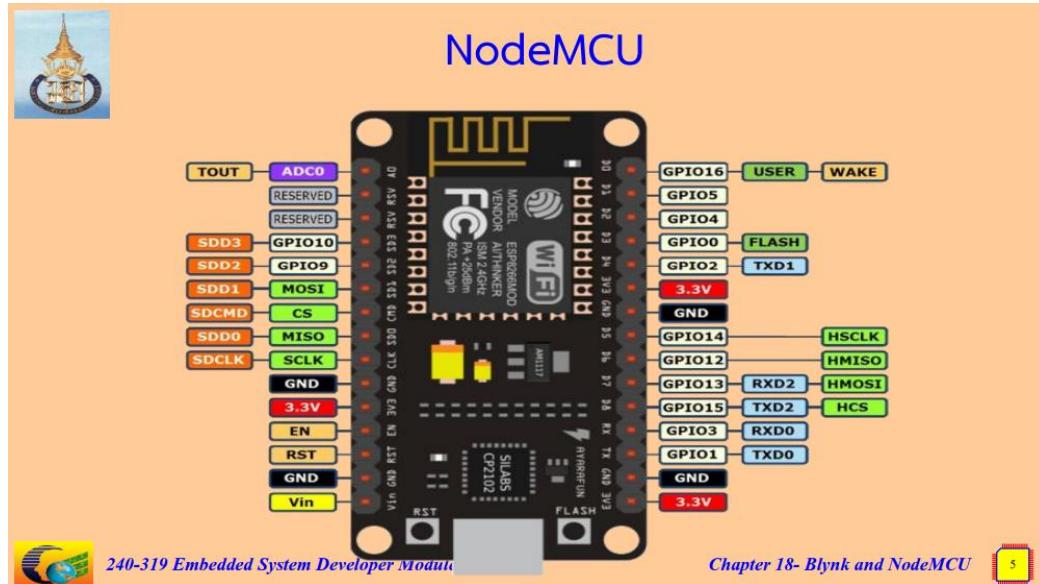
240-319 Embedded System Developer Module

Chapter 17 - EEPROM

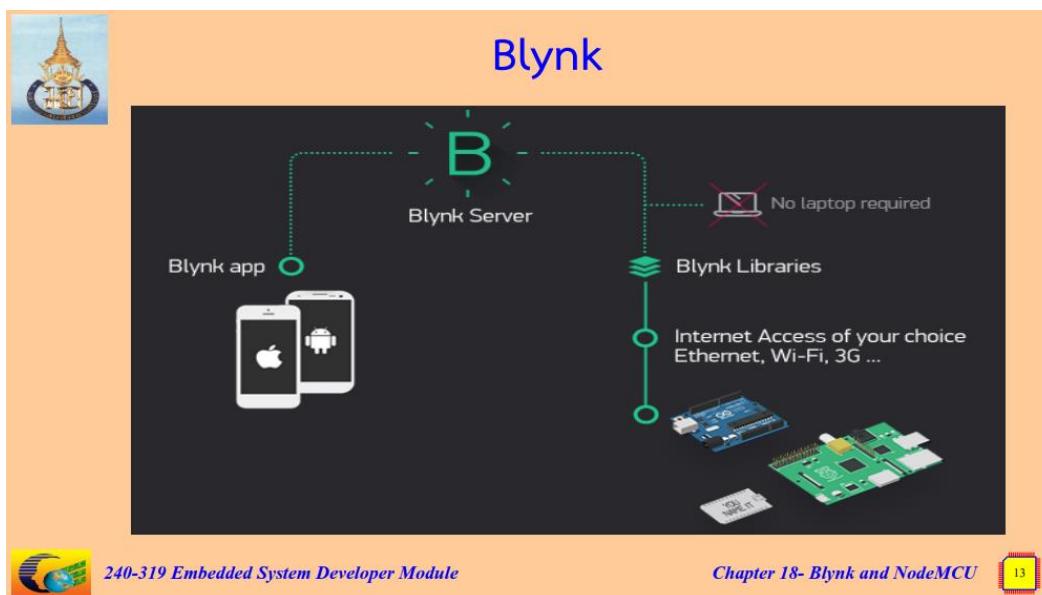
131

ภาพที่ 24 การ Read ของ EEPROM แบบ Random Read

เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บทที่ 18 NodeMCU and Blynk



ภาพที่ 25 NodeMCU Pinout

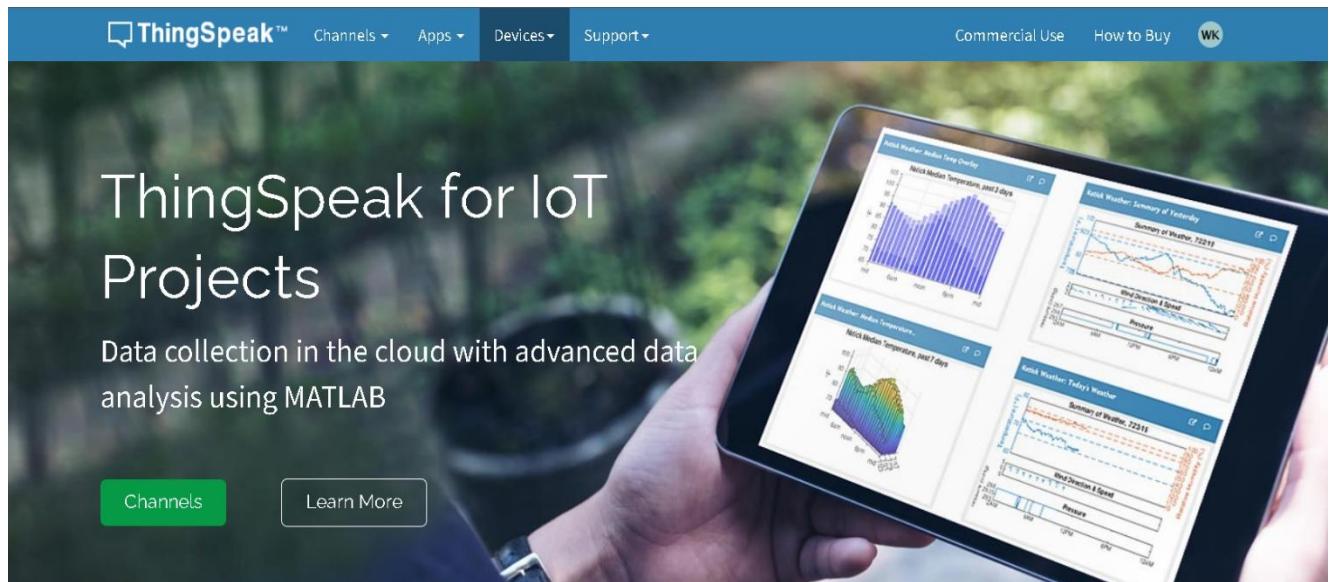


ภาพที่ 26 แอพพลิเคชัน Blynk

เอกสารประกอบการเรียน วิชา 240-319 Embedded System Module บน Odroid



ภาพที่ 27 Odroid C4 Pinout



ภาพที่ 28 ThingSpeak Website

บทที่ 3

วิธีการดำเนินงาน

โครงการเรื่อง Piggy Bank Saving Your Life ทางผู้จัดทำได้ดำเนินงานโครงการโดยมีการดำเนินงานตามขั้นตอนดังต่อไปนี้

1. การออกแบบ และรายละเอียด

รายละเอียดเกี่ยวกับขั้นงานโดยรวม:

เป็นอุปกรณ์ Internet of Things ที่ผสมผสานระหว่าง Traditional และ Modern เข้าด้วยกันอย่างเหมาะสม ก่อร่างกายจะนำสิ่งของที่มีใช้ในอดีตมารวมกับปัจจุบัน ซึ่งกลไกมาเป็นอุปกรณ์ที่เป็นกระปุกออมสิน อัจฉริยะ ที่มีการแสดงผลว่าเงินที่เราออมนั้น มีด้วยกันกี่บาทแล้ว พร้อมทั้งยังสามารถตรวจสอบผู้คนในพื้นที่ที่ตั้ง กระปุกออมสินดังกล่าว ในรูปแบบ Particular Matter หรือ PM แบบ 1.0, 2.5, 10 ที่มีค่าเป็น $\mu\text{g}/\text{m}^3$ ซึ่งสามารถแจ้งผลลัพธ์ดังกล่าวผ่าน Blynk บนโทรศัพท์มือถือ และผ่าน ThingSpeak บนเว็บไซต์ มีการส่ง ข้อมูลของคุณภาพอากาศ PM 2.5 ผ่าน Line Notify หรือหากไม่สะดวกในการเปิดแอป หรือเปิดเว็บไซต์ ยังคงสามารถเปิดผ่านโปรแกรม Shortcut หรือเรียกใช้ผ่านคำสั่งเสียง Siri ได้

1.1 วัสดุอุปกรณ์

- 1.1.1 Arduino Uno R3
- 1.1.2 CoE Arduino Shield
- 1.1.3 Odroid C4
- 1.1.4 I2C LCD 16x2
- 1.1.5 DS1307
- 1.1.6 Button
- 1.1.7 PMS5003 Sensor
- 1.1.8 Counter Module Motor Speed Sensor
- 1.1.9 NodeMCU
- 1.1.10 Keypad
- 1.1.11 PCF8574
- 1.1.12 สายไฟซิมพ์ประเภท ผู้-เมีย
- 1.1.13 หม้อแปลงจ่ายไฟ 12V
- 1.1.14 ปืนกาว
- 1.1.15 แท่งกาว
- 1.1.16 กระดาษชานอ้อย

1.1.17 สาย USB Module ในการวัดระดับแรงดัน และกระแส

1.1.18 Breadboard

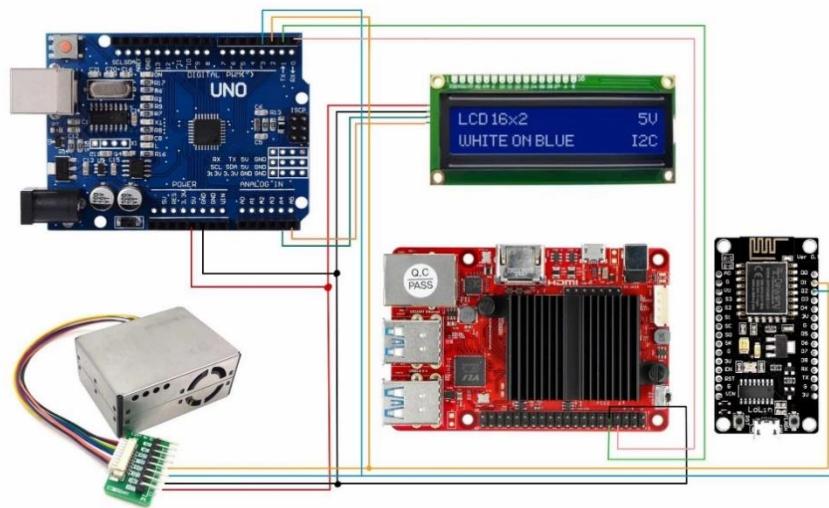
1.1.19 LAN Cable (RJ45)

1.1.20 LED

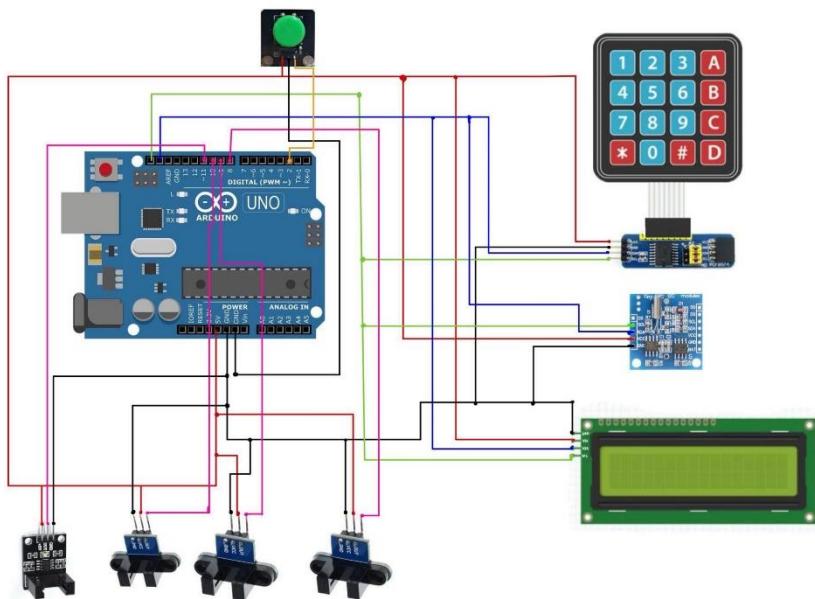
1.1.21 Fidget Spinner

2. การออกแบบวงจร

2.1 Schematic Diagram



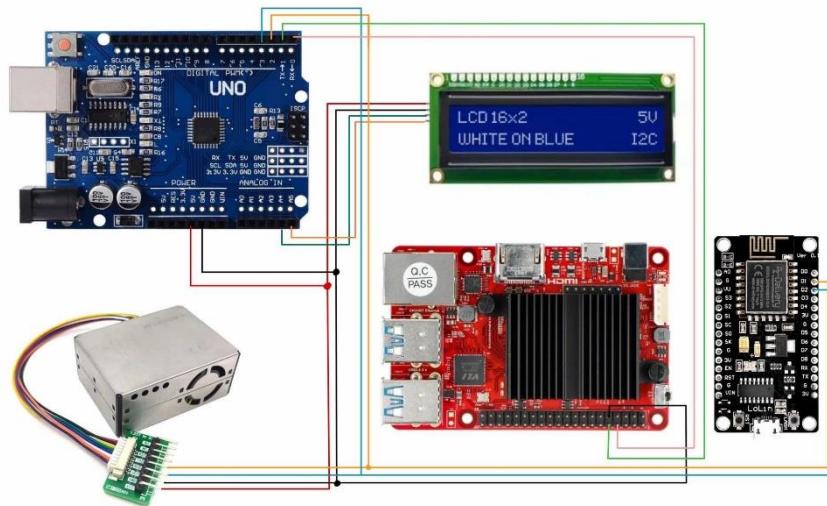
ภาพที่ 29 Diagram ของวงจรสำหรับการวัดค่าฝุ่น



ภาพที่ 30 Diagram ของวงจรสำหรับการแยกเหรียญ และนาฬิกาแสดงเวลา

2.2 Wiring

2.2.1 วงจรสำหรับการวัดค่าฝุ่น



ภาพที่ 31 Diagram ของวงจรสำหรับการวัดค่าฝุ่น

2.2.1.1 สายสีแดง เป็นสาย 5V ที่ต่อจาก UNO R3 ไปยังขา VCC ของบอร์ดทุก

บอร์ด ยกเว้น Odroid C4 และ NodeMCU

2.2.1.2 สายสีดำ เป็นสาย GND ที่ต่อจาก UNO R3 ไปยังขา GND ของบอร์ดทุก

บอร์ด ยกเว้น NodeMCU

2.2.1.3 สาย SDA ต่อจาก UNO R3 Pin A4 ไปยังขา SDA ของ I2C LCD 16x2

2.2.1.4 สาย SCL ต่อจาก UNO R3 Pin A5 ไปยังขา SCL ของ I2C LCD 16x2

2.2.1.5 สาย RX (1) ของ Uno R3 Pin 2 ไปยังขา TXD ของ PMS5003 Sensor

และขา D1 ของ NodeMCU

2.2.1.6 สาย TX (1) ของ Uno R3 Pin 3 ไปยังขา RXD ของ PMS5003 Sensor

และขา D2 ของ NodeMCU

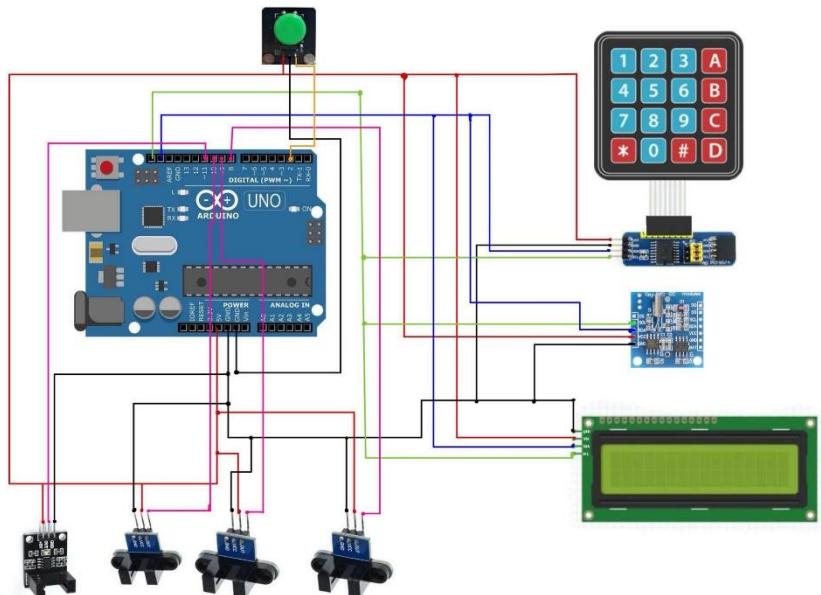
2.2.1.7 สาย RX (2) ของ Uno R3 Pin 0 ไปยังขา 8 (UART_EE_A_TX) ของ

Odroid C4

2.2.1.8 สาย TX (2) ของ Uno R3 Pin 1 ไปยังขา 10 (UART_EE_A_RX) ของ

Odroid C4

2.2.2 วงจรสำหรับการแยกเหรียญ



ภาพที่ 32 Diagram ของวงจรสำหรับการแยกเหรียญ และนาฬิกาแสดงเวลา

2.2.1.1 สายสีแดง เป็นสาย 5V ที่ต่อจาก Uno R3 ไปยังขา VCC ของทุกอุปกรณ์

2.2.1.2 สายสีดำ เป็นสาย GND ที่ต่อจาก Uno R3 ไปยังขา GND ของทุกอุปกรณ์

2.2.1.3 สาย SDA ต่อจาก Uno R3 Pin SDA ไปยังขา SDA ของ I2C LCD 16*2, DS1307 และ PCF8574

DS1307 และ PCF8574

2.2.1.4 สาย SCL ต่อจาก Uno R3 Pin A5 ไปยังขา SCL ของ I2C LCD 16*2,

DS1307 และ PCF8574

2.2.1.5 Pin 8 ของ Uno R3 ไปยังขา OUT ของ Counter Module Motor Speed Sensor ตัวที่ 1

2.2.1.6 Pin 9 ของ Uno R3 ไปยังขา OUT ของ Counter Module Motor Speed Sensor ตัวที่ 2

2.2.1.7 Pin 10 ของ Uno R3 ไปยังขา OUT ของ Counter Module Motor Speed Sensor ตัวที่ 3

2.2.1.8 Pin 11 ของ Uno R3 ไปยังขา OUT ของ Counter Module Motor

Speed Sensor ตัวที่ 4

3. ขั้นตอนการดำเนินงาน

- 3.1 กำหนดหัวข้อโครงการ โดยสมาชิกทั้ง 3 คน ประชุมร่วมกันคิด และวางแผนว่าจะนำเรื่องใด มาศึกษา
- 3.2 สมาชิกปรับปรุงแก้ไขแผนให้เข้ากับบริบท
- 3.3 ศึกษา และค้นคว้าข้อมูลที่เกี่ยวข้องกับโครงการ และเก็บข้อมูลดังกล่าวในการจัดทำเนื้อหา ต่อไป
- 3.4 ศึกษาการสร้างอุปกรณ์ที่แยก และจัดเก็บหรือโดยการใช้บอร์ด Arduino R3 และสร้าง อุปกรณ์ที่แสดงค่า Particulate Matter โดยการใช้บอร์ด Arduino R3 และอุปกรณ์อื่น ๆ ที่เกี่ยวข้องจากแหล่งข้อมูลต่าง ๆ ที่นำเสนอหลักการวิธีการสร้างกระปุกอนมินที่พึงเป็น และการนำบอร์ด Arduino UNO R3 มาใช้ในการแยกหรือ แสดงค่าฝุ่น Particulate Matter
- 3.5 เขียนที่มาและความสำคัญของปัญหา วัตถุประสงค์ สมมติฐาน โดยศึกษาหาข้อมูลจาก อินเทอร์เน็ต และเพิ่มเติมแหล่งที่มาของข้อมูล
- 3.6 ปฏิบัติการทำโครงการ เรื่อง Piggy Bank Saving Your Life
- 3.7 ปฏิบัติขั้นตอนในการสร้างอุปกรณ์ในการวัดค่าฝุ่น
 - 3.7.1 ต่อ Arduino R3 เข้ากับอุปกรณ์ที่เกี่ยวข้อง โดยอ้างอิงมาจาก Schematic Diagram จากการออกแบบวงจรได้แก่ PMS5003 เพื่อให้รับค่าฝุ่นมาแสดงบน Serial Monitor ให้เสร็จสิ้นก่อน จากนั้นนำ I2C LCD 16*2 มาแสดงค่า Particulate Matter ที่รับมา จาก PMS5003, Odroid C4 จะมีการเชื่อมต่อแบบ UART กับ Arduino UNO R3 เพื่อรับค่าฝุ่นแล้วแสดงผลบน ThingSpeak และส่งค่าดังกล่าวไปยัง Line Notify และ NodeMCU จะรับค่าโดยตรงมาจาก PMS5003 ในการรับข้อมูลมาแล้วแสดงผล ผ่าน Blynk และเรียกใช้ผ่าน Siri และ Shortcut
 - 3.7.2 เขียนโค้ดที่เกี่ยวข้องกับอุปกรณ์ 3 อุปกรณ์ ได้แก่ Arduino UNO R3, NodeMCU และ Odroid C4
 - 3.7.2.1 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003



```

1 #include <SoftwareSerial.h>
2 #include <Wire.h>
3 #include <LiquidCrystal_I2C.h>
4
5 SoftwareSerial odroidCommu(0, 1);
6 SoftwareSerial pmsSerial(2, 3);
7
8 LiquidCrystal_I2C lcd(0x27, 16, 2);
9
10 void setup() {
11   Serial.begin(9600);
12   pmsSerial.begin(9600);
13   lcd.init();
14   lcd.backlight();
15 }
16
17 struct pms5003data {
18   uint16_t framelen;
19   uint16_t pm10_standard, pm25_standard, pm100_standard;
20   uint16_t pm10_env, pm25_env, pm100_env;
21   uint16_t particles_03um, particles_05um, particles_10um, particles_25um, particles_50um, particles_100um;
22   uint16_t unused;
23   uint16_t checksum;
24 };
25
26 struct pms5003data data;
27
28 void loop() {
29   if (readPMSdata(&pmsSerial)) {
30     Serial.print("PM25:");
31     Serial.println(data.pm25_env);
32     odroidCommu.print(data.pm25_env);
33     lcd.clear();
34     lcd.setCursor(0, 0);
35     lcd.print("PM 2.5: ");
36     lcd.print(data.pm25_env);
37     lcd.print(" ug/m3");
38     delay(2000);
39   }
40 }
41

```

ภาพที่ 33 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003

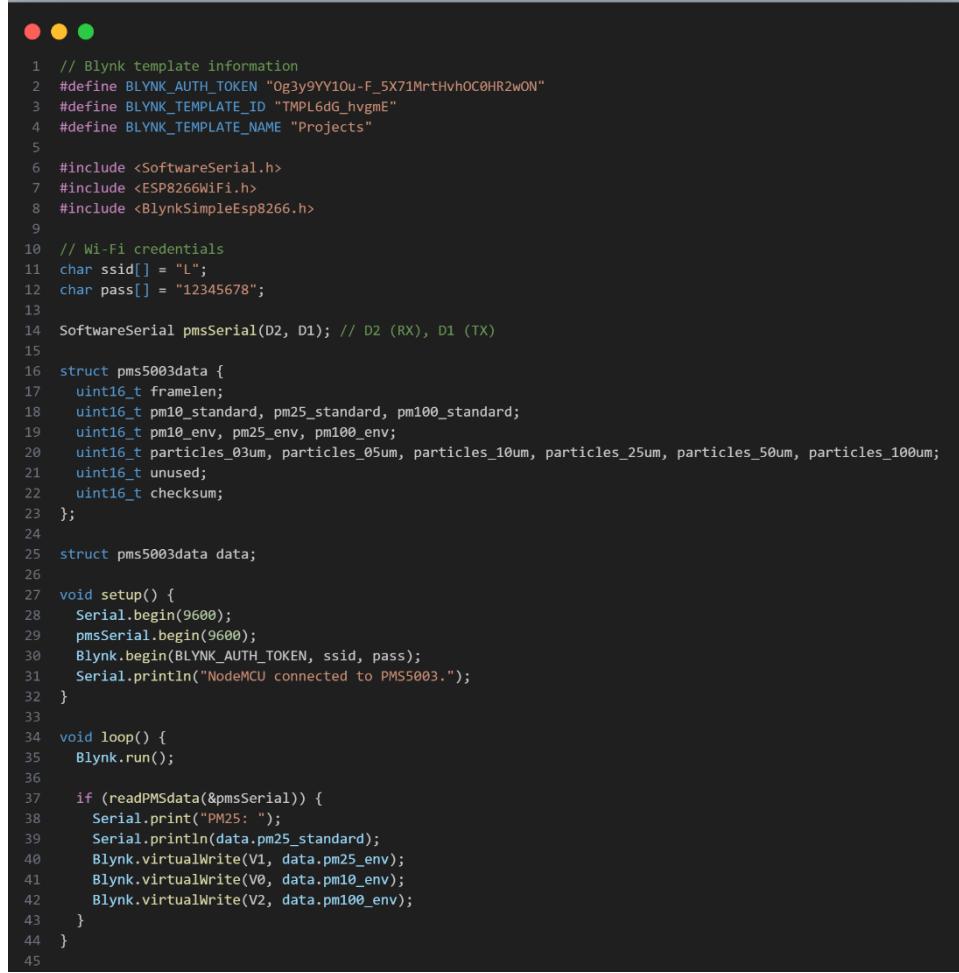
```

46 boolean readPMSdata(Stream *s) {
47   uint8_t buffer[32];
48   uint16_t sum = 0;
49   unsigned long start = millis();
50   while (s->available() < 1 || s->peek() != 0x42) {
51     if (s->available()) s->read();
52     if (millis() - start > 1000) {
53       Serial.println("Timeout waiting for start byte");
54       return false;
55     }
56   }
57   start = millis();
58   while (s->available() < 32) {
59     if (millis() - start > 1000) {
60       Serial.println("Incomplete data packet, skipping...");
61       return false;
62     }
63   }
64
65 s->readBytes(buffer, 32);
66 for (uint8_t i = 0; i < 30; i++) {
67   sum += buffer[i];
68 }
69 uint16_t receivedChecksum = (buffer[30] << 8) | buffer[31];
70 if (sum != receivedChecksum) {
71   Serial.println("Checksum mismatch, skipping packet");
72   return false;
73 }
74 data.framelen = (buffer[2] << 8) | buffer[3];
75 data.pm10_standard = (buffer[4] << 8) | buffer[5];
76 data.pm25_standard = (buffer[6] << 8) | buffer[7];
77 data.pm100_standard = (buffer[8] << 8) | buffer[9];
78 data.pm10_env = (buffer[10] << 8) | buffer[11];
79 data.pm25_env = (buffer[12] << 8) | buffer[13];
80 data.pm100_env = (buffer[14] << 8) | buffer[15];
81
82 return true;
83 }
84

```

ภาพที่ 34 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003

3.7.2.2 โค้ดสำหรับอุปกรณ์ NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk



```

1 // Blynk template information
2 #define BLYNK_AUTH_TOKEN "0g3y9YY10u-F_5X71MrthvhOC0HR2wON"
3 #define BLYNK_TEMPLATE_ID "TMPL6dG_hvgmE"
4 #define BLYNK_TEMPLATE_NAME "Projects"
5
6 #include <SoftwareSerial.h>
7 #include <ESP8266WiFi.h>
8 #include <BlynkSimpleEsp8266.h>
9
10 // Wi-Fi credentials
11 char ssid[] = "L";
12 char pass[] = "12345678";
13
14 SoftwareSerial pmsSerial(D2, D1); // D2 (RX), D1 (TX)
15
16 struct pms5003data {
17     uint16_t framelen;
18     uint16_t pm10_standard, pm25_standard, pm100_standard;
19     uint16_t pm10_env, pm25_env, pm100_env;
20     uint16_t particles_03um, particles_05um, particles_10um, particles_25um, particles_50um, particles_100um;
21     uint16_t unused;
22     uint16_t checksum;
23 };
24
25 struct pms5003data data;
26
27 void setup() {
28     Serial.begin(9600);
29     pmsSerial.begin(9600);
30     Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
31     Serial.println("NodeMCU connected to PMS5003.");
32 }
33
34 void loop() {
35     Blynk.run();
36
37     if (readPMSdata(&pmsSerial)) {
38         Serial.print("PM25: ");
39         Serial.println(data.pm25_standard);
40         Blynk.virtualWrite(V1, data.pm25_env);
41         Blynk.virtualWrite(V0, data.pm10_env);
42         Blynk.virtualWrite(V2, data.pm100_env);
43     }
44 }
45

```

ภาพที่ 35 โค้ดสำหรับอุปกรณ์ NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk

```

42 boolean readPMSSdata(Stream *s) {
43     uint8_t buffer[32];
44     uint16_t sum = 0;
45
46     unsigned long start = millis();
47
48     while (s->available() < 1 || s->peek() != 0x42) {
49         s->read();
50         if (millis() - start > 1000) {
51             Serial.println("Timeout waiting for start byte");
52             return false;
53         }
54     }
55     start = millis();
56     while (s->available() < 32) {
57         if (millis() - start > 1000) {
58             Serial.println("Incomplete data packet");
59             return false;
60         }
61     }
62     s->readBytes(buffer, 32);
63     for (uint8_t i = 0; i < 30; i++) {
64         sum += buffer[i];
65     }
66     uint16_t receivedChecksum = (buffer[30] << 8) | buffer[31];
67     if (sum != receivedChecksum) {
68         Serial.print("Checksum failure: calculated=");
69         Serial.print(sum, HEX);
70         Serial.print(" expected=");
71         Serial.println(receivedChecksum, HEX);
72         return false;
73     }
74     data.frameulen = (buffer[2] << 8) | buffer[3];
75     data.pm10_standard = (buffer[4] << 8) | buffer[5];
76     data.pm25_standard = (buffer[6] << 8) | buffer[7];
77     data.pm100_standard = (buffer[8] << 8) | buffer[9];
78     data.pm10_env = (buffer[10] << 8) | buffer[11];
79     data.pm25_env = (buffer[12] << 8) | buffer[13];
80     data.pm100_env = (buffer[14] << 8) | buffer[15];
81     data.particles_03um = (buffer[16] << 8) | buffer[17];
82     data.particles_05um = (buffer[18] << 8) | buffer[19];
83     data.particles_10um = (buffer[20] << 8) | buffer[21];
84     data.particles_25um = (buffer[22] << 8) | buffer[23];
85     data.particles_50um = (buffer[24] << 8) | buffer[25];
86     data.particles_100um = (buffer[26] << 8) | buffer[27];
87     data.unused = (buffer[28] << 8) | buffer[29];
88     data.checksum = receivedChecksum;
89
90     return true;
91 }

```

ภาพที่ 36 โค้ดสำหรับอุปกรณ์ NodeMCU ที่เข้ามต่อไปยัง PMS5003 และ Blynk

3.7.2.3 โค้ดสำหรับอุปกรณ์ Odroid C4 ที่เข้ามต่อไปยัง ThingSpeak และ Line Notify

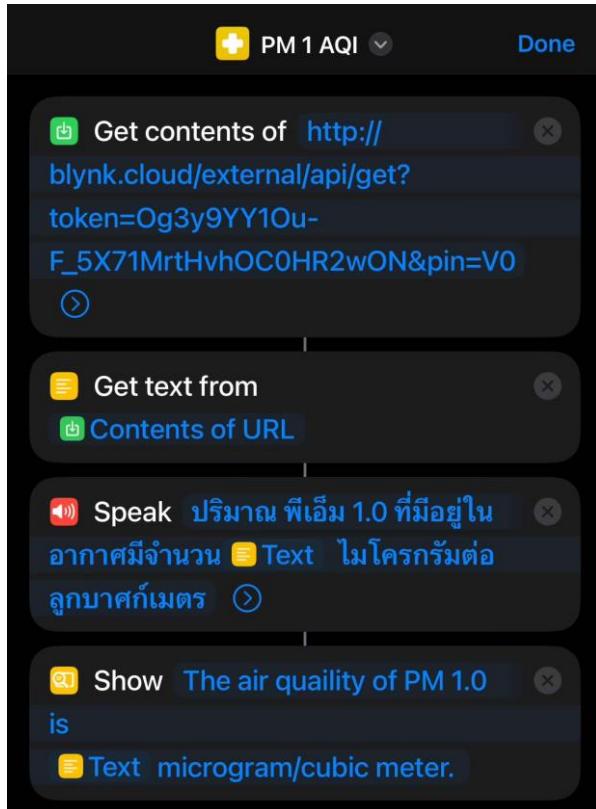
```

● ● ●

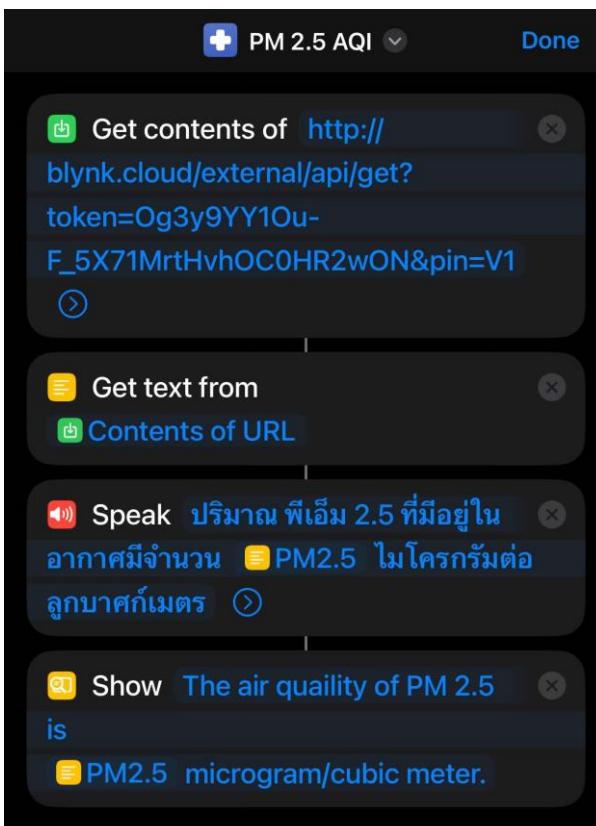
1 import serial
2 import requests
3 import time
4
5 # UART configuration
6 SERIAL_PORT = '/dev/ttyS1'
7 BAUD_RATE = 9600
8
9 # ThingSpeak configuration
10 THINGSPEAK_URL = f'https://api.thingspeak.com/update?api_key=PP37ZKIOJ1BI08J4&field1=0'
11
12 # Line Notify configuration
13 LINE_NOTIFY_URL = 'https://notify-api.line.me/api/notify'
14 LINE_NOTIFY_TOKEN = 'Z8WJVXV36iBhLiTqzcLRsIkqUNqWg0PYC0K9i4z6GWRY'
15
16 def send_to_thingspeak(pm25_value):
17     payload = {
18         'field1': pm25_value
19     }
20     try:
21         response = requests.post(THINGSPEAK_URL, params=payload)
22         if response.status_code == 200:
23             print(f'Data sent to ThingSpeak: PM2.5 = {pm25_value}')
24         else:
25             print(f'Failed to send data to ThingSpeak. Status code: {response.status_code}')
26     except Exception as e:
27         print(f'Error sending data to ThingSpeak: {e}')
28
29 def send_to_line_notify(pm25_value):
30     headers = {
31         'Authorization': f'Bearer {LINE_NOTIFY_TOKEN}'
32     }
33     payload = {
34         'message': f'PM2.5 Level: {pm25_value} ug/m3'
35     }
36     try:
37         response = requests.post(LINE_NOTIFY_URL, headers=headers, data=payload)
38         if response.status_code == 200:
39             print(f'PM2.5 notification sent to Line: {pm25_value} ug/m3')
40         else:
41             print(f'Failed to send notification to Line. Status code: {response.status_code}')
42     except Exception as e:
43         print(f'Error sending notification to Line: {e}')
44
45 def main():
46     ser = serial.Serial(SERIAL_PORT, BAUD_RATE, timeout=1)
47     print(f'Connected to {SERIAL_PORT} at {BAUD_RATE} baud.')
48     while True:
49         try:
50             line = ser.readline().decode('utf-8').strip()
51             if line.startswith("PM25:"):
52                 pm25_value = int(line.split(":")[1])
53                 print(f'Received PM2.5: {pm25_value} ug/m3')
54                 send_to_thingspeak(pm25_value)
55                 send_to_line_notify(pm25_value)
56             except Exception as e:
57                 print(f'Error: {e}')
58             time.sleep(15)
59
60 if __name__ == "__main__":
61     main()

```

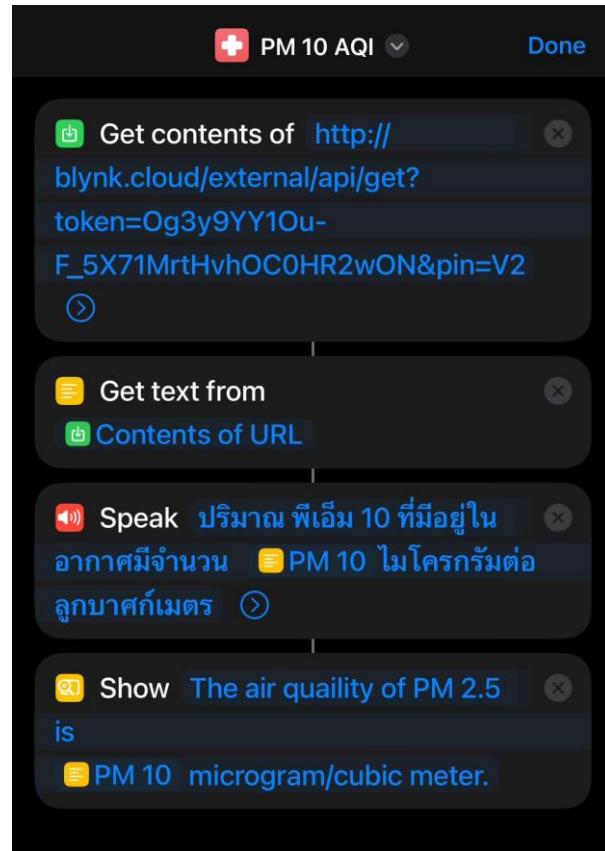
ภาพที่ 37 โค้ดสำหรับอุปกรณ์ Odroid C4 ที่เชื่อมต่อไปยัง ThingSpeak และ Line Notify



ภาพที่ 38 การแสดงค่า PM 1.0 ผ่านโปรแกรม Shortcut



ภาพที่ 39 การแสดงค่า PM 2.5 ผ่านโปรแกรม Shortcut



ภาพที่ 40 การแสดงค่า PM 10 ผ่านโปรแกรม Shortcut

3.7.3 ทดลอง และตรวจสอบประสิทธิภาพการใช้งานของอุปกรณ์ดังกล่าวทั้งหมด

3.8 ปฏิบัติขั้นตอนในการสร้างอุปกรณ์ในการแยกเหรียญ

3.8.1 ต่อ Arduino UNO R3 ให้สามารถนับเหรียญได้ โดยให้ทำงานร่วมกับ Counter

Module Motor Speed Sensor และเขียนโค้ดให้สามารถแยกเหรียญ รีเซ็ตค่า
เหรียญ พร้อมทั้งมีระบบนาฬิกา และตั้งเวลาได้

3.8.1.1 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เขียนต่อไปยัง Counter Module

Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574
ที่ต่อ กับ Keypad

```

1 #include <avr/io.h>
2 #include <avr/interrupt.h>
3 #include <avr/sleep.h>
4 #include <EEPROM.h>
5 #include <Wire.h>
6 #include <LiquidCrystal_I2C.h>
7 #include <Keypad_I2C.h>
8 #include <Keypad.h>
9 #include <TimeLib.h>
10 #include <DS1307RTC.h>
11
12 // Pin definitions for coin detector
13 const int sensorPins[] = {8, 9, 10, 11}; // Sensor pins connected to Arduino
14 const int buttonPin = 2; // Pin for the reset button
15 const int numSensors = 4; // Number of sensors
16 int sensorStates[numSensors]; // Variable to store the current state of sensors
17 int lastSensorStates[numSensors]; // Variable to store the last state of sensors
18 const int coinValues[] = {1, 2, 5, 10};
19 int totalAmount = 0; // Variable to store the total amount
20 int lastTotalAmount = 0; // Variable to store the last total amount
21 bool coinDetected = false; // Variable to track coin detection
22
23 // LCD setup
24 #define LCD_ADDRESS 0x27
25 LiquidCrystal_I2C lcd(LCD_ADDRESS, 16, 2);
26
27 // Keypad setup
28 #define I2CADDR 0x20
29 const byte ROWS = 4;
30 const byte COLS = 4;
31 char hexaKeys[ROWS][COLS] = {
32     {'1', '2', '3', 'A'},
33     {'4', '5', '6', 'B'},
34     {'7', '8', '9', 'C'},
35     {'*', '0', '#', 'D'}
36 };
37 byte rowPins[ROWS] = {7, 6, 5, 4};
38 byte colPins[COLS] = {3, 2, 1, 0};
39 unsigned long last_time = 0;
40 bool blink_txt;
41 Keypad_I2C keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS, I2CADDR);
42

```

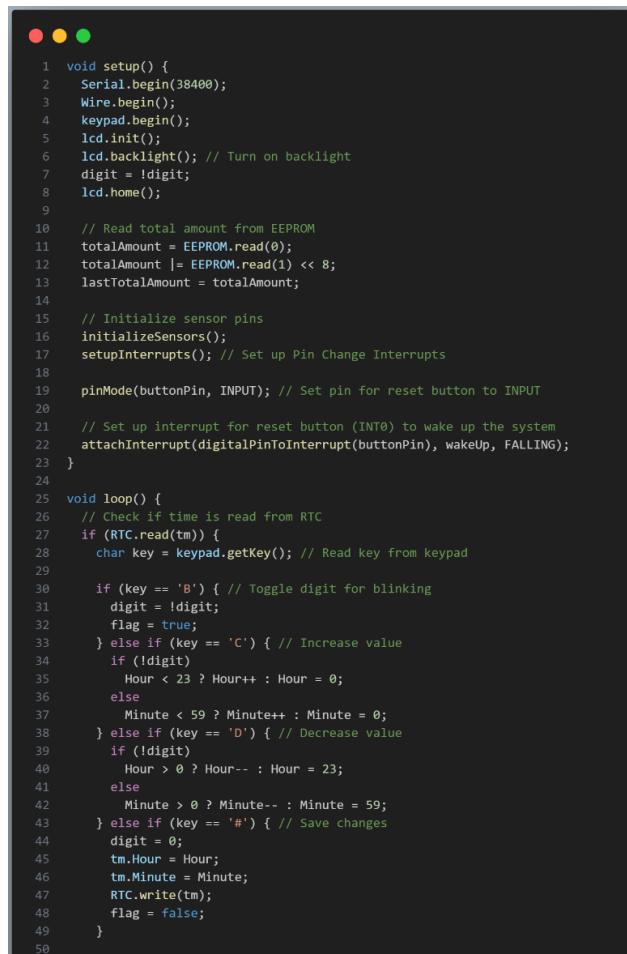
ภาพที่ 41 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อกับ Keypad

```

43 // Time tracking
44 unsigned long lastActivityTime = 0; // Time when the last activity occurred
45 const unsigned long sleepDelay = 20000; // Time to wait before entering sleep mode (20 seconds)
46
47 // Timekeeping
48 tmElements_t tm; // for DS1307
49 int Hour, Minute;
50
51 unsigned int digit;
52 bool flag = false;
53 bool blink_tx = false;
54
55 void print2digit(int val){
56     if(val < 10 && val >= 0)
57         lcd.print(0);
58     lcd.print(val);
59 }
60
61 // Sleep mode settings
62 #define SLEEP_IDLE 0
63 #define SLEEP_ADC_REDUCE 1
64 #define SLEEP_POWER_DOWN 2
65 #define SLEEP_POWER_SAVE 3
66 #define SLEEP_STANDBY 6
67 #define SLEEP_EXT_STANDBY 7
68

```

ภาพที่ 42 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad



```

1 void setup() {
2     Serial.begin(38400);
3     Wire.begin();
4     keypad.begin();
5     lcd.init();
6     lcd.backlight(); // Turn on backlight
7     digit = !digit;
8     lcd.home();
9
10    // Read total amount from EEPROM
11    totalAmount = EEPROM.read(0);
12    totalAmount |= EEPROM.read(1) << 8;
13    lastTotalAmount = totalAmount;
14
15    // Initialize sensor pins
16    initializeSensors();
17    setupInterrupts(); // Set up Pin Change Interrupts
18
19    pinMode(buttonPin, INPUT); // Set pin for reset button to INPUT
20
21    // Set up interrupt for reset button (INT0) to wake up the system
22    attachInterrupt(digitalPinToInterrupt(buttonPin), wakeUp, FALLING);
23 }
24
25 void loop() {
26    // Check if time is read from RTC
27    if (RTC.read(tm)) {
28        char key = keypad.getKey(); // Read key from keypad
29
30        if (key == 'B') { // Toggle digit for blinking
31            digit = !digit;
32            flag = true;
33        } else if (key == 'C') { // Increase value
34            if (!digit)
35                Hour < 23 ? Hour++ : Hour = 0;
36            else
37                Minute < 59 ? Minute++ : Minute = 0;
38        } else if (key == 'D') { // Decrease value
39            if (!digit)
40                Hour > 0 ? Hour-- : Hour = 23;
41            else
42                Minute > 0 ? Minute-- : Minute = 59;
43        } else if (key == '#') { // Save changes
44            digit = 0;
45            tm.Hour = Hour;
46            tm.Minute = Minute;
47            RTC.write(tm);
48            flag = false;
49    }
50 }

```

ภาพที่ 43 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad

```

51     // Update time display
52     lcd.setCursor(0, 0);
53     lcd.print("Total: ");
54     lcd.print(totalAmount); // Print total amount
55     lcd.print(" Baht "); // Print the currency
56     lcd.setCursor(0, 1); // Set cursor to the second line
57     lcd.print("Time: ");
58     if (flag) { // If B was pressed
59         if ((millis() - last_time) > 1000) { // Blink every 1 second
60             blink_tx = !blink_tx;
61             last_time = millis();
62         }
63         if (!digit) { // Blinking hour
64             if (blink_tx)
65                 lcd.print(" ");
66             else
67                 print2digit(Hour);
68             lcd.print(":");
69             print2digit(Minute);
70         } else { // Blinking minute
71             print2digit(Hour);
72             lcd.print(":");
73             if (blink_tx)
74                 lcd.print(" ");
75             else
76                 print2digit(Minute);
77         }
78     } else { // No blinking
79         Hour = tm.Hour;
80         Minute = tm.Minute;
81         print2digit(Hour);
82         lcd.print(":");
83         print2digit(Minute);
84     }
85 }
86
87 // Check if the reset button is pressed
88 if (digitalRead(buttonPin) == HIGH) {
89     resetTotalAmount(); // Reset total amount and EEPROM
90     lastActivityTime = millis(); // Update last activity time
91 }
92
93 // Update EEPROM only when totalAmount changes
94 if (totalAmount != lastTotalAmount) {
95     EEPROM.write(0, totalAmount & 0xFF);
96     EEPROM.write(1, (totalAmount >> 8) & 0xFF);
97     lastTotalAmount = totalAmount; // Update last total amount
98     lastActivityTime = millis(); // Update last activity time
99 }
100
101 // Update LCD if a coin is detected
102 if (coinDetected) {
103     lcd.backLight();
104     updateLCD(totalAmount);
105     coinDetected = false; // Reset coin detection flag
106     lastActivityTime = millis(); // Update last activity time
107 } else if (millis() - lastActivityTime >= sleepDelay) {
108     // If no activity for 5 seconds, enter sleep mode
109     enterSleepMode();
110     lastActivityTime = millis(); // Reset last activity time after waking up
111 }
112 }
```

ภาพที่ 44 โค้ดสำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad

```

1 void initializeSensors() {
2     for (int i = 0; i < numSensors; i++) {
3         pinMode(sensorPins[i], INPUT);
4         digitalWrite(sensorPins[i], HIGH); // Enable pull-up resistors
5
6         // កំណត់តារីពីលើកនា lastSensorStates[]
7         lastSensorStates[i] = digitalRead(sensorPins[i]);
8     }
9 }
10
11
12 // Function to set up interrupts
13 void setupInterrupts() {
14     sei(); // Enable global interrupts
15     PCICR |= (1 << PCIE0); // Enable Pin Change Interrupt for PCINT0..7
16     PCMSK0 |= (1 << PCINT0) | (1 << PCINT1) | (1 << PCINT2) | (1 << PCINT3); // Enable interrupts for pins 8, 9, 10, 11
17 }
18
19 // ការរាយអង្គភិជំនួយ (500ms)
20 const unsigned long debounceDelay = 500;
21 unsigned long lastDetectionTime[numSensors]; // បានពិភ័យតែទូទាត់ខ្លួនដើម្បីត្រូវសុចរាប់
22
23 // Interrupt Service Routine for Pin Change Interrupt
24 ISR(PCINT0_vect) {
25     unsigned long currentTime = millis();
26
27     for (int i = 0; i < numSensors; i++) {
28         // ត្រូវសុចរាប់អង្គភាពនៅពេលត្រូវបានបញ្ជូនឡើង
29         if (currentTime - lastDetectionTime[i] > debounceDelay) {
30             // គ្រាមឈប់ការបញ្ជូនដើម្បីត្រូវបានបញ្ជូនឡើង
31             if ((i < 3 && digitalRead(sensorPins[i]) == LOW && lastSensorStates[i] == HIGH) ||
32                 (i == 3 && digitalRead(sensorPins[i]) == HIGH && lastSensorStates[i] == LOW)) {
33                 totalAmount += coinValues[i];
34                 coinDetected = true; // ត្រូវបានបញ្ជូនឡើងដើម្បីត្រូវបានបញ្ជូនឡើង
35
36                 Serial.print("Coin of ");
37                 Serial.print(coinValues[i]);
38                 Serial.println(" detected!");
39
40                 // ប៉ូល Serial
41                 Serial.print("Updated Total amount: ");
42                 Serial.println(totalAmount);
43
44                 // បានពិភ័យតែទូទាត់ខ្លួនដើម្បីត្រូវបានបញ្ជូនឡើង
45                 lastDetectionTime[i] = currentTime;
46             }
47         }
48         lastSensorStates[i] = digitalRead(sensorPins[i]);
49     }
50 }
51
52
53 // Function to update the LCD with the total amount
54 void updateLCD(int amount) {
55     lcd.setCursor(0, 0); // Set cursor to the first line
56     lcd.print("Total: "); // Print text
57     lcd.print(amount); // Print total amount
58     lcd.print(" Baht "); // Print the currency
59 }
60
61 // Function to reset total amount and EEPROM
62 void resetTotalAmount() {
63     totalAmount = 0; // Reset total amount
64     EEPROM.write(0, 0); // Reset EEPROM value
65     EEPROM.write(1, 0);
66     lastTotalAmount = totalAmount; // Update last total amount
67
68     // Update LCD to reflect reset
69     updateLCD(totalAmount);
70 }
71
72 // Function to enter sleep mode
73 void enterSleepMode() {
74     // Disable ADC to save power
75     ADCSRA &= ~(1 << ADEN);
76
77     // Set sleep mode to power-save
78     set_sleep_mode(SLEEP_POWER_SAVE);
79     sleep_enable();
80
81     // Turn off LCD backlight before sleeping
82     lcd.noBacklight();
83
84     // Enter sleep mode
85     sleep_cpu();
86
87     // Wake up and re-enable ADC
88     sleep_disable();
89     ADCSRA |= (1 << ADEN);
90
91     // Re-initialize the LCD if necessary
92     lcd.begin(16, 2);
93     lcd.backlight();
94     updateLCD(totalAmount);
95 }
96
97 // Function to wake up the system from sleep mode
98 void wakeUp() {
99     // This function is intentionally empty
100    // The system will wake up automatically when the interrupt is triggered
101 }
102

```

រាយទី 45 គឺជា អ៊ូរូបក្រុម Arduino UNO R3 ដែលមានពីរក្រុម Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button និង PCF8574 ដែលត្រូវបានបញ្ជូនឡើង

3.8.2 สร้าง Interrupt แบบ Pin Change ให้กับตัว Sensor เพื่อให้อุปกรณ์ไม่ต้อง
จำเป็นต้องตรวจสอบ หรือรับค่าตลอดเวลา

3.8.3 ทำการเชื่อมต่อ กับ I2C LCD 16x2

3.8.4 สร้างปุ่ม Reset จำนวนเงินที่ถูกหยุดลงไปในระบบปุกออมสิน

3.8.5 นำ EEPROM มาใช้งาน เพื่อป้องกันปัญหาเงินที่แสดงผ่าน LCD สูญหาย

3.8.6 ทำการสร้าง Sleep Mode เพื่อลดการใช้ไฟของตัวอุปกรณ์

3.8.7 ทำให้ปุ่ม Reset เป็น Interrupt ในการปิดใช้งาน Sleep Mode ของอุปกรณ์

3.8.8 ทำงานพิเศษ และปุ่มตั้งเวลาจาก Keypad และ DS1307

4. รายละเอียดเกี่ยวกับโค้ดของแต่ละอุปกรณ์ภายใน

4.1 Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003

โค้ดมีรายละเอียด ดังนี้ ทำการสร้างการเชื่อมต่อในรูปแบบของ Receive-Transmit (Rx-Tx) โดยทำการเชื่อมต่อเข้าบอร์ด Odroid โดยให้ Pin 0 และ 1 เป็นขา Rx และ Tx ตามลำดับ, และ เชื่อมต่อเข้าเซนเซอร์ PMS5003 โดยให้ Pin 2 และ 3 เป็นขา Rx และ Tx ตามลำดับ สุดท้าย ทำการเชื่อมต่อ กับ I2C LCD 16*2 โดยการใช้ address ที่ 0x27 (Pin A4, A5)

ในส่วนของการทำ setup มีการแสดงผลผ่าน Serial Monitor และการติดต่อไปยัง PMS5003 โดยการใช้ Baud Rate ที่ 9600 และทำให้จอ LCD รีเซ็ตเป็นจอเปล่า ๆ ก่อนที่จะทำการแสดงผล ค่าได ๆ ส่วนของ struct pms5003data เป็นการเก็บค่า Particulate Matter มาจาก PMS5003 ในรูปแบบ uint16_t และเก็บในตัวแปร data

ในส่วนของ loop จะทำการตรวจสอบว่าถ้าอ่านค่าจาก pmsSerial จากฟังก์ชัน readPMSdata หากเจอข้อมูล จะทำการแสดงค่า PM 2.5 ในสภาพแวดล้อมที่อยู่ผ่าน Serial Monitor และส่ง ข้อมูลไปยัง Odroid จากนั้นจอ LCD จะแสดงข้อมูลของค่า PM 2.5 ในสภาพแวดล้อมที่อยู่ เช่นกัน

ในส่วนของ readPMSdata จะทำการอ่านข้อมูลจาก PMS5003 ที่จะแสดงค่า PM 1.0, 2.5, 10 และอื่น ๆ ซึ่งต้องมีการประมวลผลข้อมูลขนาด 32 ไบต์จากตัวเซนเซอร์เอง มีการตรวจสอบ errors และใส่ข้อมูลที่ได้รับจากเซนเซอร์ลงใน

4.2 NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk

โค้ดมีรายละเอียด ดังนี้ ทำการประกาศตัว Blynk Token, ID, Name ที่ได้มาจากการตั้งค่าใน Blynk และ เรียก Header files สำหรับ NodeMCU เพื่อให้สามารถเชื่อมต่อกับ Blynk ได้ เริ่มต้นด้วยการ เชื่อมต่อ Wi-Fi ให้กับ NodeMCU โดยมี ssid และ pass และทำการสร้างการเชื่อมต่อในรูปแบบของ Receive-Transmit (Rx-Tx) โดยทำการเชื่อมต่อเข้า PMS5003 โดยให้ Pin D2 และ D1 เป็นขา Rx และ Tx ตามลำดับ

ส่วนของ struct pms5003data เป็นการเก็บค่า Particulate Matter มาจาก PMS5003 ในรูปแบบ uint16_t แล้วเก็บในตัวแปร data ในส่วนของการทำ setup มีการแสดงผลผ่าน Serial Monitor และการติดต่อไปยัง PMS5003 โดยการใช้ Baud Rate ที่ 9600 และทำการ initialize Blynk ขึ้นมา โดยอ้างอิงจากตัวแปร Token, ssid, pass ก่อนหน้านี้

ในส่วนของ loop จะทำ run ตัว Blynk และทำการตรวจสอบว่าถ้าอ่านค่าจาก pmsSerial จากฟังก์ชัน readPMSdata หากเจอข้อมูล จะทำการแสดงค่า PM 2.5 ในสภาพแวดล้อมที่อยู่ผ่าน Serial Monitor และส่งข้อมูลไปยัง Blynk ทั้ง PM 1.0, 2.5 และ 10

ในส่วนของ readPMSdata จะทำการอ่านข้อมูลจาก PMS5003 ที่จะแสดงค่า PM 1.0, 2.5, 10 และอื่น ๆ ซึ่งต้องมีการประมวลผลข้อมูลขนาด 32 ไบต์จากตัวเซนเซอร์เอง มีการตรวจสอบ errors และใส่ข้อมูลที่ได้รับจากเซนเซอร์ลงไว้

4.3 Odroid C4 ที่เชื่อมต่อไปยัง ThingSpeak และ Line Notify

ทำการนำเข้า serial, requests, time และทำการเรียกใช้ port uart ttyS1 และสร้างตัวแปร BAUD_RATE ที่ 9600 เพื่อการติดต่อระหว่างอุปกรณ์ จากนั้นทำการใช้ URL ของ ThingSpeak ที่นำมาใช้ใน Projects และใช้ URL ของ Line Notify และดึง Token มาใช้

ส่วนของ sent_to_thingspeak เป็นส่วนการส่งข้อมูล PM 2.5 ไปยัง ThingSpeak โดยมี headers ที่เก็บ Line Notify Token ที่ไว้ในการ Authorization, payload ที่เก็บ message ที่มีค่า PM 2.5 อยู่, requests.post เป็นการส่ง HTTP Request POST ไปยัง Line Notify API ใน try exception เพื่อจัดการกับ errors

ส่วนของ main มีตัวแปร ser ในการเปิดใช้งาน Serial Port จาก SERIAL_PORT, BAUD_RATE และ timeout = 1 ที่หมายถึงมีการรอ 1 วินาทีก่อนจะอ่าน input, line ในการอ่านค่าจาก Serial port นั้น แล้วทำการ decode ให้กลายเป็น string และลบค่าว่าง (Whitespace) จากนั้นนำไปเข้าเงื่อนไขว่าข้อมูลจาก line มีคำว่า “PM25:” ก็จะนำค่าตัวเลขดังกล่าวมาแสดงผ่านการ print และทำการส่งไปยัง ThingSpeak และ Line Notify และมีการทำ time.sleep(15) ในการรอการทำซ้ำในอีก 15 วินาทีถัดไปก่อนที่จะอ่านข้อมูลตัวถัดไป

4.4 Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor,

I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อgether Keypad

โค้ดส่วนกระปุกคอมสินเป็นโปรแกรมที่ตรวจจับเหวี่ยงที่ใช้ Arduino โดยมีฟังก์ชันแสดงผลยอดรวมของเหวี่ยงที่ตรวจจับได้บนจอ LCD และสามารถตั้งเวลาและแสดงได้โดยใช้ RTC (Real

Time Clock) โมดูล DS1307 และยังมีฟังก์ชันประยัดพลังงานโดยเข้าสู่โหมด sleep เมื่อไม่มีการทำงานในช่วงเวลาหนึ่ง 34

4.4.1 การกำหนดขาและตัวแปร (Pin and Variable Definitions)

sensorPins[] เป็นอาร์เรย์ใช้เก็บตำแหน่งขาของเซ็นเซอร์ (counter module motor speed sensor) ที่เชื่อมต่อกับบอร์ด Arduino มีทั้งหมด 4 ขา ได้แก่ 8, 9, 10, และ 11 ซึ่งแต่ละขาจะเชื่อมต่อกับเซ็นเซอร์ สำหรับตรวจจับเหรียญ 1, 2, 5, 10

buttonPin ใช้สำหรับขาที่เชื่อมต่อกับปุ่มกดรีเซ็ตยอดรวมของเหรียญ โดยเชื่อมต่อกับขา 2

sensorStates[] และ lastSensorStates[] ใช้สำหรับเก็บสถานะปัจจุบันและสถานะก่อนหน้าของเซ็นเซอร์ เพื่อตรวจจับการเปลี่ยนแปลงสถานะ ของเหรียญว่าเหรียญถูกหยุดหรือไม่

coinValues[] ใช้ในการเก็บค่าเหรียญตามตำแหน่งของเซ็นเซอร์ เช่น เซ็นเซอร์ขาที่ 8 ตรวจจับเหรียญ 1 บาท, เซ็นเซอร์ขาที่ 9 ตรวจจับเหรียญ 2 บาท, เซ็นเซอร์ขาที่ 10 ตรวจจับเหรียญ 5 บาท และเซ็นเซอร์ขาที่ 11 ตรวจจับเหรียญ 10 บาท

totalAmount ใช้สำหรับเก็บยอดรวมของเหรียญที่ถูกตรวจจับทั้งหมด

coinDetected ใช้สำหรับตรวจสอบว่ามีการตรวจจับเหรียญใหม่หรือไม่ เมื่อเซ็นเซอร์ตรวจจับเหรียญแล้วจะเปลี่ยนค่านี้เป็น true

4.4.2 การตั้งค่าจอ LCD และคีย์แพด (LCD and Keypad Setup)

LiquidCrystal_I2C lcd() ใช้สำหรับการเชื่อมต่อจอ LCD แบบ I2C ที่อยู่ที่ address 0x27 โดยจะมีขนาด 16x2 (16 ตัวอักษรต่อบรรทัด, 2 บรรทัด)

ในส่วนของคีย์แพดต่อกับ PCF8574 เพื่อรับข้อมูลแบบ I2C ใช้ address 0x20 และกำหนดปุ่มเป็นตาราง 4x4 ซึ่งแต่ละปุ่มที่ใช้จะมีแค่ตัวอักษร โดยปุ่ม B ใช้สำหรับเลือกตำแหน่งว่าจะเปลี่ยนตำแหน่งชั่วโมงหรือนาที, โดยปุ่ม C ใช้สำหรับเพิ่มเวลา, โดยปุ่ม D ใช้สำหรับลดเวลา และปุ่ม # ใช้สำหรับเลือกเวลาที่ถูกต้อง

4.4.3 การทำงานในฟังก์ชัน setup()

ในฟังก์ชัน setup() จะเริ่มการทำงานของ Serial สำหรับการดีบัก, I2C, จอ LCD และคีย์แพด

ค่าของ totalAmount (ยอดรวมของเหรียญ) จะถูกอ่านจาก EEPROM ที่บันทึกไว้ก่อนหน้าเพื่อนำมาใช้ในโปรแกรม

ในฟังก์ชัน initializeSensors() จะตั้งค่าขาอินพุตของเซ็นเซอร์ (counter module motor speed sensor) และ setupInterrupts() จะตั้งค่าการใช้งานขัดจังหวะ (Interrupts) เพื่อให้ Arduino ตรวจจับการเปลี่ยนแปลงของเซ็นเซอร์

ในปุ่มรีเซ็ต (buttonPin) จะถูกตั้งค่าให้ใช้ขัดจังหวะ (Interrupts) เพื่อปลุกระบบเมื่อกดปุ่ม

4.4.4 การทำงานในฟังก์ชัน loop() (Main Logic in the loop() Function)

ทำการอ่านเวลา (RTC.read()) จากโมดูล DS1307 และเพื่อทำการแสดงผลเวลาปัจจุบันและแสดงบนจอ LCD

ในปุ่มคีย์แพด B, C, D, และ # จะใช้ในการตั้งค่าและปรับเวลา (เพิ่ม/ลดชั่วโมงและนาที) โดยปุ่ม B ใช้สำหรับเลือกตำแหน่งว่าจะเปลี่ยนตำแหน่งชั่วโมงหรือนาที, โดยปุ่ม C ใช้สำหรับเพิ่มเวลา, โดยปุ่ม D ใช้สำหรับลดเวลา และปุ่ม # ใช้สำหรับเลือกเวลาที่ถูกต้อง

การอัปเดตค่าของเหรียญหรือค่า totalAmount (ยอดรวมเหรียญ) บนจอ LCD

4.4.5 การตั้งค่าขัดจังหวะ (Interrupts Setup)

ใช้ sei() เพื่อเปิดใช้งานการขัดจังหวะทั่วทั้งระบบ (Global Interrupt)

ใช้ PCICR และ PCMSK0 เพื่อทำการเปิดใช้งาน Pin Change Interrupt ในขา 8, 9, 10, 11 ซึ่งเชื่อมต่อกับเซ็นเซอร์ (counter module motor speed sensor) เพื่อตรวจจับเหรียญ

4.4.6 การทำงานของฟังก์ชัน ISR() (Interrupt Service Routine) สำหรับการตรวจจับเหรียญ

ฟังก์ชันนี้เป็นการตอบสนอง Interrupt ที่เกิดขึ้นเมื่อเซ็นเซอร์ตรวจจับเหรียญ โดยฟังก์ชันนี้จะทำงานทันทีที่เซ็นเซอร์ตรวจจับการเปลี่ยนสถานะของขาที่เชื่อมต่อกับเหรียญ

สำหรับเซ็นเซอร์ (counter module motor speed sensor) 3 ตัวแรก (ขา 8, 9, 10) จะตรวจจับเมื่อขาเปลี่ยนสถานะจาก HIGH เป็น LOW (เมื่อตรวจจับเหรียญได้) ส่วนเซ็นเซอร์ (counter module motor speed sensor) ตัวสุดท้าย (ขา 11) จะตรวจจับเมื่อขาเปลี่ยนจาก LOW เป็น HIGH

เมื่อมีเหรียญถูกตรวจจับ จะเพิ่มค่ามูลค่าเหรียญลงใน totalAmount และแสดงข้อมูลการตรวจจับเหรียญผ่าน Serial Monitor

จากนั้นจะทำการอัปเดตสถานะล่าสุดของเซ็นเซอร์ลงในตัวแปร lastSensorStates[] เพื่อเตรียมพร้อมสำหรับการตรวจจับครั้งถัดไป

4.4.7 ฟังก์ชันการอัปเดตจอ LCD (Updating the LCD Display)

ฟังก์ชันนี้ทำหน้าที่อัปเดตยอดรวมของเหรียญที่ตรวจจับได้บนจอ LCD โดยจะแสดงผลในรูปแบบ "Total: [ยอดรวมของเหรียญ] Baht" โดยจะแสดงบนบรรทัดแรกของจอ

เรียกใช้ฟังก์ชันนี้ทุกครั้งเมื่อมีการหยุดรวม และเหรียญจะถูกอัปเดตและมีการเปลี่ยนแปลง

4.4.8 การรีเซ็ตยอดรวมเหรียญและ EEPROM (Resetting Total Amount and EEPROM)

ฟังก์ชันนี้ทำหน้าที่รีเซ็ตยอดรวมเหรียญทั้งหมด โดยรีเซ็ตทั้งตัวแปร totalAmount และข้อมูลที่เก็บอยู่ใน EEPROM

เมื่อรีเซ็ตเสร็จแล้ว จะอัปเดตจอ LCD เพื่อแสดงยอดรวมเหรียญเป็น 0

4.4.9 การเข้าสู่โหมดพักเครื่อง (Entering Sleep Mode)

เมื่อไม่มีการทำงานของวงจร ในระยะเวลาที่กำหนด (20 วินาที) โปรแกรมจะเข้าสู่โหมดพักเครื่อง (sleep mode) เพื่อทำการประหยัดพลังงาน

ก่อนเข้าสู่โหมดพัก (sleep mode) เครื่องจะปิดการใช้งาน ADC (Analog-to-Digital Converter) และไฟพื้นหลังของจอ LCD เพื่อช่วยลดการใช้พลังงาน

หลังจากถูกปลุกโดยการกดปุ่มหรือมีการหยุดเหรียญ เช่นเซอร์จะทำการตรวจจับเหรียญ ระบบจะกลับมาทำงานตามปกติและเปิดการใช้งาน ADC รวมถึงไฟพื้นหลังของจอ LCD อีกครั้ง

4.4.10 ฟังก์ชันสำหรับปลุกระบบ (Waking Up the System)

ฟังก์ชันนี้ถูกกำหนดให้ทำงานเมื่อเกิดเหตุการณ์ Interrupts โดยการกดปุ่มรีเซ็ต โดยฟังก์ชันนี้ไม่จำเป็นต้องมีการทำงานใดๆ เพราะเมื่อเกิดการขัดจังหวะ ระบบจะตื่นขึ้นเอง

4.5 สรุปภาพรวมการทำงานของโค้ด:

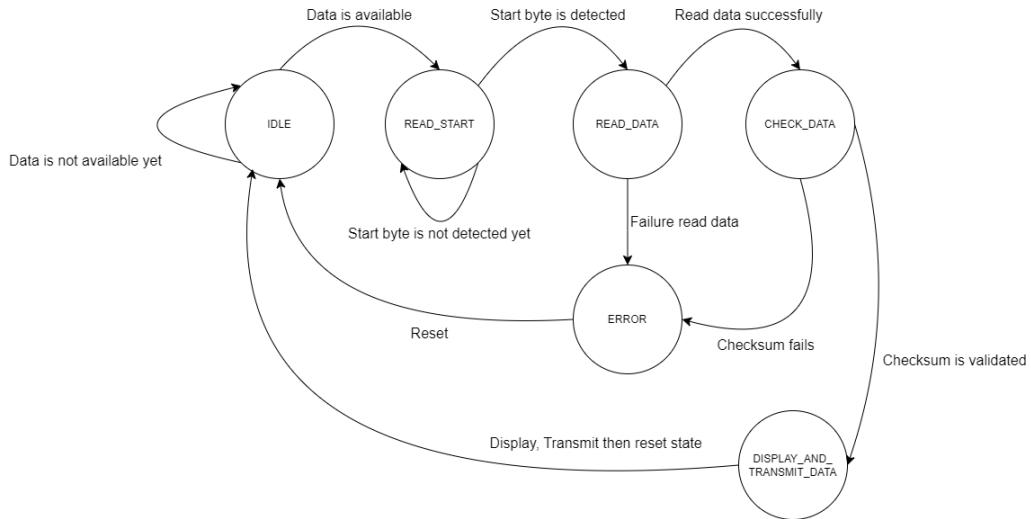
โปรแกรมนี้ใช้เซ็นเซอร์ (counter module motor speed sensor) 4 ตัวในการตรวจจับเหรียญที่ถูกหยุด โดยแต่ละเซ็นเซอร์จะตรวจจับเหรียญที่มีมูลค่าต่างกัน (1, 2, 5, และ 10 บาท)

ข้อมูลยอดรวมเหรียญจะถูกบันทึกใน EEPROM เพื่อให้สามารถบันทึกข้อมูลตอนและเรียกดูได้แม้ปิดเครื่อง ผู้ใช้งานสามารถตั้งเวลาและแสดงผลได้บนจอ LCD

หากไม่มีการใช้งานใดๆ เป็นเวลา 20 วินาที ระบบจะเข้าสู่โหมดพักเครื่อง (Sleep Mode) เพื่อประหยัดพลังงาน

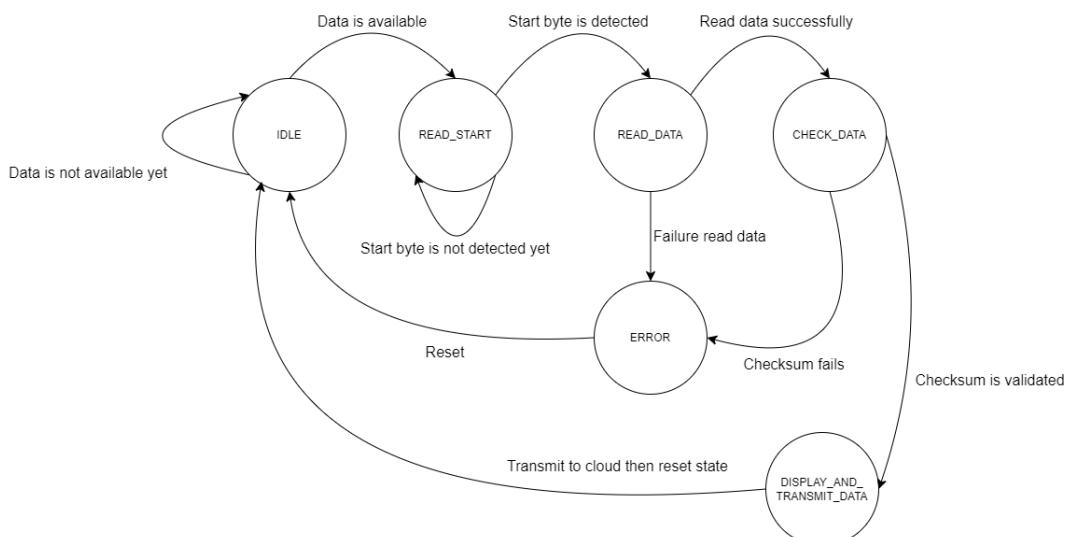
5. State Machine

5.1 Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003



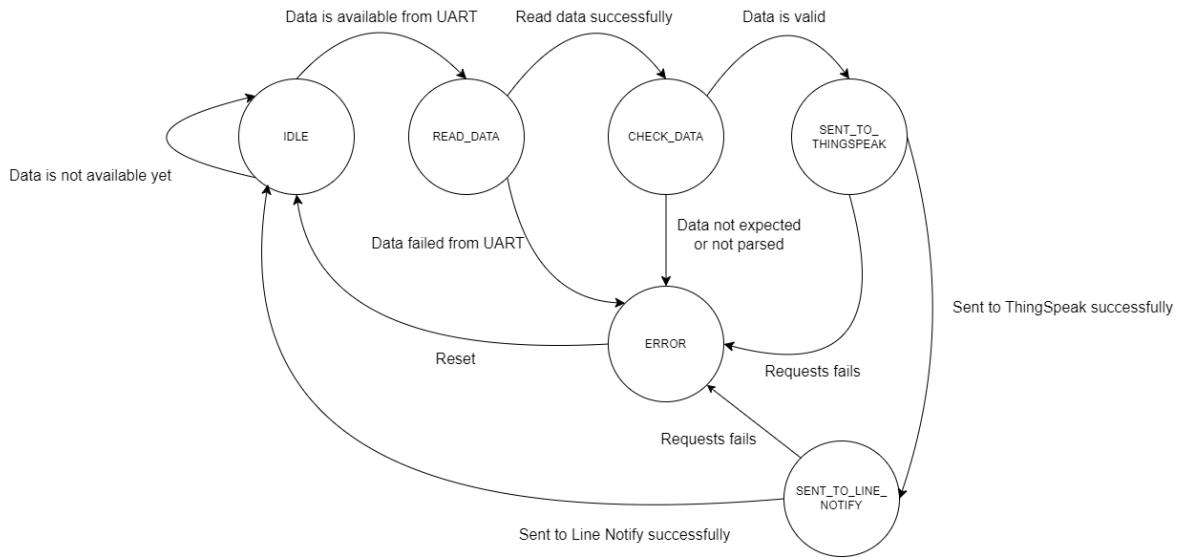
ภาพที่ 46 State Diagram สำหรับอุปกรณ์ Arduino UNO R3 ที่เชื่อมต่อไปยัง Odroid C4 และ PMS5003

5.2 NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk



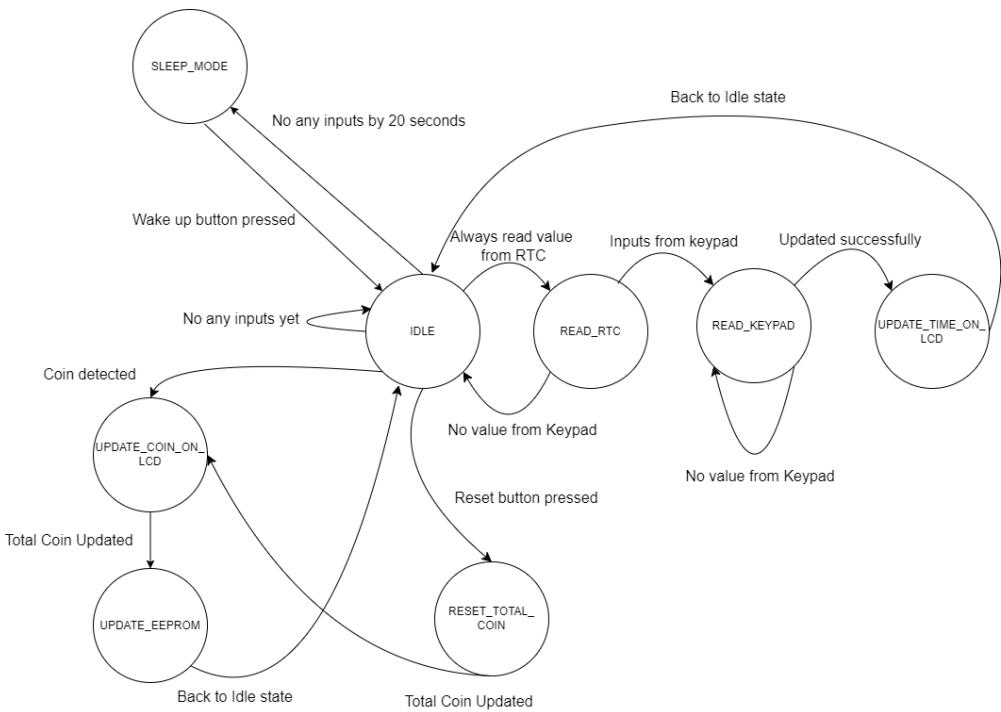
ภาพที่ 47 State Diagram สำหรับอุปกรณ์ NodeMCU ที่เชื่อมต่อไปยัง PMS5003 และ Blynk

5.3 Odroid C4 ที่เชื่อมต่อไปยัง ThingSpeak และ Line Notify



ภาพที่ 48 State Diagram สำหรับอุปกรณ์ Odroid C4 ที่เชื่อมต่อไปยัง ThingSpeak และ Line Notify

5.4 Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad



ภาพที่ 49 Arduino UNO R3 ที่เชื่อมต่อไปยัง Counter Module Motor Speed Sensor, I2C LCD 16x2, DS1307, Button และ PCF8574 ที่ต่อ กับ Keypad

บทที่ 4

ผลการดำเนินงาน

โครงการเรื่อง Piggy Bank Saving Your Life ทางผู้จัดทำได้รับผลดำเนินงาน ดังต่อไปนี้

4.1 เพื่อศึกษาการเขียนโปรแกรม และศึกษาการทำงานของบอร์ด Arduino ในการแยกเหรียญ

1 Baht	2 Baht	5 Baht	10 Baht
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	FALSE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	TRUE	FALSE
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE
TRUE	TRUE	TRUE	FALSE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	TRUE	FALSE

TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	FALSE	TRUE
FALSE	FALSE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	FALSE	TRUE
TRUE	TRUE	TRUE	FALSE
FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	FALSE
FALSE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	FALSE	TRUE
FALSE	TRUE	TRUE	FALSE
TRUE	FALSE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
FALSE	FALSE	TRUE	TRUE
TRUE	TRUE	TRUE	FALSE
TRUE	TRUE	FALSE	TRUE
FALSE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	TRUE	TRUE

	TRUE	FALSE
1 Baht	39	11
2 Baht	44	6
5 Baht	44	6
10 Baht	42	8
Percentage	78 %	22 %
	88 %	12 %
	88 %	12 %
	84 %	16 %

ตารางที่ 1 ตารางในการแสดงประสิทธิภาพในการยอดเหรียญแล้วตรวจพบเหรียญ

4.2 เพื่อศึกษาหาประสิทธิภาพของการส่งข้อมูลประเภท PM 2.5 ที่ถูกส่งไปยังอุปกรณ์ต่าง ๆ

```
Checksum failure: calculated=276 expected=242
PM25:10
PM25:10
Checksum failure: calculated=25A expected=242
PM25:10
PM25:10
Checksum failure: calculated=253 expected=242
PM25:10
PM25:10
Checksum failure: calculated=306 expected=342
PM25:10
PM25:10
Checksum failure: calculated=28F expected=242
Timeout waiting for start byte
PM25:10
PM25:10
Checksum failure: calculated=306 expected=342
PM25:10
PM25:9
Checksum failure: calculated=258 expected=242
Timeout waiting for start byte
PM25:10
PM25:9
Checksum failure: calculated=267 expected=242
PM25:10
PM25:10
Checksum failure: calculated=2F2 expected=242
PM25:12
PM25:12
Checksum failure: calculated=2FD expected=242
Timeout waiting for start byte
PM25:1636
```

ภาพที่ 50 ผลลัพธ์ผ่าน Serial Monitor ของอุปกรณ์ Arduino UNO R3

10	FALSE
#VALUE!	#VALUE!
10	FALSE
10	FALSE
#VALUE!	#VALUE!
#VALUE!	#VALUE!
10	FALSE
10	FALSE
#VALUE!	#VALUE!
10	FALSE
9	FALSE
#VALUE!	#VALUE!
#VALUE!	#VALUE!
10	FALSE
9	FALSE
#VALUE!	#VALUE!
10	FALSE
10	FALSE

ตารางที่ 2 ตารางในการแสดงผลลัพธ์ประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5

#VALUE!	#VALUE!
12	FALSE
12	FALSE
#VALUE!	#VALUE!
#VALUE!	#VALUE!
1636	TRUE
2043	TRUE
#VALUE!	#VALUE!
3672	TRUE
4079	TRUE
#VALUE!	#VALUE!
3324	TRUE
2909	TRUE
#VALUE!	#VALUE!
#VALUE!	#VALUE!
1960	TRUE
1571	TRUE

ตารางที่ 3 ตารางในการแสดงผลลัพธ์ประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5

Status	Number of status	Status	
Success	21	Success Total	13
Success false if > 1000	8	Success Rate	0.37
Checksum failed	10	Percentage	37.14 %
Timeout	4	Error Percentage	62.86 %

ตารางที่ 4 ตารางอัตราประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5

```
PM25: 28
Timeout waiting for start byte
Timeout waiting for start byte
PM25: 34
PM25: 33
PM25: 33
PM25: 31
PM25: 31
PM25: 29
PM25: 30
Timeout waiting for start byte
PM25: 26
PM25: 26
Timeout waiting for start byte
Timeout waiting for start byte
PM25: 26
Timeout waiting for start byte
Checksum mismatch, skipping packet
PM25: 24
PM25: 26
Timeout waiting for start byte
Timeout waiting for start byte
PM25: 29
PM25: 30
PM25: 29
PM25: 30
PM25: 29
PM25: 29
PM25: 29
```

ภาพที่ 51 ผลลัพธ์ผ่าน Serial Monitor ของอุปกรณ์ NodeMCU

#VALUE!	#VALUE!
#VALUE!	#VALUE!
50	FALSE
#VALUE!	#VALUE!
#VALUE!	#VALUE!
#VALUE!	#VALUE!
52	FALSE
#VALUE!	#VALUE!
#VALUE!	#VALUE!
35	FALSE
33	FALSE
33	FALSE
29	FALSE

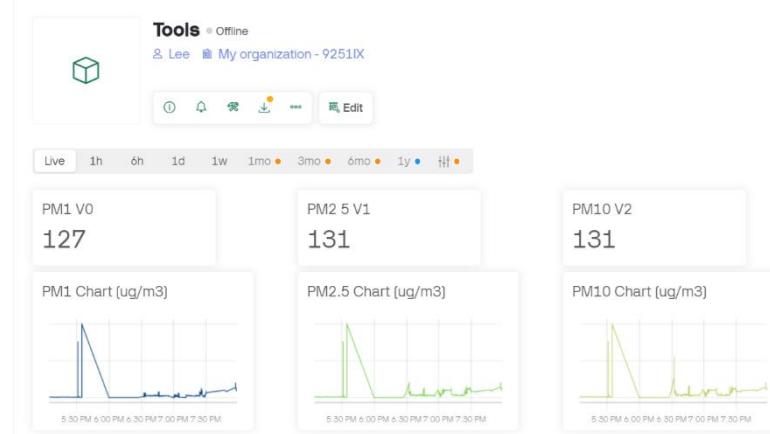
ตารางที่ 5 ตารางในการแสดงผลลัพธ์ประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5

#VALUE!	#VALUE!
28	FALSE
#VALUE!	#VALUE!
27	FALSE
27	FALSE
#VALUE!	#VALUE!
27	FALSE
#VALUE!	#VALUE!
#VALUE!	#VALUE!
26	FALSE
25	FALSE
28	FALSE

ตารางที่ 6 ตารางในการแสดงผลลัพธ์ประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5

Status	Number of status	Status	
Success	20	Success Total	20
Success false if > 1000	0	Success Rate	0.63
Checksum failed	0	Percentage	62.50 %
Timeout	12	Error Percentage	37.50 %

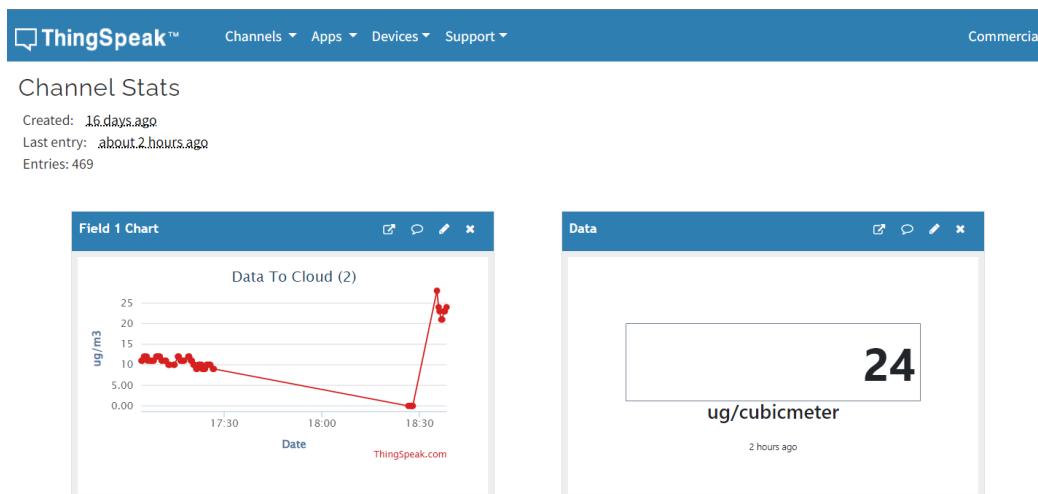
ตารางที่ 7 ตารางในการแสดงประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5



ภาพที่ 52 ผลลัพธ์ผ่าน Blynk

```
odroid@odroid:~/Documents$ python3 UploadToLine.py
Connected to /dev/ttyS1 at 9600 baud.
Received PM2.5: 28 ug/m3
Data sent to ThingSpeak: PM2.5 = 28
PM2.5 notification sent to Line: 28 ug/m3
Received PM2.5: 24 ug/m3
Data sent to ThingSpeak: PM2.5 = 24
PM2.5 notification sent to Line: 24 ug/m3
Received PM2.5: 23 ug/m3
Data sent to ThingSpeak: PM2.5 = 23
PM2.5 notification sent to Line: 23 ug/m3
Received PM2.5: 21 ug/m3
Data sent to ThingSpeak: PM2.5 = 21
PM2.5 notification sent to Line: 21 ug/m3
Received PM2.5: 21 ug/m3
Data sent to ThingSpeak: PM2.5 = 21
PM2.5 notification sent to Line: 21 ug/m3
Received PM2.5: 23 ug/m3
Data sent to ThingSpeak: PM2.5 = 23
PM2.5 notification sent to Line: 23 ug/m3
Received PM2.5: 23 ug/m3
Data sent to ThingSpeak: PM2.5 = 23
PM2.5 notification sent to Line: 23 ug/m3
Received PM2.5: 24 ug/m3
Data sent to ThingSpeak: PM2.5 = 24
PM2.5 notification sent to Line: 24 ug/m3
Received PM2.5: 24 ug/m3
Data sent to ThingSpeak: PM2.5 = 24
PM2.5 notification sent to Line: 24 ug/m3
^CTraceback (most recent call last):
  File "/home/odroid/Documents/UploadToLine.py", line 69, in <module>
    main()
  File "/home/odroid/Documents/UploadToLine.py", line 66, in main
    time.sleep(15) # Wait for a while before reading next value (adjust based on your requirement)
KeyboardInterrupt
odroid@odroid:~/Documents$ |
```

ภาพที่ 53 ผลลัพธ์ผ่านการ print การรับและส่งข้อมูลของ Odroid C4



ภาพที่ 54 ผลลัพธ์ผ่าน ThingSpeak

Status	Number of status	Status	
Success	24	Success Total	24
Success false if > 1000	0	Success Rate	1.00
Checksum failed	0	Percentage	100.00 %
Timeout	0	Error Percentage	0.00 %

ตารางที่ 8 ตารางในการแสดงประสิทธิภาพในการแสดงค่าฝุ่นชนิด PM 2.5 ของ Odroid C4

4.3 เพื่อพัฒนาให้อุปกรณ์ Internet of Things สามารถใช้ในบ้าน และชีวิตประจำวันได้



ภาพที่ 55 อุปกรณ์ Piggy Bank Saving Your Life แบบประกอบสำเร็จ



ภาพที่ 56 อุปกรณ์ Piggy Bank Saving Your Life เมื่อเทียบขนาดกับโน้ตบุ๊ค

บทที่ 5

สรุปผลการทดลอง อภิปรายผลและข้อเสนอแนะ

จากการศึกษาเรื่อง Piggy Bank Saving Your Life โดยการนำการค้นคว้าหาเอกสารเกี่ยวกับการทำงานของอุปกรณ์ Internet of Things ชนิดต่าง ๆ และการสืบค้นหาข้อมูลเพิ่มเติมเกี่ยวกับอุปกรณ์ที่นอกเหนือจากเนื้อหาการเรียน มีสรุปผลการทดลอง ดังนี้

สรุปผลการทดลอง

ตอนที่ 1 เพื่อศึกษาการเขียนโปรแกรม และศึกษาการทำงานของบอร์ด Arduino ในการแยกเหรียญ

จากการเขียนโปรแกรมให้สามารถรองรับการแยกเหรียญชนิด 1, 2, 5, 10 บาท พบว่า มีปัญหาในการแยกเหรียญบ้าง ซึ่งมีสาเหตุเกิดขึ้นจากการหยุดเหรียญ 1 ครั้งที่ใช้จำนวนเหรียญที่มากกว่า 1 เหรียญ หยุดลงไปพร้อมกันผ่านเซนเซอร์ในการรับแยกเหรียญ ดังนี้

การแยกเหรียญ 1 บาท มีประสิทธิภาพอยู่ที่ 78 เปอร์เซ็นต์

การแยกเหรียญ 2 บาท มีประสิทธิภาพอยู่ที่ 88 เปอร์เซ็นต์

การแยกเหรียญ 5 บาท มีประสิทธิภาพอยู่ที่ 88 เปอร์เซ็นต์

การแยกเหรียญ 10 บาท มีประสิทธิภาพอยู่ที่ 84 เปอร์เซ็นต์

ตอนที่ 2 เพื่อศึกษาหาประสิทธิภาพของการส่งข้อมูลประเภท PM 2.5 ที่ถูกส่งไปยังอุปกรณ์ต่าง ๆ

จากการเขียนโปรแกรมให้สามารถวัดค่าฝุ่นในพื้นที่ พบว่า มีปัญหาในการวัดฝุ่นบ้าง ซึ่งมีสาเหตุเกิดขึ้นจากการตัวเซนเซอร์ การเกิดค่าคลาดเคลื่อนทางข้อมูล ข้อมูลสูญหาย ข้อมูลไม่พร้อมส่ง และความเร็วในการส่งข้อมูล ดังนี้

การรับข้อมูลของอุปกรณ์ Arduino UNO R3 มีประสิทธิภาพอยู่ที่ 37.14 เปอร์เซ็นต์

การรับข้อมูลของอุปกรณ์ NodeMCU มีประสิทธิภาพอยู่ที่ 62.50 เปอร์เซ็นต์

การรับข้อมูลของอุปกรณ์ Odroid มีประสิทธิภาพอยู่ที่ 100 เปอร์เซ็นต์

ตอนที่ 3 เพื่อพัฒนาให้อุปกรณ์ Internet of Things สามารถใช้ในบ้าน และชีวิตประจำวันได้จากการสร้างอุปกรณ์ Internet of Things เพื่อให้สามารถใช้ในบ้าน โดยมีความสามารถในรูปแบบของการออมทรัพย์ในอดีต และมีความทันสมัยของปัจจุบัน ดังนี้ ข้อดี คือ อุปกรณ์ตั้งกล่าวสามารถตั้งบนโต๊ะ และยังสามารถนำไปตั้งในพื้นที่อื่น ๆ ได้ ข้อเสีย คือ น้ำหนักของตัวอุปกรณ์ที่ค่อนข้างมาก ความยุ่งยากในการพกพาอุปกรณ์ไปในสถานที่ต่าง ๆ และสายไฟที่มีจำนวนมาก

ประโยชน์ที่คาดว่าจะได้รับ

- 1) สามารถคาดการณ์จำนวนในพื้นที่ที่อยู่ได้
- 2) ประหยัดพื้นที่ใช้สอยจากการมี 2 อุปกรณ์ในพื้นที่กลายเป็น 1 อุปกรณ์
- 3) ลดค่าใช้จ่ายในการซื้อการะบุกออมสินประเภทใช้แล้วทิ้ง

ข้อเสนอแนะ

- 1) ทำแอบ躲在ในการแสดงข้อมูลบนโทรศัพท์มือถือ เพื่อลดความยุ่งยากในการทำงาน
- 2) ลดน้ำหนัก หรือขนาดของตัวอุปกรณ์ในการทำให้อุปกรณ์มีขนาดกะทัดรัดมากขึ้น
- 3) การลดจำนวนสายเชื่อมต่อต่าง ๆ ในการทำให้อุปกรณ์ใช้งานสะดวกมากขึ้น

เอกสารอ้างอิง

AnalogRead (ม.ป.ป.) สืบค้นข้อมูลจาก <https://www.analogread.com/product/1472>

/counting-sensor-module-เซนเซอร์ก้ามนูญ

Maxim Integrated Products (2558) สืบค้นข้อมูลจาก <https://www.analog.com/media/en/technical-documentation/data-sheets/ds1307.pdf>

Handson Technology (ม.ป.ป.) สืบค้นข้อมูลจาก https://www.handsontec.com/dataspecs/module/I2C_1602_LCD.pdf

Parallax (2554) เข้าถึงได้จาก <https://cdn.sparkfun.com/assets/f/f/a/5/0/DS-16038.pdf>

Handson Technology (ม.ป.ป.) สืบค้นข้อมูลจาก <https://handsontec.com/dataspecs/module/esp8266-V13.pdf>

Odroid Wiki (2024) สืบค้นข้อมูลจาก <https://wiki.odroid.com/odroid-c4/hardware/hardware>

Zhou Yong (2016) สืบค้นข้อมูลจาก https://github.com/m2mlorawan/datasheet/blob/master/plantower-pms5003-manual_v2-3.pdf