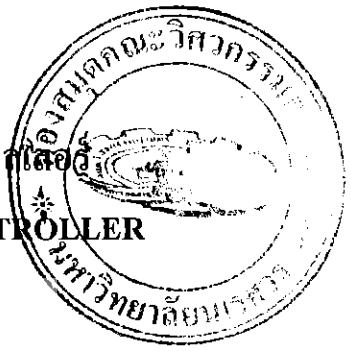
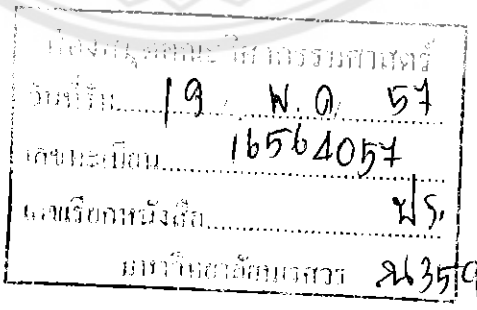


นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์

ALARM CLOCK FOR THE DEAF BY MICROCONTROLLER

นายณัฐวัตร แย้มศรี รหัส 53362709

นายภูวดล แย้มชื่น รหัส 53362983



ป3519 น
2556

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2556

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน	นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์		
ผู้ดำเนินโครงงาน	นายณัฐวัตร	แย้มศรี	รหัส 53362709
	นายภูวดล	แย้มชื่น	รหัส 53362983
ที่ปรึกษาโครงงาน	ดร.มุฑิตา สงฆ์จันทร์		
สาขาวิชา	วิศวกรรมไฟฟ้า		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2556		

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

.....ที่ปรึกษาโครงงาน
(ดร. มุฑิตา สงฆ์จันทร์)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. ศุภวรรณ พลพิทักษ์ชัย)

.....กรรมการ
(อาจารย์เศรษฐา ตั้งค้าวานิช)

ชื่อหัวข้อโครงงาน	นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์		
ผู้ดำเนินโครงงาน	นายณัฐวัตร	แย้มศรี	รหัส 53362709
	นายภูวดล	แย้มชื่น	รหัส 53362983
ที่ปรึกษาโครงงาน	ดร.มุฑิตา สงฆ์จันทร์		
สาขาวิชา	วิศวกรรมไฟฟ้า		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2556		

.....

บทคัดย่อ

โครงงานนี้เป็นการออกแบบและสร้างนาฬิกาปลุกคนหูหนวก โดยใช้ไมโครคอนโทรลเลอร์เป็นตัวควบคุมการทำงาน สามารถทำงานบนพื้นฐานของเวลาจริง เลือกการทำงานระบบเวลาได้ทั้ง 12 ชั่วโมง และ 24 ชั่วโมง มีการทำงานในส่วนของนาฬิกาปลุกซึ่งสามารถตั้งปลุกได้ 2 ครั้ง ในส่วนของนาฬิกาปลุกจะมีระบบการสั่งปลุก เป็นแบบการสั่นสะเทือน สามารถเลือกระดับการสั่นสะเทือนได้ 4 ระดับ การกำหนดแรงสั่นสะเทือนจะมีการคำนึงถึงความเหมาะสมในการใช้งาน ผลจากการทดลองแสดงให้เห็นว่าระดับความแรงสูงสุดไม่ก่อให้เกิดอันตราย ทั้งนี้เพื่อความปลอดภัยของผู้ใช้งาน

Project title	Alarm Clock for the Deaf by Microcontroller
Name	Mr. Natthawat Yamsri ID. 53362709
	Mr. Puwadol Yamchuen ID. 53362983
Project advisor	Ms. Mutita Songjun, Ph.D.
Major	Electrical Engineering
Department	Electrical and Computer Engineering
Academic year	2013

Abstract

This Project is to design and construct the alarm clock for the deaf. This alarm clock controlled by microcontroller is able to operate based on real time. It can be set the time to be 12 hour or 24 hours. Moreover, It can be set the time twice for making the alarm. The alarm system uses the vibration which can be adjusted into 4 levels. The experiments show that the maximum level of vibration is not harm the user and the alarm clock is able to wake the deaf up.

กิตติกรรมประกาศ

โครงงานวิศวกรรมเรื่องนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์ สำเร็จลุล่วงไปด้วยดีเนื่องจากความอนุเคราะห์จากอาจารย์ที่ปรึกษาโครงงานคือ ดร.มุฑิตา สงฆ์จันทร์ ที่ให้คำปรึกษา คำแนะนำ และช่วยเหลือตรวจสอบแก้ไขข้อบกพร่องต่างๆ ตลอดจนให้ความรู้และข้อคิดเห็นที่เป็นประโยชน์ต่อโครงงานนี้ด้วยความเอาใจใส่เป็นอย่างดีจนกระทั่งโครงงานเสร็จสมบูรณ์

ขอขอบคุณ ดร.ศุภวรรณ พลพิทักษ์ชัย และอาจารย์เศรษฐา ตั้งค้าวานิช ซึ่งเป็นคณะกรรมการโครงงาน ที่ให้คำแนะนำตลอดระยะเวลาในการทำโครงงานและท่านที่มีส่วนร่วมในการทำโครงงานนี้ จนทำให้โครงงานนี้สำเร็จลุล่วงไปได้ด้วยดีมา ณ โอกาสนี้

นอกจากนี้ยังต้องขอขอบคุณภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ที่ให้ยืมอุปกรณ์และเครื่องมือวัดมาใช้งาน จนทำให้โครงงานนี้สำเร็จลุล่วงไปได้

นายณัฐวัตร แย้มศรี
นายภูวดล แย้มชื่น

สารบัญ

หน้า

ใบรับรองปริญญานิพนธ์ ..... ก
บทคัดย่อภาษาไทย ..... ข
บทคัดย่อภาษาอังกฤษ ..... ค
กิตติกรรมประกาศ ..... ง
สารบัญ ..... จ
สารบัญตาราง ..... ซ
สารบัญรูป ..... ฌ

บทที่ 1 บทนำ 1
1.1 ที่มาและความสำคัญของโครงงาน 1
1.2 วัตถุประสงค์ของโครงงาน 1
1.3 ขอบเขตของโครงงาน 1
1.4 ขั้นตอนและแผนการดำเนินงาน 2
1.5 แผนการดำเนินงาน 2
1.6 ประโยชน์ที่คาดว่าจะได้รับ 3
1.7 งบประมาณที่ใช้ 4

บทที่ 2 ทฤษฎีและหลักการที่เกี่ยวข้อง 5
2.1 ลักษณะผู้มีความบกพร่องทางการได้ยิน 5
2.2 ไมโครคอนโทรลเลอร์ 6
2.2.1 ไมโครคอนโทรลเลอร์ตระกูล AVR 6
2.2.2 คุณสมบัติไมโครคอนโทรลเลอร์ตระกูล AVR 6
2.2.3 โครงสร้างภายในไมโครคอนโทรลเลอร์ตระกูล AVR 7
2.2.4 ตำแหน่งขาของไมโครคอนโทรลเลอร์ AVR เบอร์ AT mega328 8
2.2.5 การเขียนโปรแกรมไมโครคอนโทรลเลอร์ 10
2.3 ระบบสั่นสะเทือนในนาฬิกาปลุก 10
2.3.1 มอเตอร์ไฟฟ้ากระแสตรง (DC Motor) 11
2.3.2 การควบคุมความเร็วของมอเตอร์กระแสตรง 12
2.3.3 ระดับความแรงของมอเตอร์สั่นสะเทือน 13

สารบัญ (ต่อ)

หน้า

2.4 ไอซีเวลาจริง (Real Time Clock) DS1307 13
2.5 หน้าจอผลึกเหลว (Liquid Crystal Display) 15
2.6 เซนเซอร์ตรวจจับการเคลื่อนไหว (Liquid Crystal Display) 17

บทที่ 3 การออกแบบและสร้างนาฬิกาปลุกคนหูหนวก 19
3.1 ขั้นตอนและหลักการทำงานของนาฬิกาปลุกคนหูหนวก 19
3.2 การออกแบบและการสร้างนาฬิกาปลุกคนหูหนวก 22
3.2.1 โครงสร้างในส่วนตัวเครื่องนาฬิกาปลุกคนหูหนวก 22
3.2.2 โครงสร้างในส่วนชุดมอเตอร์สั่นสะเทือน 22
3.3 วงจรที่ใช้ในการทำงานของนาฬิกาปลุกคนหูหนวก 24
3.3.1 วงจรไมโครคอนโทรลเลอร์ 24
3.3.2 วงจรแปลงไฟและจ่ายไฟ 24
3.3.3 วงจรขับมอเตอร์ 25

บทที่ 4 ผลการทดสอบและผลการวิเคราะห์ 26
4.1 ทดสอบการทำงานของระบบเวลาจริง 26
4.2 ทดสอบระดับการสั่นสะเทือนของอุปกรณ์สั่นปลุก 27
4.3 ทดสอบการทำงานของเซ็นเซอร์ตรวจจับการเคลื่อนไหว 29
4.3.1 ทดสอบระยะการตรวจจับสูงสุดของเซ็นเซอร์ 29
4.3.2 ทดสอบระยะเวลาก่อนการทำงานโหมดพักหน้าจอ 29

บทที่ 5 สรุปผลและข้อเสนอแนะ 30
5.1 สรุปผลการดำเนินโครงการ 30
5.2 ปัญหาและการแก้ไข 31
5.3 ข้อเสนอแนะและแนวทางการพัฒนา 31

เอกสารอ้างอิง 32

สารบัญ (ต่อ)

หน้า

ภาคผนวก ก โปรแกรมการทำงานของนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์ ... 33
ภาคผนวก ข คู่มือการใช้งานนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์ 77

ประวัติผู้ดำเนินโครงงาน 85

สารบัญตาราง

ตารางที่ หน้า

1.1 แผนการปฏิบัติงาน 3

4.1 ผลการทดสอบการทำงานระบบเวลาจริงก่อนเที่ยงวัน 27

4.2 ผลการทดสอบการทำงานระบบเวลาจริงหลังเที่ยงวัน 27

4.3 ผลการทดสอบค่าแรงดันไฟฟ้าที่มอเตอร์สั่นสะเทือนได้รับในแต่ละระดับ 28

4.4 ผลการทดสอบระยะการทำงานสูงสุดที่เซ็นเซอร์ตรวจจับได้ 29

4.5 ผลการทดสอบระยะเวลาก่อนการทำงานโหมดพักหน้าจอ 29

มหาวิทยาลัยนเรศวร

สารบัญรูป

รูปที่ หน้า

2.1 โครงสร้างพื้นฐานของไมโครคอนโทรลเลอร์ 8

2.2 ตำแหน่งขาของไมโครคอนโทรลเลอร์ AVR เบอร์ ATmega328 10

2.3 ส่วนประกอบมอเตอร์ไฟฟ้ากระแสตรง 11

2.4 ตำแหน่งขาไอซี RTC DS1307 14

2.5 การเชื่อต่อDS1307 เข้ากับไมโครคอนโทรลเลอร์ 15

2.6 หน้าจอแสดงผลผลึกเหลว 16

2.7 โครงสร้างการทำงานของหน้าจอแสดงผลผลึกเหลว 16

2.8 เซนเซอร์ตรวจจับการเคลื่อนไหว 17

2.9 ลักษณะการทำงานของเซนเซอร์ตรวจจับการเคลื่อนไหว 18

2.10 จุดเชื่อมต่อสำหรับใช้งานของเซนเซอร์ตรวจจับการเคลื่อนไหว 18

3.1 ขั้นตอนการทำงานของนาฬิกาปลุกคนหูหนวก 20

3.2 โหมดการทำงานนาฬิกาปลุก 21

3.3 โหมดการทำงานนาฬิกาปกติ 22

3.4 โครงสร้างนาฬิกาปลุกคนหูหนวก 23

3.5 โครงสร้างชุดมอเตอร์สั่นสะเทือน 23

3.6 วงจรไมโครคอนโทรเลอร์ 24

3.7 วงจรแปลงไฟและจ่ายไฟ 25

3.8 วงจรขับเคลื่อนมอเตอร์ 25

4.1 ตัวอย่างการทดสอบการทำงานของระบบเวลาจริง 26

4.2 วิธีทำการทดสอบแรงดันที่จ่ายให้กับมอเตอร์สั่นสะเทือน 28

4.3 วิธีการทดสอบการทำงานของเซ็นเซอร์ตรวจจับการเคลื่อนไหว 29

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

ในหลายปีที่ผ่านมา ปัญหาการเพิ่มขึ้นของจำนวนผู้พิการต่อจำนวนประชากรนั้นสูงขึ้นอย่างมาก ในจำนวนผู้พิการเหล่านั้นกว่าร้อยละ 1.85 นั้นสามารถทำงานได้เหมือนคนปกติ ทางรัฐบาลได้เล็งเห็นปัญหาในส่วนนี้และได้ให้ความช่วยเหลือผู้พิการที่สามารถทำงานได้โดยการออกพระราชบัญญัติฟื้นฟูสมรรถภาพคนพิการ เนื้อหาหลักคือการให้สถานประกอบการของเอกชนที่มีลูกจ้างตั้งแต่ 200 คนขึ้นไปรับคนพิการที่สามารถทำงานได้ในอัตราลูกจ้างทั้งหมดทุก 200 คนต่อคนพิการ 1 คน และยังมีมาตรการช่วยเหลืออื่นๆอีกมากมาย ทั้งนี้ทั้งนั้นการแก้ปัญหานี้แก้ปัญหาได้เพียงส่วนหนึ่งเท่านั้น ปัญหาที่แท้จริงคือทำอย่างไรที่จะจัดหาหรือพัฒนาเครื่องมือและอุปกรณ์ที่จะสนับสนุนและส่งเสริมคนพิการให้สามารถดำรงชีวิตในสังคมร่วมกับคนปกติได้โดยไม่เป็นภาระกับผู้อื่น

ดังนั้นจึงทำให้เกิดแนวคิดในการจัดทำนาฬิกาปลุกสำหรับผู้พิการทางการได้ยินด้วยระบบไมโครคอนโทรลเลอร์เพื่อเป็นอุปกรณ์อำนวยความสะดวกสำหรับผู้พิการทางการได้ยิน

1.2 วัตถุประสงค์ของโครงงาน

ออกแบบและสร้างนาฬิกาปลุกสำหรับผู้พิการทางการได้ยินด้วยระบบไมโครคอนโทรลเลอร์ ที่สามารถใช้งานได้ในสถานการณ์จริง

1.3 ขอบเขตของโครงงาน

1. นาฬิกาปลุกสามารถตั้งเวลาโดยระบบเวลาจริง
2. นาฬิกาปลุกสามารถปรับระดับแรงสั่นสะเทือนได้ 3 ระดับ
3. นาฬิกาปลุกสามารถตั้งเวลาและระดับแรงสั่นสะเทือนโดยควบคุมผ่านทางไมโครคอนโทรลเลอร์

1.4 ขั้นตอนและแผนการดำเนินงาน

1. ศึกษาทฤษฎีพื้นฐานไมโครคอนโทรลเลอร์และการประยุกต์ใช้งาน

2. ศึกษาทฤษฎีของอุปกรณ์ต่างๆที่ใช้ในการสร้างนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์

3. ออกแบบและสร้างนาฬิกาปลุกหูหนวกด้วยระบบไมโครคอนโทรลเลอร์

4. เขียนโปรแกรมเพื่อควบคุมการทำงานนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์

5. ทดสอบการทำงานและปรับปรุงแก้ไขในส่วนคำสั่งที่ผิดพลาด

6. สรุปผลการทำงานและจัดทำรูปเล่มปริญญานิพนธ์

1.5 แผนการดำเนินงาน

โครงงานการสร้างนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์มีแผนปฏิบัติงานดังตารางที่ 1.1

ตารางที่ 1.1 แผนการปฏิบัติงาน

การปฏิบัติงาน	ปี 2556								
	เม.ย	พ.ค	มิ.ย	ก.ค	ส.ค	ก.ย	ต.ค	พ.ย	ธ.ค
1. ศึกษาทฤษฎีพื้นฐานไมโครคอนโทรลเลอร์	←	—	→						
2. ศึกษาทฤษฎีของอุปกรณ์ต่างๆ		←	—	→					
3. ออกแบบและสร้างนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์		←	—	—	→				
4. เขียนโปรแกรมควบคุมการทำงาน					←	—	—	→	
5. ทดสอบการทำงานและปรับปรุงแก้ไข								←	→
6. สรุปผลการทำงานและจัดทำรูปเล่มโครงงาน								←	→

1.6 ประโยชน์ที่คาดว่าจะได้รับจากโครงงาน

1. ได้นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์จำนวน 1 เครื่อง

2. ได้พัฒนาความรู้เกี่ยวกับไมโครคอนโทรลเลอร์และโปรแกรมควบคุม

3. สามารถนำความรู้ที่ได้ไปประยุกต์ใช้งานจริงหรือประยุกต์สร้างอุปกรณ์อื่นๆ ที่มีลักษณะการทำงานใกล้เคียงกัน

4. สามารถสร้างองค์ความรู้พื้นฐานเพื่อที่จะพัฒนาไปสู่ความรู้ทางวิชาการและวิศวกรรมที่สูงขึ้น

1.7 งบประมาณที่ใช้

1. ค่าอุปกรณ์อิเล็กทรอนิกส์ที่ใช้ในการทำวงจรการทำงาน	500	บาท
2. ค่าโครงสร้างนาฬิกาปลุกหุ่นวกด้วยระบบไมโครคอนโทรลเลอร์	500	บาท
3. ค่าจัดทำรูปเล่มรายงาน	1,000	บาท
รวมเป็นเงินทั้งสิ้น (สองพันบาทถ้วน)	2,000	บาท

หมายเหตุ: ถัวเฉลี่ยในทุกรายการ

บทที่ 2

ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 ลักษณะของผู้มีความบกพร่องทางการได้ยิน

บุคคลที่มีความบกพร่องทางการได้ยิน ได้แก่ บุคคลที่สูญเสียการได้ยินตั้งแต่ระดับหูตึงเล็กน้อยจนถึงหูหนวก ซึ่งเป็น 2 ประเภทดังนี้

1. คนหูหนวก หมายถึง บุคคลที่สูญเสียการได้ยินมากจนไม่สามารถเข้าใจภาษาพูดผ่านทางการได้ยินไม่ว่าจะใส่หรือไม่ใส่เครื่องช่วยฟัง ซึ่งโดยทั่วไปหากตรวจการได้ยินจะมีการสูญเสียการได้ยิน 90 เดซิเบลขึ้นไป

2. คนหูตึง หมายถึง บุคคลที่มีการได้ยินเหลืออยู่เพียงพอที่จะได้ยินเสียงพูดผ่านทางการได้ยินโดยทั่วไปจะใส่เครื่องช่วยฟัง ซึ่งหากตรวจการได้ยินจะมีการได้ยินน้อยกว่า 90 เดซิเบลลงมาถึง 26 เดซิเบล

คนหูตึงอาจแบ่งตามระดับ การได้ยินได้ 4 กลุ่ม คือ

1. หูตึงระดับที่ 1 มีการได้ยินเฉลี่ยระหว่าง 26 - 40 เดซิเบล

2. หูตึงระดับที่ 2 มีการได้ยินเฉลี่ยระหว่าง 41 - 55 เดซิเบล

3. หูตึงระดับที่ 3 มีการได้ยินเฉลี่ยระหว่าง 56 - 70 เดซิเบล

4. หูตึงระดับที่ 4 มีการได้ยินเฉลี่ยระหว่าง 71 - 90 เดซิเบล

หูตึงระดับที่ 1 (26-40 เดซิเบล) ตึงเล็กน้อย จะมีปัญหาในการรับฟังเสียงเบาๆเช่น เสียงกระซิบหรือเสียงจากที่ไกลๆ เด็กกลุ่มนี้สามารถเรียนร่วมกับเด็กปกติในห้องเรียนธรรมดาได้ หากมีที่นั่งเรียนที่สามารถมองเห็นครูและเพื่อนได้ดี หากมีเครื่องช่วยฟังที่เหมาะสมก็จะเป็นประโยชน์มาก

หูตึงระดับที่ 2 (41-55 เดซิเบล) ตึงปานกลาง จะมีปัญหาในการฟังเสียงพูดคุยที่ดังในระดับปกติที่มีระยะห่าง 3-5 ฟุต และไม่เห็นหน้าผู้พูด ดังนั้นเมื่อพูดคุยด้วยเสียงธรรมดาก็จะไม่ได้ยินหรือได้ยินไม่ชัดจับใจความไม่ได้ นอกจากนี้มีปัญหาในการพูดเล็กน้อย เช่นพูดไม่ชัด ออกเสียงเพี้ยนพูดเสียงเบาหรือเสียงผิดปกติ

หูตึงระดับที่ 3 (56-70 เดซิเบล) ตึงมาก มีปัญหาในการรับฟังและเข้าใจคำพูดเมื่อพูดคุยกันด้วยเสียงดังเต็มที่ ก็ยังไม่ได้ยินมีปัญหาในการรับฟังเสียงหลายเสียงพร้อมกัน เช่น เสียงในห้องประชุม มีพัฒนาการทางภาษาและการพูดช้ากว่าเด็กปกติพูดไม่ชัด เสียงเพี้ยนบางคนไม่พูด

หูตึงระดับที่ 4 (71 – 90 เดซิเบล) ตึงรุนแรง เป็นกลุ่มเด็กหูตึงระดับรุนแรง จึงมีปัญหาใน การรับฟังเสียงและการเข้าใจคำพูดอย่างมาก เด็กจะสามารถได้ยินเฉพาะเสียงที่ดังใกล้หูในระยะทาง 1 ฟุต ต้องตะโกนหรือใช้เครื่องขยายเสียง จึงจะได้ยินเด็กกลุ่มนี้แม้จะใช้เครื่องช่วยฟังก์ มีปัญหาในการแยกเสียง อาจแยกเสียงสระได้แต่แยกเสียงพยัญชนะได้ยาก มักพูดไม่ชัด และมีความผิดปกติ บางคนไม่พูด

2.2 ไมโครคอนโทรเลอร์

นาฬิกาปลุกคนหูหนวกจะถูกควบคุมโดยไมโครคอนโทรลเลอร์ ซึ่งไมโครคอนโทรลเลอร์คือ ชิพประมวลผลอย่างหนึ่งทำหน้าที่ประมวลผลตามโปรแกรมหรือชุดคำสั่งโครงสร้างภายในจะเป็นวงจรรวมขนาดใหญ่ที่รวมกันอยู่ในชิพหรือไอซีไมโครคอนโทรลเลอร์ถูกออกแบบเพื่อใช้งานควบคุมสามารถติดต่อกับอุปกรณ์อินพุตและเอาท์พุตได้สะดวกใช้งานง่ายสามารถทำงานได้ในชิพเดียว ไมโครคอนโทรลเลอร์ที่ใช้ในโครงงานนี้เป็นตระกูล AVR

2.2.1 ไมโครคอนโทรลเลอร์ตระกูล AVR

ไมโครคอนโทรลเลอร์ AVR เป็นไอซีไมโครคอนโทรลเลอร์ของบริษัท ATMEL มีสถาปัตยกรรมภายในเป็นแบบ RISC (Reduced Instruction Set Computer) โดยใช้สัญญาณนาฬิกาเพียง 1 ลูกในการปฏิบัติงานใน 1 คำสั่ง โดยจะประกอบด้วยหน่วยความจำโปรแกรมภายในที่เป็นแบบแฟลต โปรแกรมข้อมูลได้แบบ In-System Programmable และในบางเบอร์มีการกำหนดตำแหน่งของหน่วยความจำที่สร้างเป็นบูตโหลดเดอร์ มีขนาดของหน่วยความจำตามเบอร์ของไอซีแต่ละตัว

2.2.2 คุณสมบัติของไมโครคอนโทรลเลอร์ AVR

สำหรับไมโครคอนโทรเลอร์ AVR ที่ผู้ดำเนินโครงงานเลือกใช้นั้นเป็นรุ่น ATMEGA 328 ที่เลือกใช้รุ่นนี้เนื่องจากเป็นรุ่นที่มีหน่วยความจำรองรับมากกว่ารุ่นทั่วไป มีชุดคำสั่งที่สามารถจัดการกับหน่วยความจำที่มีขนาดใหญ่มากขึ้น ทั้งยังมีส่วนอุปกรณ์เสริมพิเศษหลายอย่างในตัว จึงทำให้

สามารถใช้งานได้สะดวก รวมถึงราคาของไมโครคอนโทรลเลอร์ตัวนี้ที่ค่อนข้างถูก เมื่อเทียบกับความสามารถและประสิทธิภาพ ไมโครคอนโทรลเลอร์ AVR ATMEGA 328 มีคุณสมบัติ ดังนี้

1. เป็นไมโครคอนโทรลเลอร์ขนาด 8 บิต

2. มีหน่วยความจำภายในแบบแฟลตขนาด 8 กิโลไบต์ ที่โปรแกรมสามารถเขียนและลบได้เป็นหมื่นครั้งครั้ง

3. มีโครงสร้างภายในแบบ RISC (Reduced Instruction Set Computer)

4. มีหน่วยความจำภายในแบบ EEPROM ขนาด 512 ไบต์

5. มีหน่วยความจำภายในแบบ SRAM ขนาด 1 กิโลไบต์

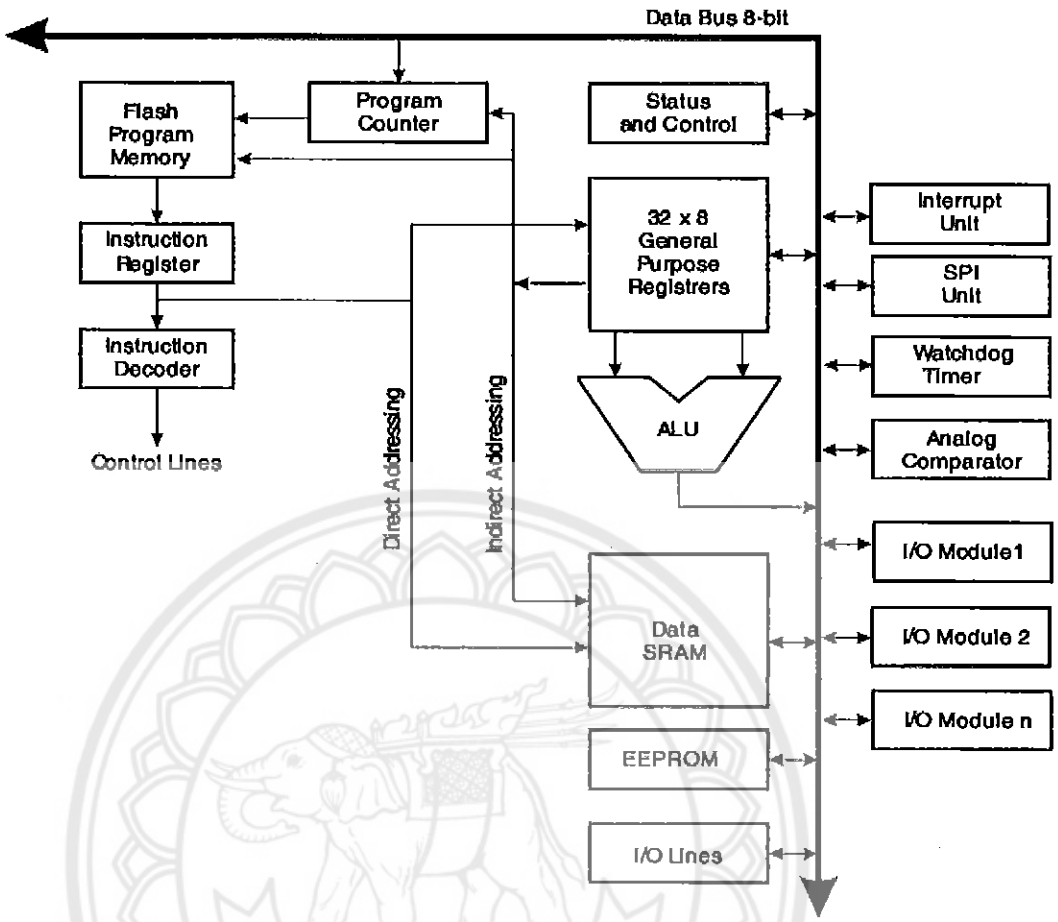
6. มีความเร็วสัญญาณนาฬิกา 16 เมกกะเฮิรตซ์

7. มีคำสั่งควบคุมการทำงานไมโครคอนโทรลเลอร์ 130 คำสั่ง คำสั่งส่วนมากจะทำสำเร็จในรอบสัญญาณนาฬิกาเดี่ยว

8. มีความเร็วประมวลผล 1 คำสั่ง ต่อ 1 สัญญาณนาฬิกา

2.2.3 โครงสร้างภายในของไมโครคอนโทรลเลอร์ AVR

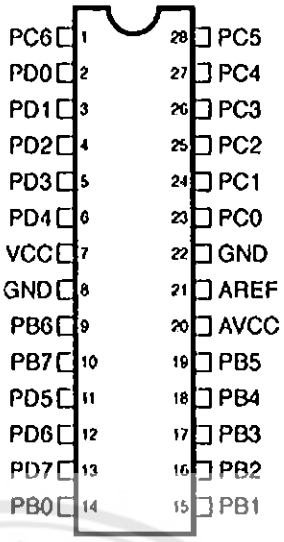
วงจรภายในของไมโครคอนโทรลเลอร์ตระกูล AVR ประกอบด้วยวงจรพอร์ตอินพุตหรือเอาท์พุตทั้งหมด 4 พอร์ต แต่ละพอร์ตจะเป็นแบบ 8 บิต หน่วยความจำโปรแกรมภายใน (EPROM, EEPROM และแฟลต) หน่วยความจำที่เป็นข้อมูล (RAM) ซึ่งรวมอยู่ในวงจรหลักของไมโครคอนโทรลเลอร์ตลอดจนวงจรการคำนวณทางคณิตศาสตร์และลอจิก (ALU) วงจรรีจิสเตอร์การควบคุมไมโครคอนโทรลเลอร์นั้นกระทำผ่านกระบวนการควบคุมโดยโปรแกรมที่เขียนขึ้น เพื่อบอกถึงการทำงานของไมโครคอนโทรลเลอร์ โดยลักษณะที่ถูกระบุขึ้นโดยผู้เขียนโปรแกรมควบคุม ซึ่งควบคุมการทำงานทั้งหมดของไมโครคอนโทรลเลอร์ ในการกำหนดพอร์ตให้เป็นอินพุตและเอาท์พุต และยังสามารถกำหนดหน่วยความจำภายในซึ่งเป็นที่เก็บข้อมูล และเป็นที่พักข้อมูลตามความต้องการ โดยในการทำงานของไมโครคอนโทรเลอร์แต่ละคำสั่ง จะอ้างอิงเวลาจากสัญญาณนาฬิกาที่ส่งให้กับไมโครคอนโทรลเลอร์ โดยมีโครงสร้างพื้นฐานการทำงานรูปที่ 2.1



รูปที่ 2.1 โครงสร้างพื้นฐานของไมโครคอนโทรลเลอร์

2.2.4 ตำแหน่งขาของไมโครคอนโทรลเลอร์ AVR เบอร์ AT mega328 และหน้าที่การทำงาน

ไมโครคอนโทรลเลอร์ตระกูล AVR ทุกเบอร์จะมีโครงสร้างและการใช้งานพื้นฐานต่างกัน ในที่นี้เราเลือกใช้งานไมโครคอนโทรลเลอร์ AVR เบอร์ AT mega328 โดยโครงสร้างจะเป็นดังรูปที่ 2.2



รูปที่ 2.2 ตำแหน่งขาของไมโครคอนโทรลเลอร์ AVR เบอร์ ATmega328

- VC ขาแรงดันไฟตรง

- GND ขากราวด์

- Port A (PA0..PA7) ขาพอร์ตเป็นอินพุตเอาท์พุตดิจิตอล กำหนดการพูลอัพภายในขาพอร์ตได้ (Internal Pull-Up Register) และสามารถกำหนดใช้งานเป็นพอร์ตอินพุตสัญญาณอานาลอก (A/D Converter) ได้

- Port B (PB0..PB7) เป็นขาพอร์ตอินพุตเอาท์พุตดิจิตอล กำหนดการพูลอัพภายในขาพอร์ตได้ (Internal Pull-Up Register) และเป็นขาพอร์ตหน้าที่พิเศษอีกด้วย เช่น ขาสำหรับการโปรแกรมชิพ ขาป้อนสัญญาณนาฬิกาภายนอก เป็นต้น

- Port C (PC0..PC7) นอกจากเป็นพอร์ตขาอินพุตเอาท์พุตดิจิตอล ที่กำหนดการพูลอัพภายในขาพอร์ตได้ (Internal Pull-Up Register) แล้วเป็นขาพอร์ตหน้าที่พิเศษ เช่น ขาเชื่อมต่อกับดีบักและโปรแกรมด้วยการเชื่อมต่อแบบ JTAG เป็นต้น

- Port D (PD0..PD7) เป็นพอร์ตขาอินพุตเอาท์พุตดิจิตอล ที่กำหนดการพูลอัพภายในขาพอร์ตได้ (Internal Pull-Up Register) และเป็นขาพอร์ตหน้าที่พิเศษ เช่น ขาเชื่อมต่อพอร์ต อนุกรม ขาอินเตอร์รับเนื่องจากสัญญาณภายนอก เป็นต้น

- RESET ขารีเซตวงจร

- XTAL1 ขาต่อคริสตอลออสซิลเลเตอร์ ช่องที่ 1 ด้านอินพุต

- XTAL2 ขาต่อคริสตอลออสซิลเลเตอร์ ช่องที่ 2 ด้านเอาท์พุต

- AVCC ขาแรงดันสำหรับพอร์ต A และโมดูลแปลงสัญญาณอานาลอกเป็นดิจิตอล

- AREF ขาแรงดันอานาลอกอ้างอิงสำหรับ โมดูลแปลงสัญญาณอานาลอกเป็นดิจิตอล

2.2.5 การเขียนโปรแกรมไมโครคอนโทรลเลอร์

ภาษาที่ใช้สำหรับการเขียนโปรแกรมบนไมโครคอนโทรลเลอร์แบ่งได้เช่นเดียวกับการเขียนโปรแกรมบนคอมพิวเตอร์คือ ภาษาเครื่อง ภาษาแอสเซมบลี และภาษาซี

ภาษาเครื่อง เป็นภาษาระดับต่ำสุด ประกอบไปด้วยรหัสฐานสอง คือ 0 กับ 1 เท่านั้น ซึ่งเป็นภาษาที่ไมโครคอนโทรลเลอร์เข้าใจ แต่มนุษย์จะทำความเข้าใจได้ยาก เพราะต้องอาศัยการจดจำรหัสคำสั่งต่างๆ รวมถึงต้องเข้าใจโครงสร้างภายในของไมโครคอนโทรลเลอร์ด้วย จึงได้มีการคิดค้นสิ่งที่เรียกว่า คอมไพล์เลอร์ (Compiler) ขึ้นมาเพื่อทำให้มนุษย์สามารถเขียนโปรแกรมด้วยภาษาระดับสูงที่มนุษย์เข้าใจได้ โดยคอมไพล์เลอร์ทำหน้าที่เปลี่ยนภาษาระดับสูงเหล่านั้นกลายเป็นภาษาเครื่องเอง

ภาษาแอสเซมบลี เป็นภาษาที่ใช้รหัสคำสั่งที่เป็นตัวอักษรภาษาอังกฤษมาแทนคำสั่งเลขฐานสอง ในแบบของภาษาเครื่องทำให้ภาษาแอสเซมบลีกลายเป็นภาษาที่มนุษย์ทำความเข้าได้ง่ายขึ้น นอกจากนั้นแล้ว ยังเป็นภาษาที่ทำให้โปรแกรมทำงานได้อย่างรวดเร็ว เพราะมีการสั่งงานไปที่ฮาร์ดแวร์โดยตรง ด้วยเหตุนี้ทำให้ผู้พัฒนาโปรแกรมจำเป็นจะต้องเข้าใจโครงสร้างภายในไมโครคอนโทรลเลอร์อย่างละเอียดด้วย ซึ่งกลายเป็นข้อด้อยของภาษาแอสเซมบลีไป คอมไพล์เลอร์ที่ทำหน้าที่แปลงภาษาแอสเซมบลีให้เป็นภาษาเครื่องเรียกว่า แอสเซมเบอร์

ภาษาซี เป็นภาษาระดับสูงที่มีความใกล้เคียงกับภาษามนุษย์ ทำให้ทำความเข้าใจได้ง่าย นอกจากนั้นแล้วการเขียนโปรแกรมภาษาซี ก็ไม่ต้องจำเป็นต้องเข้าใจโครงสร้างภายในของไมโครคอนโทรลเลอร์อย่างละเอียด เพียงแต่เข้าใจการเขียนโปรแกรมแบบโครงสร้างก็เพียงพอแล้ว ภาษาซีสามารถใช้ในการเข้าถึงโครงสร้างภายในของไมโครคอนโทรลเลอร์ได้โดยตรง ทำให้ได้โปรแกรมที่เขียนขึ้นทำงานได้รวดเร็ว ดังนั้นภาษาซี จึงเป็นที่นิยมแพร่หลายในการเขียนโปรแกรมไมโครคอนโทรลเลอร์คอมไพล์เลอร์ที่ใช้ในการแปลงภาษาซีเป็นภาษาเครื่องมีอยู่มากมาย

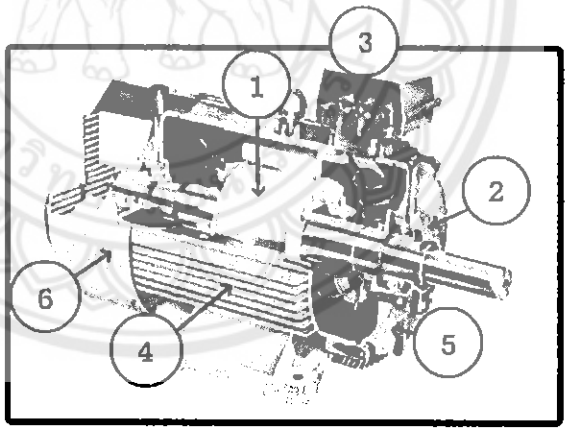
2.3 ระบบสั่นสะเทือนในนาฬิกาปลุก

เป็นระบบที่ได้รับคำสั่งว่าจะให้ทำงานหรือไม่ทำงานจากไมโครคอนโทรลเลอร์ โดยระบบนี้จะทำงานด้วยมอเตอร์เป็นตัวสั่นสะเทือนเพื่อที่จะให้ผู้ใช้งานรับรู้ว่านาฬิกาได้ทำงานสั่งปลุกแล้ว

2.3.1 มอเตอร์กระแสตรง (DC Motor)

มอเตอร์ไฟฟ้ากระแสตรงเป็นอุปกรณ์ที่ใช้เปลี่ยนพลังงานไฟฟ้าเป็นพลังงานกล ประกอบด้วย แม่เหล็กถาวร 2 ขั้ววางอยู่ระหว่างขดลวดตัวนำ ขดลวดตัวนำจะได้รับแรงดันไฟตรงป้อนให้ในการทำงาน ทำให้เกิดอำนาจแม่เหล็ก 2 ชุด มีขั้วแม่เหล็กเหมือนกันวางใกล้กัน เกิดแรงผลักดันทำให้ขดลวดตัวนำหมุนเคลื่อนที่ได้ ความเร็วของมอเตอร์ไฟฟ้ากระแสตรงนั้น สามารถปรับเปลี่ยนได้ง่าย ดังนั้นจึงสามารถนำมาใช้ประโยชน์ในการทำงานได้หลายอย่างโดยเฉพาะในงานที่ต้องควบคุมความเร็ว (Speed Control) หรืองานที่ต้องการความเที่ยงตรงของตำแหน่งงาน (Positioning)

มอเตอร์ไฟฟ้ากระแสตรงที่ผลิตมาใช้งาน มีโครงสร้างและส่วนประกอบคล้ายกับเครื่องกำเนิดไฟฟ้ากระแสตรง มีส่วนประกอบที่สำคัญเหมือนกัน มีรูปร่างลักษณะภายนอกคล้ายกัน แตกต่างกันตรงการนำไปใช้งาน โดยเครื่องกำเนิดไฟฟ้ากระแสตรงจะทำให้เกิดไฟฟ้าในรูปของแรงดันไฟตรงออกมา ส่วนมอเตอร์ไฟฟ้ากระแสตรงเมื่อจ่ายแรงดันไฟตรงให้มอเตอร์ ทำให้มอเตอร์หมุนเกิดพลังงานกลขึ้นมา ส่วนประกอบของมอเตอร์ไฟฟ้ากระแสตรงดังรูปที่ 2.3



รูปที่ 2.3 ส่วนประกอบของมอเตอร์ไฟฟ้ากระแสตรง

ส่วนประกอบหลักๆ ของมอเตอร์ไฟฟ้ากระแสตรง ประกอบด้วยส่วนต่างๆ ดังนี้

1. ขดลวดสนามแม่เหล็ก (Field Coil) คือขดลวดที่ถูกพันอยู่กับขั้วแม่เหล็กที่ยึดติดกับโครงมอเตอร์ ทำหน้าที่กำเนิดขั้วแม่เหล็กขั้วเหนือ (N) และขั้วใต้ (S) แทนแม่เหล็กถาวรขดลวดที่ใช้เป็นขดลวดอาบน้ำยาฉนวน สนามแม่เหล็กจะเกิดขึ้นเมื่อจ่ายแรงดันไฟตรงให้มอเตอร์

2. ขั้วแม่เหล็ก (Pole Pieces) คือแกนสำหรับรองรับขดลวดสนามแม่เหล็กถูกยึดติดกับโครงมอเตอร์ด้านใน ขั้วแม่เหล็กทำมาจากแผ่นเหล็กอ่อนบางๆ อัดซ้อนกัน (Lamination Sheet Steel) เพื่อลดการเกิดกระแสไหลวนที่จะทำให้ความเข้มของสนามแม่เหล็กลดลง ขั้วแม่เหล็กทำหน้าที่ให้กำเนิดขั้วสนามแม่เหล็กมีความเข้มสูงสุด แทนขั้วสนามแม่เหล็กถาวร

3. โครงมอเตอร์ (Motor Frame) คือส่วนเปลือกหุ้มภายนอกของมอเตอร์และยึดส่วนอยู่กับที่ (Stator)ของมอเตอร์ไว้ภายในร่วมกับฝาปิดหัวท้ายของมอเตอร์ โครงมอเตอร์ทำหน้าที่เป็นทางเดินของเส้นแรงแม่เหล็กระหว่างขั้วแม่เหล็กให้เกิดสนามแม่เหล็กครบวงจร

4. อาเมเจอร์ (Armature) คือส่วนเคลื่อนที่ (Rotor) ถูกยึดติดกับเพลา (Shaft) และรองรับการหมุนด้วยที่รองรับการหมุน (Bearing) ตัวอาเมเจอร์ทำจากเหล็กแผ่นบางๆอัดซ้อนกัน ถูกเซาะร่องออกเป็นส่วนๆ เพื่อไว้พันขดลวดอาเมเจอร์ (Armature Winding) ขดลวดอาเมเจอร์เป็นขดลวดอาบน้ำยาฉนวน ร่องขดลวดอาเมเจอร์จะมีขดลวดพันอยู่และมีลิ่มไฟเบอร์อัดแน่นยึดขดลวดอาเมเจอร์ไว้ปลายขดลวดอาเมเจอร์ต่อไว้กับคอมมิวเตเตอร์อาเมเจอร์ผลักดันของสนามแม่เหล็กทั้งสอง ทำให้อาเมเจอร์หมุนเคลื่อนที่

5. คอมมิวเตเตอร์ (Commutator) คือส่วนเคลื่อนที่อีกส่วนหนึ่ง ถูกยึดติดเข้ากับอาเมเจอร์และเพลาร่วมกัน คอมมิวเตเตอร์ทำจากแท่งทองแดงแข็งประกอบเข้าด้วยกันเป็นรูปทรงกระบอก แต่ละแท่งทองแดงของคอมมิวเตเตอร์ถูกแยกออกจากกันด้วยฉนวนไมกา (Mica) อาเมเจอร์ คอมมิวเตเตอร์ทำหน้าที่เป็นขั้วรับแรงดันไฟตรงที่จ่ายมาจากแปรงถ่าน เพื่อส่งไปให้ขดลวดอาเมเจอร์

6. แปรงถ่าน (Brush) คือตัวสัมผัสกับคอมมิวเตเตอร์ ทำเป็นแท่งสี่เหลี่ยมผลิตมาจากคาร์บอนหรือแกรไฟต์ผสมผงทองแดง เพื่อให้แข็งและนำไฟฟ้าได้ดี มีสายตัวนำต่อร่วมกับแปรงถ่านเพื่อไปรับแรงดันไฟตรงที่จ่ายเข้ามา แปรงถ่านทำหน้าที่รับแรงดันไฟตรงจากแหล่งจ่าย จ่ายผ่านไปให้คอมมิวเตเตอร์

2.3.2 การควบคุมความเร็วของมอเตอร์กระแสตรง

มอเตอร์ไฟฟ้ากระแสตรงมี 3 ชนิดคือมอเตอร์แบบอนุกรมแบบขนานและแบบผสมในการควบคุมความเร็วต้องใช้วิธีการควบคุมที่แตกต่างกันไปตามชนิดของมอเตอร์ ซึ่งอาจจะใช้วิธีการควบคุมแบบพื้นฐานทั่วไปเช่นการควบคุมด้วยวิธีการใช้ตัวต้านทานปรับค่าโดยต่ออนุกรมกับมอเตอร์ หรือใช้วิธีการการควบคุมโดยการเปลี่ยนค่าของระดับแรงดันที่ป้อนให้กับมอเตอร์ การ

ควบคุมในวิธีดังกล่าวถึงแม้ว่าจะควบคุมความเร็วมอเตอร์ให้คงที่ได้แต่ที่ความเร็วต่ำจะส่งผลให้แรงบิดต่ำไปด้วยเช่นกัน

ข้อดีของมอเตอร์กระแสตรง คือ

1. การควบคุมแรงบิดหรือความเร็วทำได้ง่ายและดีมาก
2. มีผลตอบสนองต่อการเปลี่ยนแปลง (response) ได้รวดเร็ว
3. การปรับความเร็วสามารถทำได้ในช่วงกว้าง

ข้อเสียของมอเตอร์กระแสตรง คือ

1. การบำรุงรักษาสูงมากเนื่องจากมีส่วนสึกหรอของแปรงถ่าน
2. ราคาแพงมากเมื่อเทียบกับมอเตอร์กระแสสลับ (AC motor) ที่มีขนาดกำลังแรงม้าเท่ากัน
3. มีขนาดใหญ่กว่ามอเตอร์กระแสสลับที่ขนาดแรงม้าเท่ากัน
4. หาแหล่งจ่ายที่เป็นไฟกระแสตรงได้ยาก
5. ไม่สามารถนำไปใช้ในที่มีสารไวไฟได้

2.3.3 ระดับความแรงของมอเตอร์สั่นสะเทือน

ในนาฬิกาปลุกคนหูหนวกระดับแรงสั่นสะเทือนในการปลุก ใช้ระบบการกำหนดระดับแรงดันที่จ่ายไปยังมอเตอร์ โดยการใช้อุปกรณ์ทางอิเล็กทรอนิกส์ซีเล็กเตอร์ เป็นตัวเลือกการจ่ายค่าผ่านตัวต้านทานสามตัว โดยแบ่งระดับตัวต้านทานที่ใช้ในการแบ่งระดับความแรงในการสั่นสะเทือนดังนี้

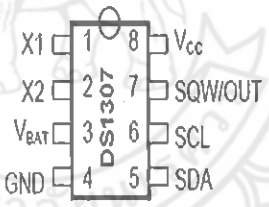
1. ตัวต้านทาน ขนาด 3 โอห์ม
2. ตัวต้านทาน ขนาด 5 โอห์ม
3. ตัวต้านทาน ขนาด 15 โอห์ม

ระดับที่กำหนดเรายึดหลักตามความเป็นจริงไม่มากเกินไป และไม่น้อยเกินไปถ้าตั้งค่ามากเกินไปจะทำให้มอเตอร์สั่นเบาจนไม่อาจปลุกได้ หรือ มากเกินไปอาจทำให้ให้เกิดความอันตรายต่อผู้ใช้ได้

2.4 ไอซีเวลาจริง (Real Time Clock) DS1307

ระบบฐานเวลา เป็นสิ่งสำคัญที่สามารถนำไปใช้ในอุปกรณ์อิเล็กทรอนิกส์ได้หลากหลาย ภายในไมโครคอนโทรลเลอร์เองก็มีไทเมอร์เพื่อใช้ในการจับเวลา หรือนำไปใช้เป็นฐานเวลาจริงได้เช่นกัน แต่เนื่องจากไมโครคอนโทรลเลอร์สามารถทำงานได้ต่อเมื่อมีไฟเลี้ยงเท่านั้น ดังนั้นการใช้ไทเมอร์ของไมโครคอนโทรลเลอร์ สร้างฐานเวลาจริงจึงไม่ เหมาะสมในบางแอพพลิเคชั่น

DS1307 เป็น IC ฐานเวลาของดัลลัสเซมิคอนดักเตอร์ (Dallas Semiconductor) มีบัสรับส่งข้อมูลแบบ I^2C ซึ่งเป็นแบบ 2 เส้นสามารถสื่อสารได้ 2 ทิศทาง (bi-direction bus) ฐานเวลาของ DS1307 นั้นสามารถเก็บข้อมูล วินาที, นาที, ชั่วโมง, วัน, วันที่, เดือน และปี ได้ ระบบเวลาสามารถ ทำงานโหมดรูปแบบ 24 ชั่วโมง หรือ 12 ชั่วโมง AM/PM ก็ได้ ภายมีระบบตรวจจับแหล่งจ่ายไฟ โดยถ้าแหล่งจ่ายไฟหลักถูกตัดไป DS1307 สามารถสวิตช์ไปใช้ไฟจากแบตเตอรี่ และทำงานต่อไป โดยที่ยังสามารถรักษาข้อมูลไว้ได้ โครงสร้างมีขาทั้งหมด 8 ขาดังแสดงในรูปที่ 2.4 และมีรายละเอียดการทำงานของขาต่าง ๆ ดังนี้



รูปที่ 2.4 ตำแหน่งขาไอซี RTC DS1307

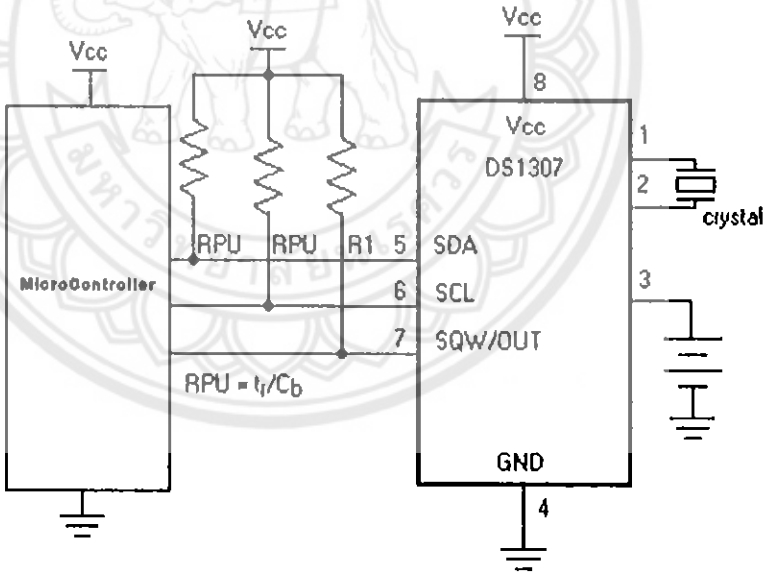
การทำงานของไอซี RTC DS1307 ในขาต่างๆมีดังนี้

- VCC: ใช้ต่อไฟเลี้ยง +5 โวลต์
- GND: ใช้ต่อกราวด์
- VBAT: ใช้ต่อกับแบตเตอรี่ 3 โวลต์ เพื่อรักษาการทำงาน ในกรณีที่ไม่มีไฟเลี้ยงจ่าย
- SDA: ขารับส่งข้อมูลด้วยระบบบัส I^2C
- SCL: ขาสัญญาณนาฬิกาสำหรับการรับส่งข้อมูลด้วยระบบบัส I^2C
- SQW/OUT: ขาเอาท์พุตสัญญาณสแควเวฟสามารถเลือกความถี่ได้

- X1, X2: ใช้ต่อกับคริสตอลความถี่มาตรฐาน 32.768 กิโลเฮิรตซ์เพื่อสร้างฐานเวลาจริงให้กับ IC

ระบบบัสข้อมูลแบบ I^2C (Inter-IC Communication) ได้ถูกพัฒนาขึ้นโดยบริษัทฟิลลิปส์ (Phillips) การรับส่งข้อมูลใช้สายสัญญาณเพียงแค่ 2 เส้น คือสายสัญญาณข้อมูล SDA (Serial Data line) และสายสัญญาณนาฬิกา SCL (Serial Clock line) มีการทำงานเป็นแบบเจ้านาย,ทาส โดยอุปกรณ์ที่ทำหน้าที่เป็นเจ้านาย (ไมโครคอนโทรลเลอร์) จะควบคุมการรับส่งข้อมูล และควบคุมสัญญาณนาฬิกาบน SCL ส่วนอุปกรณ์ทาส (DS1307) นั้นจะทำงานภายใต้การควบคุมของอุปกรณ์ที่ทำหน้าที่เป็นเจ้านาย

การต่อใช้งานร่วมกับไมโครคอนโทรลเลอร์ด้วยระบบบัส I2C นั้นสามารถทำได้โดยต่อตัวต้านทาน Pull up ดังแสดงในรูปที่ 2.5 ในกรณีที่ต้องการต่อร่วมกับอุปกรณ์ทาสหลายตัว ก็สามารถทำได้โดยต่ออุปกรณ์ทาสขนานกันไป การติดต่อสื่อสารระหว่างอุปกรณ์เจ้านายกับทาสแต่ละตัวนั้นจะถูกแยกโดยที่อยู่ของอุปกรณ์ทาสซึ่งจะถูกส่งจากอุปกรณ์เจ้านายไปยังอุปกรณ์ทาสก่อนเริ่มการรับส่งข้อมูล



รูปที่ 2.5 การเชื่อมต่อ DS1307 เข้ากับไมโครคอนโทรลเลอร์

การรับส่งข้อมูลแบบ I^2C นั้นมีข้อกำหนดอยู่ 2 ประการด้วยกันคือ

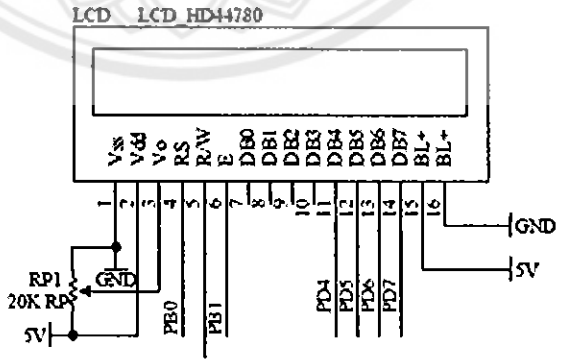
1. การรับส่งข้อมูลจะเริ่มขึ้นได้เมื่อบัสมีสถานะว่างเท่านั้น

2. ในช่วงที่ทำการรับส่งข้อมูลอยู่ สายสัญญาณข้อมูลต้องไม่เปลี่ยนสถานะในช่วงที่สายสัญญาณนาฬิกามีสถานะเป็นลอจิก “ 1 ” ถ้าสายสัญญาณข้อมูลมีการเปลี่ยนสถานะในช่วงที่สายสัญญาณนาฬิกาเป็นลอจิก “ 1 ” จะถือว่าเป็นสัญญาณควบคุมการรับส่งข้อมูล

2.5 หน้าจอแสดงผลผลึกเหลว (Liquid Crystal Display)

หน้าจอแสดงผลผลึกเหลว หรือ LCD (Liquid Crystal Display) คือหน้าจอแสดงผลขนาด 16 ตัวอักษร 2 บรรทัด ในปัจจุบัน หน้าจอผลึกเหลวเป็นที่นิยมมากเนื่องจากความสวยงามและความหลาหลายในการแสดงผล มากว่าหน้าจอแสดงผลชนิดอื่น อีกทั้งยังมีความสว่างในตัว ต่างจากหน้าจอแสดงผลแบบอื่นๆ ลักษณะของหน้าจอแสดงผลผลึกเหลวจะแสดงในรูปที่ 2.6 และรูปโครงสร้างการทำงานรูปที่ 2.7

รูปที่ 2.6 หน้าจอแสดงผลผลึกเหลว



รูปที่ 2.7 โครงสร้างการทำงานของหน้าจอแสดงผลผลึกเหลว

หน้าจอแสดงผลแบบผลึกเหลว จะมีขาอยู่ 16 ขา โดยแต่ละขาจะมีการทำงานดังนี้

- LCD RS ต่อกับ ดิจิตอล 12

- LCD Enable ต่อกับ ดิจิตอล 11

- LCD D4 ต่อกับ ดิจิตอล 5

- LCD D5 ต่อกับ ดิจิตอล4

- LCD D6 ต่อกับ ดิจิตอล 3

- LCDD7ต่อกับ ดิจิตอล 2

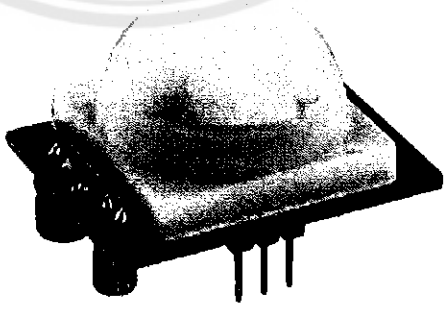
- LCD Gnd ต่อกับ LCD R/W ต่อเข้ากับ ขาข้างใดข้างหนึ่งของ พอร์ต 10 กิโลโอห์มแล้วต่อเข้ากับขากราวด์

- LCD Vcc ต่อกับ ขาอีกด้านของพอร์ต 10 กิโลกโอห์ม แล้วต่อเข้ากับไฟเลี้ยง 5 โวลต์ LCD VO ต่อกับ ขากลางของ พอร์ต 10 กิโลโอห์ม

- LCD VO ต่อเข้ากลับ ขากลางของ พอร์ต 10 กิโลโอห์ม

2.6 เซ็นเซอร์ตรวจจับการเคลื่อนไหว (PIR Motion Sensor)

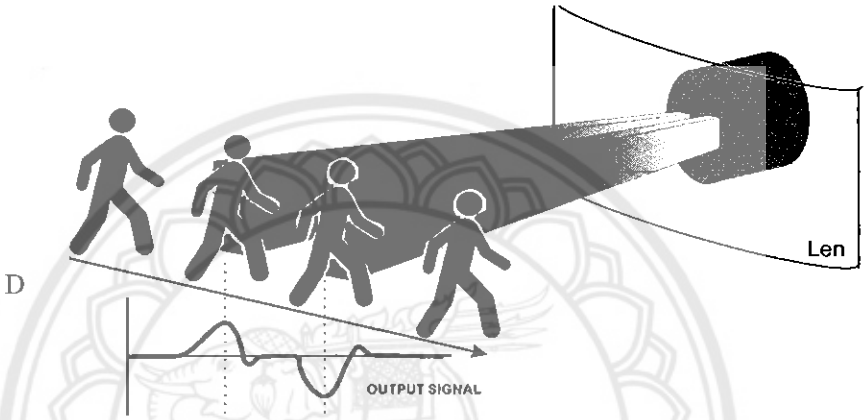
เซ็นเซอร์ตรวจจับการเคลื่อนไหว เป็นส่วนการทำงานพิเศษที่เพิ่มขึ้นมากับตัวนาฬิกาปลุกคนหูหนวก เพื่อให้มีการทำงานที่เสมือนนาฬิกาทั่วไปมากขึ้น คือการที่สามารถปิดการทำงานแสงไฟหน้าจอแสดงผลขณะที่ไม่มีผู้ใช้ เพื่อลดการใช้แสงที่ไม่จำเป็นและตอบสนองการทำงานในช่วงเวลาที่ผู้ใช้ไม่ต้องการแสงจากนาฬิกาตลอดเวลา ลักษณะของเซ็นเซอร์ตรวจจับการเคลื่อนไหวและลักษณะการทำงานจะแสดงอยู่ในรูปที่ 2.8



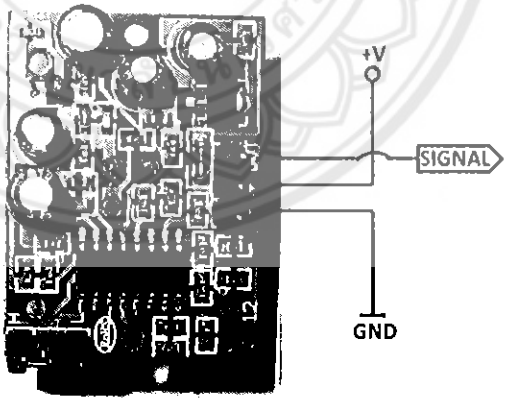
รูปที่ 2.8 เซ็นเซอร์ตรวจจับการเคลื่อนไหว

เซ็นเซอร์ตรวจจับความเคลื่อนไหว เป็นอุปกรณ์ที่แปลงการตรวจจับความเคลื่อนไหวเป็นสัญญาณไฟฟ้า เซ็นเซอร์ตรวจจับโดยทั่วไปมีหลายประเภทแต่ประเภทที่เราเลือกใช้คือเซ็นเซอร์ตรวจจับการเคลื่อนไหวชนิด Passive Infrared Sensor (PIR) โดยจะมีลักษณะการทำงานเป็น

เซ็นเซอร์ที่รับความร้อนจากร่างกายเมื่อเคลื่อนที่ ไม่มีการปล่อยพลังงานออกมาจากเซ็นเซอร์ ความร้อนวัดได้จากการเปลี่ยนแปลงระดับรังสีอินฟราเรดที่ปล่อยออกมาจากวัตถุ เมื่อวัตถุเคลื่อนที่ (สิ่งมีชีวิตทุกชนิดจะแผ่รังสีอินฟราเรดออกมาจากตัวเอง การแผ่รังสีดังกล่าวเกิดจากการเคลื่อนที่ของอิเล็กตรอนในอะตอม ปริมาณรังสีจะมีมากน้อยตามแต่โครงสร้างทางเคมีและอุณหภูมิของวัตถุหรือสิ่งมีชีวิตนั้นๆ) จึงทำให้สามารถตรวจจับสัญญาณลอจิกที่เปลี่ยนแปลงที่ขาเอาท์พุตได้ ลักษณะการทำงานและจุดเชื่อมต่อสำหรับใช้งานจะแสดงในรูปที่ 2.9 และ 2.10 ตามลำดับ



รูปที่ 2.9 ลักษณะการทำงานของเซ็นเซอร์ตรวจจับการเคลื่อนไหว



รูปที่ 2.10 จุดเชื่อมต่อสำหรับใช้งานของเซ็นเซอร์ตรวจจับการเคลื่อนไหว

จุดเชื่อมต่อสำหรับใช้งานมีทั้งหมด 3 จุด

1. ขาไฟเลี้ยง (+) สำหรับต่อไฟเลี้ยงแรงดัน +3.3 ถึง +5 โวลต์
2. ขาเอาท์พุต (OUT) สำหรับต่อเข้ากับขาอินพุตของไมโครคอนโทรเลอร์
3. ขากราวด์ (-) สำหรับต่อกราวด์ 0 โวลต์

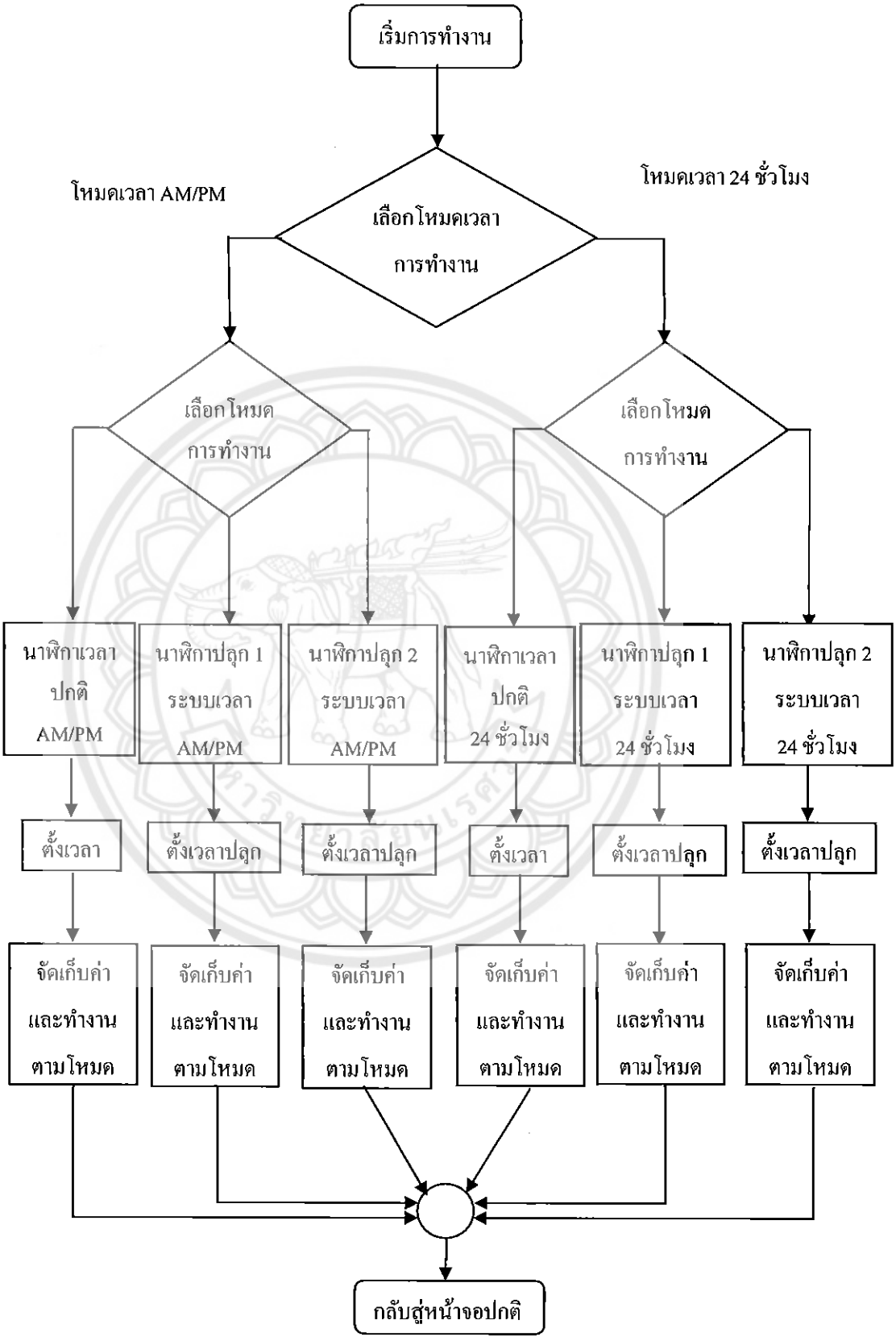
บทที่ 3

การออกแบบและสร้างนาฬิกาปลุกคนหูหนวก

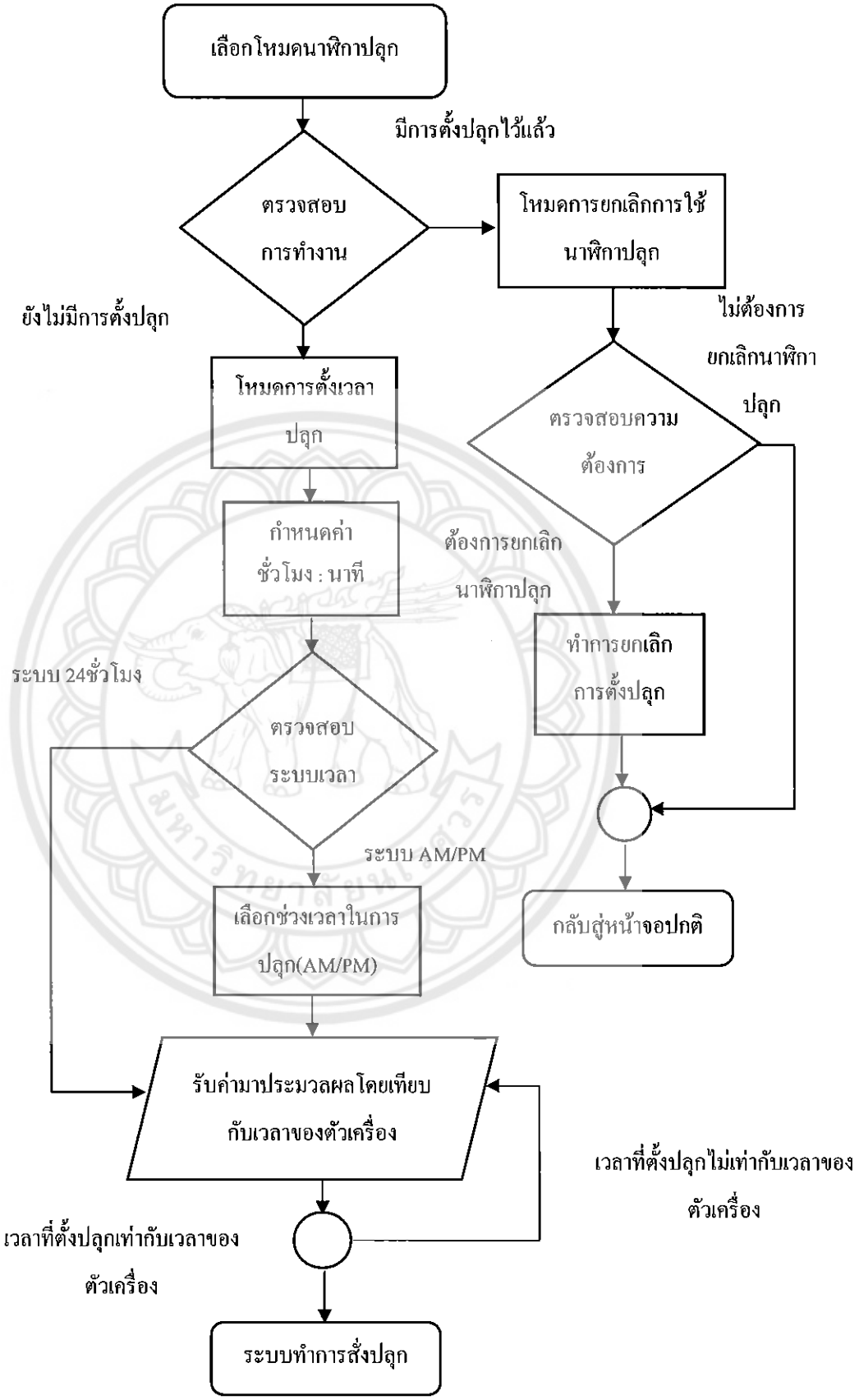
ในบทนี้จะบอกถึงขั้นตอนในการออกแบบและสร้างนาฬิกาปลุกคนหูหนวก รวมถึงอุปกรณ์ต่างๆที่ใช้ในการทำด้วย

3.1 ขั้นตอนการทำงานของนาฬิกาปลุกคนหูหนวก

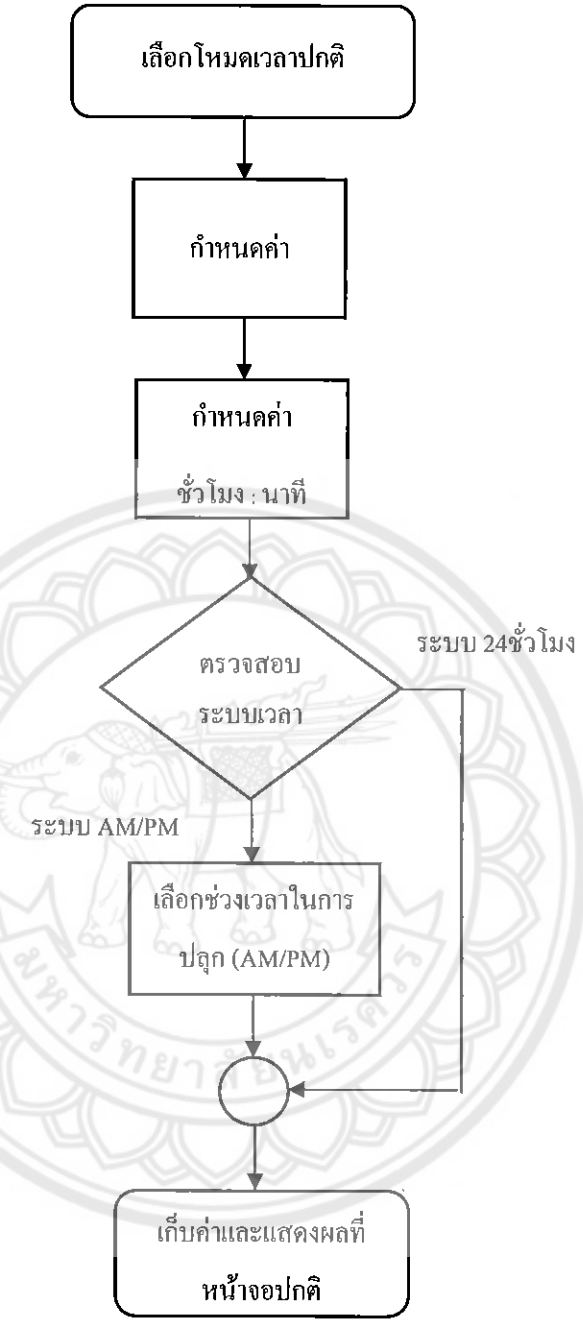
นาฬิกาปลุกคนหูหนวกมีขั้นตอนการทำงานแบ่งออกเป็น 2 โหมดการทำงานหลักได้แก่ โหมดการทำงานระบบเวลา Ante Meridian (AM) / Post Meridian (PM) และโหมดการทำงานระบบเวลา 24 ชั่วโมง โดยระบบการทำงานต่างๆภายในแต่ละโหมดจะคล้ายคลึงกัน ต่างกันที่ช่วงเวลาที่แต่ละโหมดกำหนด โดยระบบการทำงานภายในแต่ละโหมดจะมีด้วยกัน 3 ระบบ คือ การตั้งเวลาปกติ การตั้งปลุกครั้งที่หนึ่ง และการตั้งปลุกครั้งที่สอง ระบบโครงสร้างการทำงานทั้งหมดและโครงสร้างการทำงานในโหมดต่างๆจะแสดงในรูปที่ 3.1 รูปที่ 3.2 และ รูปที่ 3.3 ตามลำดับ



รูปที่ 3.1 ขั้นตอนการทำงานของนาฬิกาปลุกคนหูหนวก



รูปที่ 3.2 โหมดการทำงานนาฬิกาปลุก



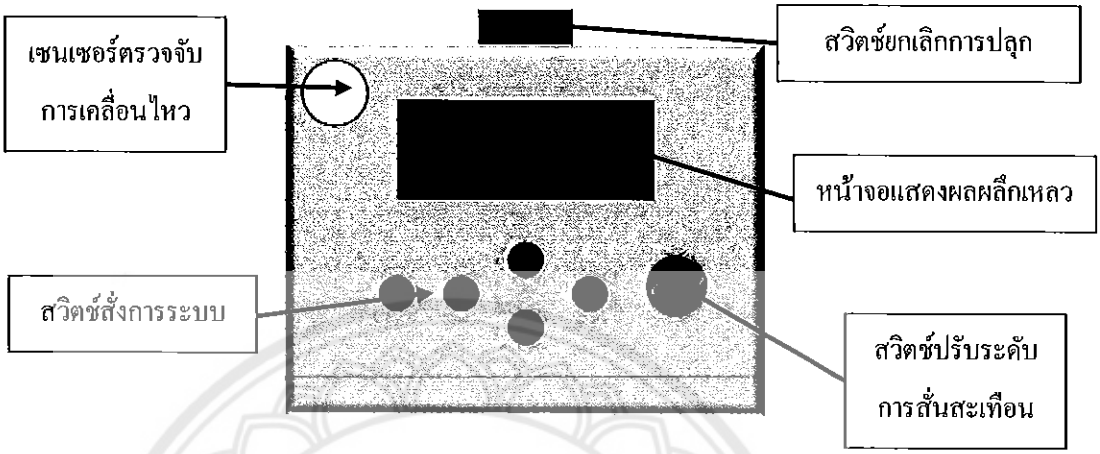
รูปที่ 3.3 โหมดการทำงานนาฬิกาปกติ

3.2 การออกแบบและการสร้างนาฬิกาปลุกคนหูหนวก

โครงสร้างนาฬิกาปลุกคนหูหนวกจะประกอบไปด้วย 2 ส่วนใหญ่ๆคือส่วนของตัวเครื่องนาฬิกาปลุกคนหูหนวกและส่วนของชุดมอเตอร์สั่นสะเทือน

3.2.1 โครงสร้างส่วนควบคุมและแสดงผลของนาฬิกาปลุกคนหูหนวก

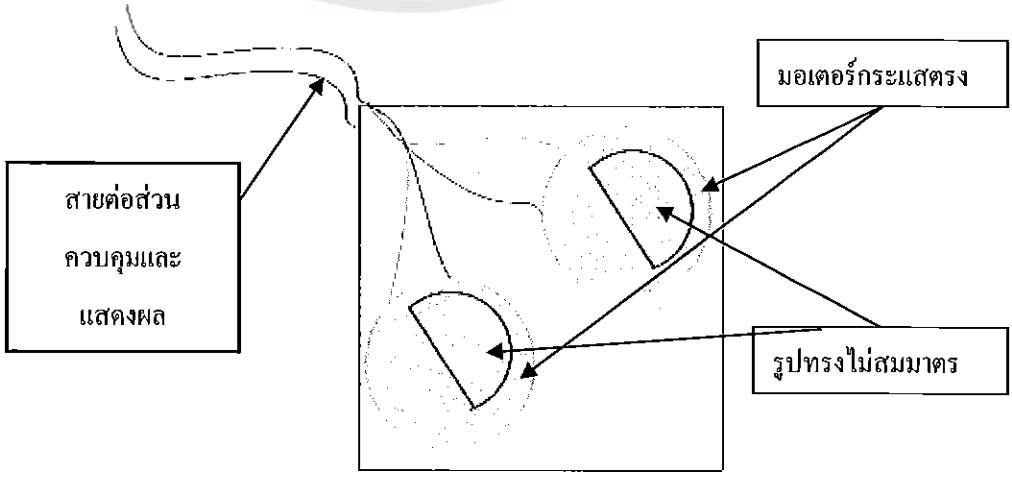
โครงสร้างในส่วนตัวเครื่องนาฬิกาปลุก ประกอบไปหน้าจอแสดงผล ปุ่มตั้งค่าตัวเลข ปุ่มเมนู ปุ่มตั้งความแรงในการสั่นปลุก ดังรูปที่ 3.4



รูปที่ 3.4 โครงสร้างนาฬิกาปลุกคนหูหนวก

3.2.2 โครงสร้างชุดมอเตอร์สั่นสะเทือน

โครงสร้างในส่วนชุดมอเตอร์สั่นสะเทือน เป็นส่วนที่ทำหน้าที่รับคำสั่งเมื่อถึงเวลาที่กำหนด หลักการคือการสั่งให้มอเตอร์กระแสตรงทำงานโดยในส่วนมอเตอร์ได้ติดตั้งกับโครงสร้างยึดมอเตอร์ให้อยู่กับที่และที่ปลายของมอเตอร์ต่อกับรูปทรงที่ไม่สมมาตรเพื่อสร้างแรงสั่นสะเทือนเมื่อมอเตอร์หมุน ส่วนประกอบของส่วนนี้ประกอบไปด้วย ส่วนต่อสายกับส่วนโครงสร้างนาฬิกา มอเตอร์กระแสตรง รูปทรงไม่สมมาตร ดังรูปที่ 3.5



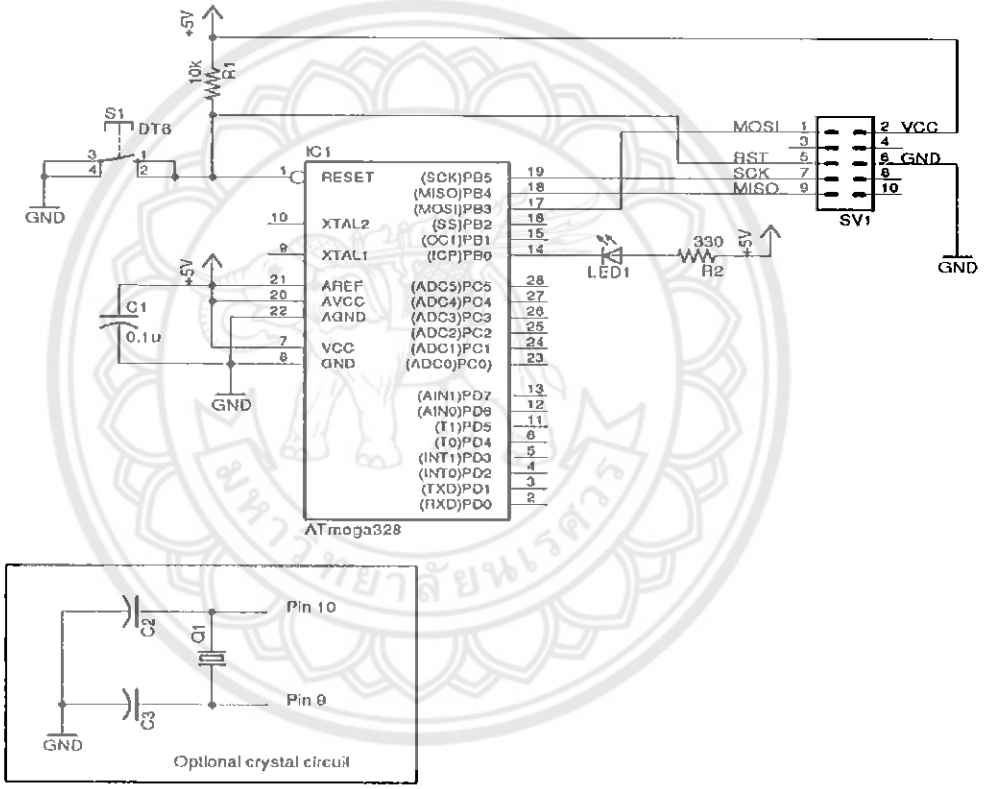
รูปที่ 3.5 โครงสร้างชุดมอเตอร์สั่นสะเทือน

3.3 วงจรที่ใช้ในการทำงานของนาฬิกาปลุกคนหูหนวก

วงจรที่ใช้ในนาฬิกาปลุกคนหูหนวก ประกอบไปด้วย

3.3.1 วงจรไมโครคอนโทรลเลอร์

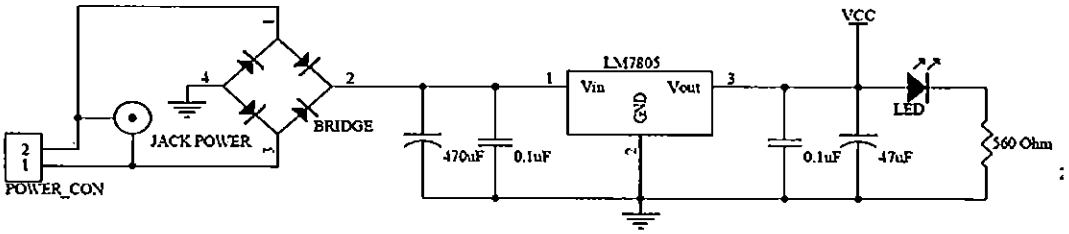
วงจรไมโครคอนโทรลเลอร์สามารถต่อได้ตามวงจรในรูปที่ 3.6 โดยแหล่งจ่ายไฟ 5 โวลต์ นั้นใช้จากแหล่งจ่ายไฟของวงจรจ่ายไฟ จะเห็นได้ว่าไมโครคอนโทรลเลอร์มีพอร์ต อินพุต/เอาท์พุต อยู่ทั้งหมด 3 พอร์ต (พอร์ต B – พอร์ต C) แต่ละพอร์ตสามารถทำงานเป็นพอร์ต อินพุต หรือ เอาท์พุต นอกจากนี้บางพอร์ตยังสามารถทำงานพิเศษเฉพาะทางได้อีก



รูปที่ 3.6 วงจรไมโครคอนโทรลเลอร์

3.3.2 วงจรแปลงไฟ และจ่ายไฟ

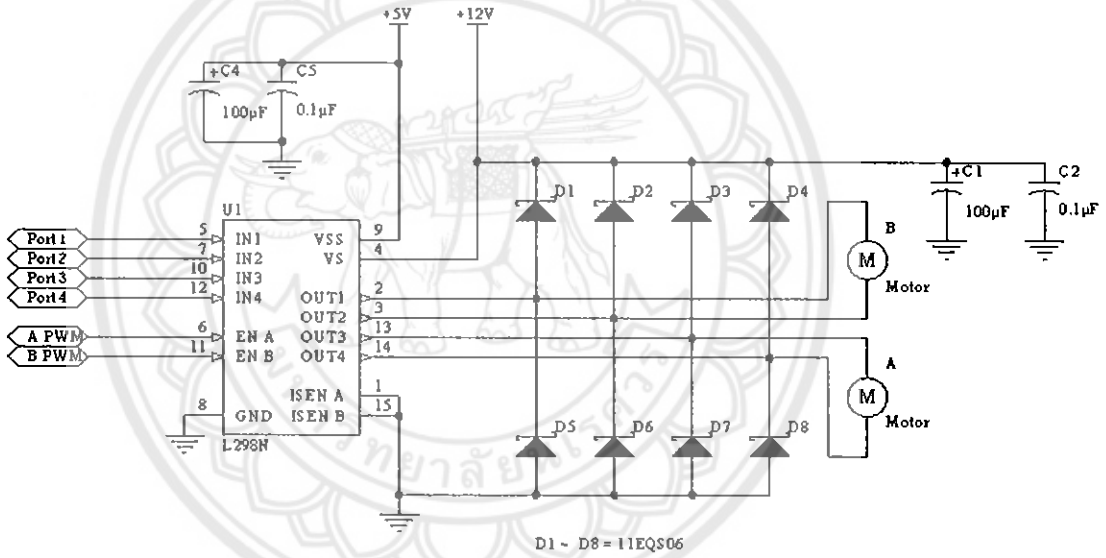
วงจรแปลงไฟ และจ่ายไฟจะใช้หม้อแปลงแปลงไฟฟ้ากระแสสลับเป็นกระแสตรง ผ่านวงจรเรียงกระแสแบบบริดจ์เพื่อป้องกันการสลับขั้วของแหล่งจ่าย จะใช้ไอซีหมายเลข LM7805 ซึ่งทำหน้าที่แปลงไฟจาก 9-12 โวลต์ ให้เป็นไฟกระแสตรง 5 โวลต์เพื่อจ่ายเป็นไฟเลี้ยงให้กับไมโครคอนโทรลเลอร์ต่อไป ซึ่งสามารถต่อวงจรได้ดังรูปที่ 3.7



รูปที่ 3.7 วงจรแปลงไฟและจ่ายไฟ

3.3.3 วงจรขับมอเตอร์

ใช้ในการขับมอเตอร์ให้มอเตอร์เกิดการหมุน การทำงานของวงจรขับมอเตอร์นั้นจะเหมือนกับการทำงานของสวิตช์ปิดเปิดตามสัญญาณที่ชุดควบคุมส่งออกมาและใช้ในการควบคุมความเร็วของมอเตอร์



รูปที่ 3.8 วงจรขับเคลื่อนมอเตอร์

จากรูปที่ 3.8 แสดงลักษณะของวงจรขับมอเตอร์โดยใช้วงจรรวมหมายเลข L298N โดยทำการป้อนไฟเลี้ยง 5 โวลต์ เพื่อจ่ายให้กับไอซีและทำการป้อนไฟเลี้ยง 9-12 โวลต์ เพื่อขับเคลื่อนมอเตอร์กระแสตรงเมื่อสัญญาณที่ขา EN A และ EN B เป็นลอจิก “1” จะทำให้มอเตอร์ A และมอเตอร์ B อยู่ในสภาวะพร้อมทำงาน เมื่อขา IN 1 และขา IN 3 เป็นลอจิก “1” มอเตอร์ A และมอเตอร์ B จะหมุนตามเข็มนาฬิกา (เคลื่อนที่ไปข้างหน้า) เมื่อขา IN 2 และขา IN 4 เป็นลอจิก “1” มอเตอร์ A และมอเตอร์ B จะหมุนทวนเข็มนาฬิกา (เคลื่อนที่ถอยหลัง)

บทที่ 4
ผลการทดสอบและผลการวิเคราะห์

หลังจากศึกษาทฤษฎี หลักการทำงานและลงมือสร้างนาฬิกาปลุกคนหูหนวกแล้ว ในบทนี้จะเป็นการทดสอบการทำงานของระบบนาฬิกาปลุกคนหูหนวกโดยแบ่งการทดสอบเป็น 3 หัวข้อดังนี้

1. การทดสอบการทำงานของระบบเวลาจริง
2. การทดสอบระดับการสั่นสะเทือนของอุปกรณ์สั่นปลุก
3. การทดสอบการทำงานของเซ็นเซอร์ตรวจจับการเคลื่อนไหว

4.1 การทดสอบการทำงานของระบบเวลาจริง

การทดลองนี้จะทำการทดลองการตั้งเวลาของตัวเครื่อง เทียบกับเครื่องเล่นเสียงดิจิตอลของบริษัท APPLE (IPOD) ซึ่งมีการทำงานในส่วนแสดงเวลาที่ตรงกับเวลาสากล การทดลองนี้จะทำการทดลองรวมเป็นเวลาทั้งสิ้น 24 ชั่วโมง จะมีการตรวจสอบเวลาจากเครื่องทั้ง 2 เทียบเคียงกันทุกๆ 2 ชั่วโมงดังรูปที่ 4.1 และแสดงผลการทดลองในตารางที่ 4.1 และตารางที่ 4.2

(ก) (ข)

รูปที่ 4.1 ตัวอย่างการทดสอบการทำงานของระบบเวลาจริง

(ก) หน้าจอแสดงเวลาของนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์

(ข) หน้าจอแสดงเวลาของเครื่องเล่นเสียงดิจิตอล (IPOD)

ตารางที่ 4.1 ผลการทดสอบการทำงานระบบเวลาจริง (ก่อนเที่ยงวัน)

รายการ	ค่าเวลาที่แสดง(ชั่วโมง)					
นาฬิกาปลุกคนหูหนวก	1.30AM	3.30AM	5.30AM	7.30AM	9.30AM	11.30AM
เครื่องเล่นเสียงดิจิตอล (IPOD)	1.30	3.30	5.30	7.30	9.30	11.30

ตารางที่ 4.2 ผลการทดสอบการทำงานระบบเวลาจริง (หลังเที่ยงวัน)

รายการ	ค่าเวลาที่แสดง(ชั่วโมง)					
นาฬิกาปลุกคนหูหนวก	1.30PM	3.30PM	5.30PM	7.30PM	9.30PM	11.30PM
เครื่องเล่นเสียงดิจิตอล (IPOD)	13.30	15.30	17.30	19.30	21.30	23.30

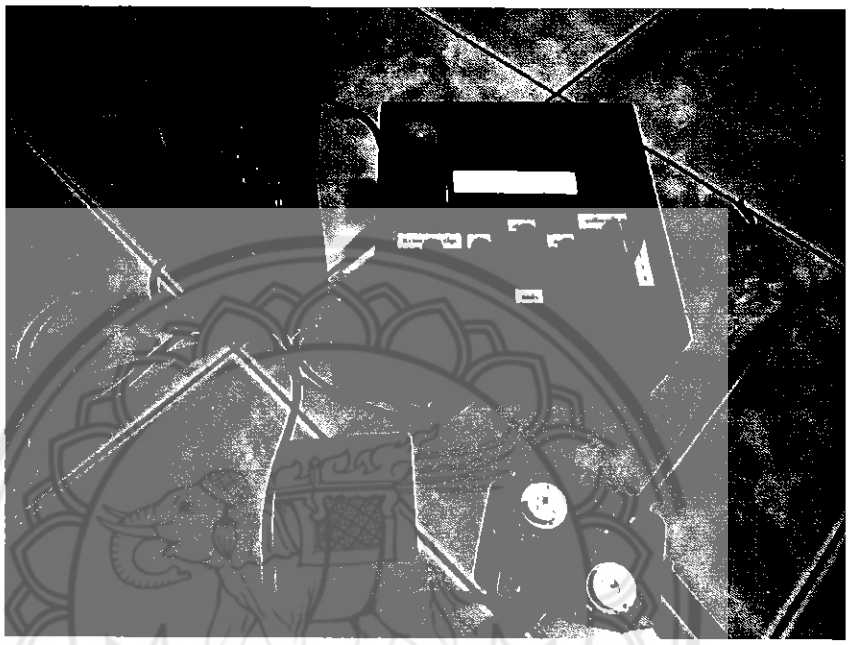
จากการทดสอบการทำงานระบบเวลาจริงของนาฬิกาปลุกคนหูหนวก เปรียบเทียบกับนาฬิกาของเครื่องเล่นเสียงดิจิตอล(IPOD) จะเห็นได้ว่าระบบเวลาของนาฬิกาปลุกคนหูหนวกมีการทำงานที่ตรงกับระบบเวลาสากล ในระยะเวลา 1 วัน

4.2 การทดสอบระดับการสั่นสะเทือนของอุปกรณ์สั่นปลุก

ระบบการสั่นปลุกในตัวอุปกรณ์นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์นี้ จะใช้หลักการการจ่ายกระแสผ่านอุปกรณ์ซีเล็กเตอร์แล้วผ่านตัวต้านทานที่กำหนดก่อนเข้าสู่มอเตอร์สั่นสะเทือน โดยแต่ละระดับจะขึ้นอยู่กับค่าตัวต้านทานที่ซีเล็กเตอร์ทำการเลือกในแต่ละระดับจะมีค่าตัวต้านทานดังนี้

1. ตัวต้านทานขนาด 15 โอห์ม
2. ตัวต้านทานขนาด 5 โอห์ม
3. ตัวต้านทานขนาด 3 โอห์ม
4. ไม่มีตัวต้านทาน (ให้ซีเล็กเตอร์จ่ายไฟเข้ามอเตอร์โดยตรง)

การทดลองในส่วนนี้จะเป็นการวัดระดับการสั่นเทือนของอุปกรณ์สั่นปลุก โดยการทดลองนี้จะใช้การวัดค่าแรงดันไฟฟ้าที่จ่ายให้มอเตอร์สั่นสะเทือนในระดับระดับนั้น โดยในแต่ระดับจะทำการวัดค่าแรงดันด้วยกัน 5 ครั้ง แล้วนำค่านั้นมาหาค่าแรงดันเฉลี่ยในแต่ละระดับ โดยมีวิธีการทดลองดังรูปที่ 4.2 และแสดงผลการทดลองดังตารางที่ 4.3



รูปที่ 4.2 วิธีทำการทดสอบแรงดันที่จ่ายให้กับมอเตอร์สั่นสะเทือน

ตารางที่ 4.3 ผลการทดสอบค่าแรงดันไฟฟ้าที่มอเตอร์สั่นสะเทือนได้รับในแต่ละระดับ

ระดับการสั่นสะเทือน	ค่าแรงดันไฟฟ้าที่มอเตอร์สั่นสะเทือนได้รับในแต่ละระดับ (โวลต์)					
	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	ค่าเฉลี่ย
1	2.032	1.937	1.926	1.978	2.104	1.988
2	3.236	3.334	3.258	3.124	3.153	3.221
3	4.872	4.651	4.743	4.632	4.722	4.724
4	5.205	5.102	5.300	5.304	4.950	5.172

หมายเหตุ : การสั่นปลุกในแต่ละครั้งจะมีค่าแรงดันที่ไม่คงที่ดังนั้นค่าในตารางจึงมาจากค่าแรงดันสูงสุดในการทดสอบแต่ละครั้ง

จากการทดสอบการทำงานส่วนสั่นปลุกของนาฬิกาปลุกคนหูหนวก จะเห็นได้ว่าค่าแรงดันที่จ่ายให้มอเตอร์ในแต่ระดับมีค่าที่ต่างกันมากในระดับที่ 1 ระดับที่ 2 และระดับที่ 3 ในหน่วยโวลต์ แต่ในระดับที่ 3 และระดับที่ 4 จะมีค่าแรงดันต่างกันเล็กน้อย จึงทำให้การสั่นสะเทือนในสองระดับมีความแรงในการสั่นต่างกันไม่มาก หรือไม่ต่างกันเลย

4.3 การทดสอบการทำงานของเซ็นเซอร์ตรวจจับการเคลื่อนไหว

นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์ จะมีระบบการทำงานพักหน้าจอโดยใช้เซ็นเซอร์ตรวจจับการเคลื่อนไหว มีหลักการการทำงานคือ เมื่อไม่มีผู้ใช้งานหรือผู้ใช้งานไม่ได้อยู่ในระยะการตรวจจับของเซ็นเซอร์ ระบบจะทำการหยุดการทำงานในส่วนของไฟหน้าจอแสดงผล และเมื่อมีการใช้งานหรือเซ็นเซอร์ทำงานการทำงานในส่วนของไฟหน้าจอแสดงผลจะทำงานปกติโดยมีลักษณะการทดสอบดังรูปที่ 4.3

รูปที่ 4.3 วิธีทดสอบการทำงานเซ็นเซอร์ตรวจจับการเคลื่อนไหว

4.3.1 ทดสอบระยะการตรวจจับสูงสุดของเซ็นเซอร์

จากการศึกษาเซ็นเซอร์ตรวจจับการเคลื่อนไหวชนิดตรวจจับความร้อนจากการเคลื่อนไหวของสิ่งมีชีวิต จะมีรัศมีการทำงานที่ความยาว 10 เมตร กว้าง 55 องศาในแนวระดับ และ

แนวดิ่ง ในการทดสอบนี้จะเป็นการทดสอบหาระยะทางไกลสุดที่เซ็นเซอร์จะตรวจจับได้โดยจะทำการเคลื่อนที่เข้าหานาฬิกาปลุกโดยทางหน้าของนาฬิกา แสดงผลการทดลองในตารางที่ 4.4

ตารางที่ 4.4 ผลการทดสอบระยะการทำงานสูงสุดที่เซ็นเซอร์ตรวจจับได้

การทดสอบระยะการทำงานสูงสุดของเซ็นเซอร์ (เมตร)						
ระยะการทดสอบ	10	9	8	7	6	5
การทำงานของเซ็นเซอร์	ไม่ทำงาน	ไม่ทำงาน	ไม่ทำงาน	ไม่ทำงาน	ไม่ทำงาน	ทำงาน

จากการทดสอบระยะการทำงานสูงสุดที่เซ็นเซอร์ตรวจจับได้ ของเซ็นเซอร์ตรวจจับการเคลื่อนไหวในนาฬิกาปลุกคนหูหนวก จะเห็นได้ว่าเซ็นเซอร์มีการตรวจจับการเคลื่อนไหวของสิ่งมีชีวิต ที่ระยะไกลสุดไม่เกิน 5 เมตร ในแนวตรง

4.3.2 ทดสอบระยะเวลาก่อนการทำงานโหมดพักหน้าจอ

การทำงานในโหมดพักหน้าจอจะถูกกำหนดในโปรแกรมให้ทำงานขณะที่ไม่มีผู้ใช้งาน หรือไม่มีสิ่งมีชีวิตเคลื่อนไหวในรัศมีการทำงานของเซ็นเซอร์ เป็นระยะเวลา 10 วินาที โดยในการทดสอบนี้จะทำการทดสอบระยะเวลาก่อนการทำงานในโหมดพักหน้าจอ จำนวน 5 ครั้งพร้อมทั้งหาค่าเวลาเฉลี่ย และแสดงผลการทดสอบในตารางที่ 4.5

ตารางที่ 4.5 ผลการทดสอบระยะเวลาก่อนการทำงานโหมดพักหน้าจอ

ระยะเวลาก่อนการทำงานโหมดพักหน้าจอ (วินาที)						
รายการการทดสอบ	**ครั้งที่ 1**	**ครั้งที่ 2**	**ครั้งที่ 3**	**ครั้งที่ 4**	**ครั้งที่ 5**	**เฉลี่ย**
ระยะเวลา	12	11	13	12	14	12.4

จากการทดสอบระยะเวลาก่อนการทำงานโหมดพักหน้าจอของนาฬิกาปลุกคนหูหนวก จะเห็นได้ว่า ความไวในการตรวจจับของเซ็นเซอร์ในแต่ละครั้งจะไม่เท่ากันจึงต้องหาค่าเฉลี่ย โดยค่าเฉลี่ยเวลาที่ออกมานั้นจะต่างจากค่าเวลาที่ตั้งไว้ในโปรแกรมเพียงเล็กน้อย

บทที่ 5
สรุปผลและข้อเสนอแนะ

จากการศึกษา ออกแบบ ประกอบ ทดสอบ และทำการปรับปรุงชิ้นงานจนเป็นนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ โดยใช้ระยะเวลาดำเนินโครงงาน 2 ภาคการศึกษา ทำให้ทราบถึงปัญหาที่เกิดขึ้นในการทำโครงงาน สำหรับบทนี้จะเป็นการสรุปผลที่ได้จากการทดสอบในโครงงานพร้อมเสนอแนะแนวทางในการทำนำโครงงานนี้ไปพัฒนาหรือต่อยอดให้มีประสิทธิภาพมากยิ่งขึ้น

5.1 สรุปผลการดำเนินโครงการ

จากการทดสอบการทำงานของนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์จะเห็นได้ว่า นาฬิกาปลุกคนหูหนวกมีระบบการทำงานให้เลือกทำงานได้ทั้งสองระบบเวลา คือระบบเวลาแบบ Ante Meridian (AM) / Post Meridian (PM) และ 24 ชั่วโมง มีระบบนาฬิกาปลุกที่สามารถตั้งปลุกได้สองครั้งในทั้งสองระบบเวลา นอกจากนี้นาฬิกาปลุกคนหูหนวกยังมีผลการทดสอบเพิ่มเติมดังนี้

1. นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์มีระบบการทำงานเวลาจริง และมีการแสดงเวลาที่มีค่าตรงกับเวลาสากลเมื่อทำการเทียบค่าเวลากับอุปกรณ์อิเล็กทรอนิกส์เครื่องเล่นเสียงดิจิตอล (IPOD)

2. นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์มีระบบแจ้งปลุกแบบการสั่นสะเทือนที่สามารถเลือกระดับการสั่นสะเทือนได้ โดยระดับการสั่นสะเทือนถูกกำหนดด้วยค่าตัวต้านทานที่ตัวซีเล็กเตอร์เป็นตัวกำหนดให้แรงดันไฟฟ้าตกคร่อมก่อนเข้ามอเตอร์

3. นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ มีการติดตั้งระบบการตรวจจับการเคลื่อนไหว โดยขณะมีการเคลื่อนไหวบริเวณหน้าจอของตัวเครื่อง หน้าจอแอลซีดีจะมีการแสดงผลปกติ แต่เมื่อไม่มีการกระทำใดๆกับนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ตามเวลาที่กำหนด ระบบจะทำการสั่งการให้หน้าจอแอลซีดีหยุดการทำงานในส่วนของไฟหน้าจอเพื่อแก้ปัญหา แสงรบกวนผู้ใช้ขณะพักผ่อน

5.2 ปัญหาและการแก้ไข

1. นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ เป็นอุปกรณ์ที่จำเป็นต้องมีขนาดไม่ใหญ่จนเกินไป แต่วงจรที่ใช้ในการทำงานมีหลายวงจร จึงแก้ปัญหาโดยการรวมวงจรเพื่อลดขนาดให้มีขนาดเล็กลง

2. การเขียนโปรแกรมภายในนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ มีความซับซ้อน และต้องการให้มีระบบที่ตอบสนองความต้องการของผู้ใช้ จึงแก้ไขปัญหาด้วยการใช้โปรแกรมเข้ามาช่วยในการเขียน

3. ระบบการสั่นสะเทือนของนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ จากเดิมจะมีการรับสัญญาณการทำงานจากไมโครคอนโทรเลอร์ พร้อมทั้งแบ่งไฟฟ้ามาเพื่อใช้เป็นแรงดันในการทำงาน แต่เกิดปัญหาไฟเลี้ยงไมโครคอนโทรเลอร์ไม่เพียงพอ จึงแก้ปัญหาด้วยการต่อไฟที่เข้าส่วนสั่นสะเทือนแยกจากไมโครคอนโทรเลอร์

4. นาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ใช้การสั่งการการทำงานจากสวิตช์ทั้งหมด จะมีปัญหาในส่วนของสวิตช์ทำงานเกินคำสั่งหรือการกดสองครั้งในทีเดียว จึงมีการแก้ไขปัญหาที่โปรแกรม โดยให้ในการกดแต่ละครั้งมีระยะเวลาหน่วงการทำงาน

5.3 ข้อเสนอแนะและแนวทางการพัฒนา

จากปัญหาที่พบในการนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ ทำให้ศักยภาพในการทำงานของเครื่องลดลง จึงต้องมีการศึกษาแนวทางวิธีในการแก้ไขปัญหาที่เกิดขึ้นเพื่อพัฒนานาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรเลอร์ ดังต่อไปนี้

1. จัดทำวงจรให้มีขนาดเล็กลง โดยการรวมวงจรให้เป็นวงจรที่มีจำนวนน้อยที่สุดเพื่อความกะทัดรัด

2. ควรเลือกใช้ไมโครคอนโทรเลอร์ที่มีฟังก์ชั่นการทำงานง่าย และมีโปรแกรมเสริมสำหรับไมโครคอนโทรลเลอร์ตัวนั้นๆด้วย

3. ควรเลือกใช้วงจรแปลงไฟที่ทนแรงดันได้สูงขึ้นเพื่อสามารถเพิ่มขนาดของแหล่งจ่ายไฟเพื่อให้ไฟในระบบเพียงพอต่อการใช้งานมากขึ้น

4. ควรเลือกใช้สวิตช์ที่ดีกว่าเดิมมีศักยภาพสูงกว่านี้เพื่อให้ระบบทำงานได้ราบรื่น

เอกสารอ้างอิง

[1] ลักษณะผู้พิการทางการได้ยิน สืบค้นเมื่อ 21 กรกฎาคม 2556

จาก http://disabilities.nfe.go.th/popup.php?name=knowledge&file=p_readknowledge&id=32

[2] ไมโครคอนโทรลเลอร์หมายเลข ATmega328P สืบค้นเมื่อ 18 กรกฎาคม 2556

จาก http://www.atmel.com/devices/atmega328p.aspx

[3] เรียลไทม์คล็อก สืบค้นเมื่อ 21 กรกฎาคม 2556

จาก http://en.wikipedia.org/wiki/Real-time_clock

[4] เซ็นเซอร์ตรวจจับการเคลื่อนไหว สืบค้นเมื่อ 21 กรกฎาคม 2556

จาก http://www.thaieasyelec.com/Review-Product-Article/PIR-Motion-Sensor-Getting-Started.html

[5] มอเตอร์กระแสตรง สืบค้นเมื่อ 21 กรกฎาคม 2556

จากhttp://electricalsafety.igetweb.com/

[6] จอแอลซีดี สืบค้นเมื่อ 21 กรกฎาคม 2556

จาก http://www.rmutphysics.com/CHARUD/invention/invention2/Nanodisplay/nanodisplay4.htm

[7] อะแดปเตอร์ สืบค้นเมื่อ 25 พฤษจิกายน 2556

จาก http://atcloud.com/stories/74246

ภาคผนวก ก

โปรแกรมการทำงานของนาฬิกาปลุกคนหูหนวก

```
#include <Wire.h>                          //ฟังก์ชันเรียวไทม์คล็อก
#include <RTClib.h>                        //ฟังก์ชันเรียวไทม์คล็อก
#include <LiquidCrystal.h>                 //ฟังก์ชันหน้าจอแอลซีดี
RTC_DS1307 RTC;
DateTime now;
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
int selectMenu = 0;                        //กำหนดตัวแปร
int lcd_key = 0;
int adc_key_in = 0;
int lastDay = 0;
int lastMonth = 0;
int lastYear = 0;
int lastHour = 0;
int lastMinute = 0;
int movementTimer = 0;
int menuOptions = 3;
int menuOption = 0;
int modeTime = 0;
int modeTimeOption = 2;
int seeAls = 2;
int seeAl = 0;
int alarm1Hours = 0;
int alarm1Minutes = 0;
bool alarm1PM = 0;
bool alarm1Set = 0;
int alarm2Hours = 0;
int alarm2Minutes = 0;
bool alarm2PM = 0;
bool alarm2Set = 0;
int seeAl2s = 2;
int seeAl2 = 0;
int alarm3Hours = 0;
```

```
int alarm3Minutes = 0;
bool alarm3PM = 0;
bool alarm3Set = 0;
int alarm4Hours = 0;
int alarm4Minutes = 0;
bool alarm4PM = 0;
bool alarm4Set = 0;
bool backLightOn = 1;
bool resetClock = false;
const int backLight = 13;
const int pirPin = 2;
int LDRValue = 0;
int light_sensitivity = 500;
#define btnRIGHT  0
#define btnUP     1
#define btnDOWN   2
#define btnLEFT   3
#define btnSELECT 4
#define btnNONE   5
#define motor A3
#define beeper A1
#define longmotor 100
#define shortBeep 100
#define longBeep  500
void setup ()                                    //กำหนดการตั้งค่าระบบ
{
 Serial.begin(57600);
 pinMode(backLight, OUTPUT);
digitalWrite(backLight,LOW);
 pinMode(beeper, OUTPUT);
 digitalWrite(beeper, LOW);
 pinMode(motor, OUTPUT);
```

```
  digitalWrite(motor, LOW);
  pinMode(pirPin, INPUT);
  Wire.begin();
  RTC.begin();
  if (! RTC.isrunning()) {
   Serial.println("RTC is NOT running!")
  }
}
void loop ()                                    //กำหนดการทำงานต่อเนื่อง
{
  now = RTC.now();
  digitalClockDisplay();
  movementTimer++;
  if (movementTimer > 10)
  {
   digitalWrite(13, LOW);
   movementTimer = 0;
  }
  for (int i = 0; i < 10000; i++)
  {
   button_loop();
   int val = analogRead(2);
   if (val > 50)
   {
    digitalWrite(13, HIGH);
    movementTimer = 0;
   }
  }
}
void seeAlarm()
{
 if(selectMenu==0)
```

```
  {
   seeAlarm1();
   delay(500);
  }
  if(selectMenu==1)
  {
   seeAlarm2();
   delay(500);
  }
  else
  {
   digitalClockDisplay();
  }
}
void printDigits(byte digits)
{
  lcd.print(":");
  if(digits < 10)
   lcd.print('0');
  lcd.print(digits,DEC);
}
void button_loop()                                    //ฟังก์ชันการทำงานสวิตช์
{
  int button = read_LCD_buttons();
  if (button == btnSELECT)
  {
   timedBeep(shortBeep,1);
   selectmodeTime();
  }
  if (button == btnLEFT)
  {
   timedBeep(shortBeep,1);
```

```
    seeAlarm();
  }
 }
int read_LCD_buttons()                                    //กำหนดตัวแปรสวิตช์
 {
  adc_key_in = analogRead(0);
  if (adc_key_in > 1000) return btnNONE;
     delay(50);
  if (adc_key_in < 50)   return btnRIGHT;
     delay(50);
  if (adc_key_in < 195)  return btnUP;
     delay (10);
  if (adc_key_in < 380)  return btnDOWN;
     delay (10);
  if (adc_key_in < 555)  return btnLEFT;
     delay (50);
  if (adc_key_in < 790)  return btnSELECT;
    delay (50);
  return btnNONE;
}
void timedCountDown(int secondCount, char countLabel[])      //ฟังก์ชันการทำงานสวิตช์
{
  long seconds = 0;
  long minutes = 0;
  lcdClear();
  lcd.print(countLabel);
  for (int i = secondCount; i >= 0; i--)
  {
    seconds = i;
    minutes = i / 60;
    if (minutes > 0)
    {
```

```
    seconds = seconds - (minutes * 60);
   }
   if (minutes > 0)
   {
    lcd.setCursor(0,1);
    lcd.print(minutes);
    lcd.print(" min ");
   }
   else
   {
    lcd.setCursor(0,1);
   }
   if (seconds < 10) lcd.print("0");
   lcd.print(seconds);
   lcd.print(" sec remaining");
   if (seconds > 0) delay(1000);
   if (read_LCD_buttons() == btnSELECT)
   {
    timerCancelled("Timer");
    i = 0;
    return;
   }
  }
  lcd.setCursor(6,1);
  timedBeep(longBeep,3);
}
int getTimerMinutes(char timerText[], int startNum, int maxCount)
//กำหนดตัวแปลพร้อมทำงาน TimerMinute
{
  int minutes = startNum;
  int button = 0;
  lcdClear();
```

```
lcd.print(timerText);
lcd.setCursor(0,1);
lcd.print(minutes);
while (button != btnSELECT)
{
 button = read_LCD_buttons();
 Serial.println(button);

 if (button == btnUP)
 {
  if (minutes < maxCount)
  {
   timedBeep(shortBeep,1);
   minutes++;
  }
  else
  {
   timedBeep(shortBeep,2);
  }
 }
 if (button == btnDOWN)
 {
  if (minutes > 0)
   timedBeep(shortBeep,1);
   minutes--;
  }
  else
  {
   timedBeep(shortBeep,2);
  }
 }
 if (button == btnRIGHT)
```

```
    {
      timedBeep(shortBeep,1);
      return minutes;
    }
    lcd.setCursor(0,1);
    lcd.print(minutes);
    lcd.print("   ");
  }
  return 0;
}
void timedBeep(int beepTime, int beepCount)              //ฟังก์ชันหน่วงเวลาพร้อมเสียง
{
  for (int i = 0; i < beepCount; i ++)
  {
    digitalWrite(beeper, HIGH);
    delay(beepTime);
    digitalWrite(beeper, LOW);
    delay(beepTime);
  }
}
void timedMotor(int motorTime, int motorCount) //ฟังก์ชันหน่วงเวลาพร้อมทำงานมอเตอร์
{
  for (int i = 0; i < motorCount; i ++)
  {
    digitalWrite(motor, HIGH);
    delay(motorTime);
    digitalWrite(motor, LOW);
    delay(motorTime);
    digitalWrite(beeper, HIGH);
    delay(motorTime);
    digitalWrite(beeper, LOW);
    delay(motorTime);
```

```
  }
}
void lcdClear(){                                    //ฟังก์ชันเคลียร์หน้าจอแสดงผล
  resetClock = true;
  lcd.clear();
  lcd.begin(16,2);
  lcd.setCursor(0,0);
}
void timerCancelled(char message[])                 //ฟังก์ชันยกเลิกการทำงานเวลา
{
  lcdClear();
  lcd.print(message);
  lcd.print(" Cancelled");
  timedBeep(shortBeep,3);
}
void selectmodeTime()
{
  int button = 0;
 menuOption = 1;
  lcdClear();
  lcd.print("ModeTime 12hr");
  while (menuOption <= 2)
  {
    button = read_LCD_buttons();
    if (button == btnSELECT)
    {
      timedBeep(shortBeep,1);
      menuOption++;
      if (menuOption == 2)
      {
        lcdClear();
```

```
      lcd.print("ModeTime 24hr");
     }
    }
    if (button == btnRIGHT)
    {
     if (menuOption == 1)
     {
      timedBeep(shortBeep,1);
        selectMenu1();
        selectMenu = 0;
      return;
     }
     if (menuOption == 2)
     {
      timedBeep(shortBeep,1);
      selectMenu2();
      selectMenu = 1;
      return;
     }
    }
  }
  //menuOption = 1;
}
void digitalClockDisplay()                                    //ฟังก์ชันแสดงผลหน้าจอ
{
   if(selectMenu == 0)
 {
  bool clockPM = 0;
 if (now.day() != lastDay || resetClock == true)
 {
  lcd.begin(16,2);
  lcd.setCursor(3,0);
```

```
 if(now.day() < 10)
  lcd.print('0');
 lcd.print(now.day(), DEC);
 lcd.print("/");
    if(now.month() < 10)
  lcd.print('0');
 lcd.print(now.month(), DEC);
 lcd.print("/");
 int thisYear = now.year();
 lcd.print(thisYear, DEC);
}
if (now.minute() != lastMinute || resetClock == true)
{
 if(now.hour() < 10)
  lcd.setCursor(5,1);
  lcd.setCursor(4,1);
 if(now.hour() > 11)
 {
  if (now.hour() == 12)
  {
   lcd.print(now.hour(), DEC);
  }
  else{
   lcd.print(now.hour()-12, DEC);
  }
  printDigits(now.minute());
  clockPM = true;
  lcd.print(" PM ");
  if (alarm1Set)
   lcd.print("*");
  if (alarm2Set)
```

```
      lcd.print("#");
    }
    else
    {
     lcd.print(now.hour(), DEC);
     printDigits(now.minute());
     clockPM = false;
     lcd.print(" AM ");
    if (alarm1Set)
      lcd.print("*");
     if (alarm2Set)
      lcd.print("#");
    }
  }
}
if(selectMenu == 1)
  {
 if (now.day() != lastDay || resetClock == true)
 {
  lcd.begin(16,2);
  lcd.setCursor(3,0);
  if(now.day() < 10)
    lcd.print('0');
  lcd.print(now.day(), DEC);
  lcd.print("/");
  if(now.month() < 10)
    lcd.print('0');
  lcd.print(now.month(), DEC);
  lcd.print("/");
  int thisYear = now.year();
  lcd.print(thisYear, DEC);
 }
```

```
if (now.minute() != lastMinute || resetClock == true)
{
 if(now.hour() < 10)
  lcd.setCursor(5,1);
  lcd.setCursor(5,1);
 if(now.hour() > 23)
 {
  if (now.hour() == 24)
  {
   lcd.print(now.hour(), DEC);
  }
  else{
   lcd.print(now.hour()-24, DEC);
  }
  printDigits(now.minute());
  if (alarm3Set)
   lcd.print("*");
  if (alarm4Set)
   lcd.print("#");
 }
 else
 {
  lcd.print(now.hour(), DEC);
  printDigits(now.minute());
 if (alarm3Set)
   lcd.print("*");
 if (alarm4Set)
   lcd.print("#");
 }
}
}
```

```
resetClock = false;
 lastDay = now.day();
 lastMonth = now.month();
 lastYear = now.year();
 lastHour = now.hour();
 lastMinute = now.minute();
 if (alarm1Set)
 {
  if (alarm1Hours == lastHour && alarm1Minutes == lastMinute)
  {
   setOffAlarm1();
  }
 }
 if (alarm2Set)
 {
  if (alarm2Hours == lastHour && alarm2Minutes == lastMinute)
  {
   setOffAlarm2();
  }
 }
 if (alarm3Set)
 {
  if (alarm3Hours == lastHour && alarm3Minutes == lastMinute)
  {
   setOffAlarm3();
  }
 }
 if (alarm4Set)
 {
  if (alarm4Hours == lastHour && alarm4Minutes == lastMinute)
  {
   setOffAlarm4();
```

```
    }
   }
  }
```

```
////////////////////////////////////////////ฟังก์ชันการทำงานในระบบเวลา AM/PM////////////////////////////////
void selectMenu1()                                          //ฟังก์ชันเลือกเมนูการทำงาน
{
 int button = 0;
 menuOption = 1;
 lcdClear();
 lcd.print("Set Date Time");
 while (menuOption <= menuOptions)
 {
  button = read_LCD_buttons();
   if (button == btnSELECT)
   {
    timedBeep(shortBeep,1);
    menuOption++;
    if (menuOption == 2)
    {
     lcdClear();
     lcd.print("Set/Clear Alarm1");
    }
    if (menuOption == 3)
    {
     lcdClear();
     lcd.print("Set/Clear Alarm2");
    }
   }
  if(button == btnLEFT)
  {
     timerCancelled("");
```

```
    return;
  }
  if (button == btnRIGHT)
  {
   if (menuOption == 3)
   {
    timedBeep(shortBeep,1);
    if (alarm2Set)
    {
     clearAlarm2();
    }
    else
    {
     setAlarm2();
    }
    return;
   }
   if (menuOption == 2)
   {
    timedBeep(shortBeep,1);
    if (alarm1Set)
    {
     clearAlarm1();
    }
    else
    {
     setAlarm1();
    }
    return;
   }
   if (menuOption == 1)
   {
```

```
      timedBeep(shortBeep,1);
      setDateTime1();
      return;
     }
    }
   }
}
void setAlarm1()                                    //ฟังก์ชันนาฬิกาปลุกตัวที่ 1
{
 int button = 0;
 char *ampm = "AM";
 alarm1Hours = getTimerMinutes("Set Alarm1 Hour", alarm1Hours, 12);
 if (alarm1Hours > 0 && alarm1Hours < 13)
 {
  alarm1Minutes = getTimerMinutes("Set Alarm1 Minutes", alarm1Minutes, 59);
  if (alarm1Minutes < 60)
  {
   lcdClear();
   lcd.print("Toggle AM/PM");
   lcd.setCursor(0,1);
   lcd.print(alarm1Hours);
   lcd.print(":");
   if (alarm1Minutes < 10)
    lcd.print("0");
   lcd.print(alarm1Minutes);
   lcd.setCursor(6,1);
   lcd.print(ampm);
   button = 5;
   while (button != btnSELECT && button != btnRIGHT)
   {
    button = read_LCD_buttons();
    if (button == btnUP)
```

```
    {
     timedBeep(shortBeep,1);
     if (ampm == "AM")
     {
      ampm = "PM";
     }
     else
     {
      ampm = "AM";
     }
     lcd.setCursor(6,1);
     lcd.print(ampm);
    }
   }
   if (button == btnRIGHT)
   {
    timedBeep(shortBeep,1);
    alarm1Set = true;
    if (ampm == "PM") alarm1Hours += 12;
    lcd.setCursor(0,0);
    lcd.print("   Alarm1 Set ");
    delay(1000);
    return;
   }
   else
   {
    timerCancelled("Alarm1");
    return;
   }
  }
 }
```

```
}
void setAlarm2()                                    //ฟังก์ชันนาฬิกาปลุกตัวที่ 2
{
 int button = 0;
 char *ampm = "AM";
 alarm2Hours = getTimerMinutes("Set Alarm2 Hour", alarm2Hours, 12);
 if (alarm2Hours > 0 && alarm2Hours < 13)
 {
  alarm2Minutes = getTimerMinutes("Set Alarm2 Minutes", alarm2Minutes, 59);
  if (alarm2Minutes < 60)
  {
   lcdClear();
   lcd.print("Toggle AM/PM");
   lcd.setCursor(0,1);
   lcd.print(alarm2Hours);
   lcd.print(":");
   if (alarm2Minutes < 10)
    lcd.print("0");
   lcd.print(alarm2Minutes);
   lcd.setCursor(6,1);
   lcd.print(ampm);
   button = 5;
   while (button != btnSELECT && button != btnRIGHT)
   {
    button = read_LCD_buttons();
    if (button == btnUP)
    {
     timedBeep(shortBeep,1);
     if (ampm == "AM")
     {
      ampm = "PM";
     }
```

```
        else
        {
         ampm = "AM";
        }
        lcd.setCursor(6,1);
        lcd.print(ampm);
      }
    }
    if (button == btnRIGHT)
    {
     timedBeep(shortBeep,1);
     alarm2Set = true;
     if (ampm == "PM") alarm2Hours += 12;
     lcd.setCursor(0,0);
     lcd.print("   Alarm2 Set ");
     delay(1000);
     return;
    }
    else
    {
     timerCancelled("Alarm2");
     return;
    }
   }
  }

}
void clearAlarm1()                                    //ฟังก์ชันยกเลิกนาฬิกาปลุกตัวที่ 1
{
 int button = 0;
 bool clearIt = true;
 char *ampm = "AM";
```

```
lcdClear();
lcd.print("Alarm 1 Set For");
lcd.setCursor(0,1);
lcd.print(alarm1Hours);
lcd.print(":");
lcd.print(alarm1Minutes);
lcd.print(" ");
if (alarm1PM == 1)
 ampm = "PM";
lcd.print(ampm);
delay(2000);
lcdClear();
lcd.print("Clear Alarm 1?");
lcd.setCursor(0,1);
lcd.print("Yes");
while (button != btnSELECT)
{
 button = read_LCD_buttons();
 if (button == btnUP)
 {
  timedBeep(shortBeep,1);
  clearIt = !clearIt;
 }
 if (button == btnDOWN)
 {
  timedBeep(shortBeep,1);
  clearIt = !clearIt;
 }
 if (button == btnRIGHT)
 {
  timedBeep(shortBeep,1);
```

```
    alarm1Set = !clearIt;
    if (clearIt)
    {
     lcdClear();
     timedBeep(shortBeep,2);
     lcd.print("Alarm 1 Cleared!");
     delay(2000);
    }
    return;
   }
   lcd.setCursor(0,1);
   if (clearIt)
   {
    lcd.print("Yes");
   }
   else{
    lcd.print("No ");
   }
  }
}
void clearAlarm2()                          //ฟังก์ชันยกเลิกนาฬิกาปลุกตัวที่ 2
{
 int button = 0;
 bool clearIt = true;
 char *ampm = "AM";
 lcdClear();
 lcd.print("Alarm 2 Set For");
 lcd.setCursor(0,1);
 lcd.print(alarm2Hours);
 lcd.print(":");
 lcd.print(alarm2Minutes);
```

```
lcd.print(" ");
if (alarm2PM == 1)
 ampm = "PM";
lcd.print(ampm);
delay(2000);
lcdClear();
lcd.print("Clear Alarm 2?");
lcd.setCursor(0,1);
lcd.print("Yes");
while (button != btnSELECT)
{
 button = read_LCD_buttons();
 if (button == btnUP)
 {
  timedBeep(shortBeep,1);
  clearIt = !clearIt;
 }
 if (button == btnDOWN)
 {
  timedBeep(shortBeep,1);
  clearIt = !clearIt;
 }
 if (button == btnRIGHT)
 {
  timedBeep(shortBeep,1);
  alarm2Set = !clearIt;
  if (clearIt)
  {
   lcdClear();
   timedBeep(shortBeep,2);
   lcd.print("Alarm 2 Cleared!");
   delay(2000);
```

```
    }
    return;
   }
   lcd.setCursor(0,1);
   if (clearIt)
   {
    lcd.print("Yes");
   }
   else{
    lcd.print("No ");
   }
  }
}
void setDateTime1()                                    //ฟังก์ชันตั้งเวลาการทำงานของเครื่อง
{
 int button = 0;
 char *ampm = "AM";
 int setMonth = getTimerMinutes("Set Month", lastMonth, 12);
 if (setMonth > 0 && setMonth < 13)
 {
  int setDay = getTimerMinutes("Set Day", lastDay, 31);
  if (setDay > 0 && setDay < 32)
  {
   int setYear = getTimerMinutes("Set Year", lastYear, 2999);
   if (setYear > 2000 && setYear < 3000)
   {
    int thisHour = lastHour;
    if (thisHour > 12)
    {
     thisHour -= 12;
     ampm = "PM";
    }
```

```
int setHour = getTimerMinutes("Set Hour", thisHour, 12);
if (setHour > 0 && setHour < 13)
{
 int setMinute = getTimerMinutes("Set Minute", lastMinute, 59);
 if (setMinute < 60)
 {
  lcdClear();
  lcd.print("Toggle AM/PM");
  lcd.setCursor(0,1);
  lcd.print(setHour);
  lcd.print(":");
  if (setMinute < 10)
   lcd.print("0");
  lcd.print(setMinute);
  lcd.setCursor(6,1);
  lcd.print(ampm);
  button = 5;
  while (button != btnRIGHT&&button != btnLEFT&&button != btnDOWN&&button !=
btnSELECT)
  {
   button = read_LCD_buttons();
   if (button == btnUP)
   {
    timedBeep(shortBeep,1);
    if (ampm == "AM")
    {
     ampm = "PM";
    }
    else
    {
     ampm = "AM";
```

```
        }
        lcd.setCursor(6,1);
        lcd.print(ampm);
       }
      }
      if (button == btnRIGHT)
      {
       timedBeep(shortBeep,1);
       if (ampm == "PM")
        setHour = setHour + 12;
       RTC.adjust(DateTime(setYear,setMonth,setDay,setHour,setMinute));
       lcd.setCursor(0,0);
       lcd.print("Saving...    ");
       delay(1000);
       return;
      }
     }
     else
     {
      timerCancelled("");
     }
    }
    else
    {
     timerCancelled("");
    }
   }
   else
   {
    timerCancelled("");
   }
  }
```

```
    else
    {
     timerCancelled("");
    }
   }
   else
   {
    timerCancelled("");
   }
}
void setOffAlarm1()                          //ฟังก์ชันปลุกนาฬิกาปลุกตัวที่ 1
{
 int button = 0;
 int i = 0;
 Serial.println(i);
 digitalWrite(backLight, HIGH);
 while (button != btnLEFT)
 {
  button = read_LCD_buttons();
  lcdClear();
  i++;
  if (i > 50)
  {
   lcdClear();
   lcd.print("Alert Alert");
   lcd.setCursor(0,1);
   lcd.print("    Alert Alert");
   i = 0;
   timedMotor(longmotor,3);
  }
 }
 timerCancelled("Alarm");
```

```
 alarm1Set = false;
}
void setOffAlarm2()                                    //ฟังก์ชันปลุกนาฬิกาปลุกตัวที่ 2
{
 int button = 0;
 int i = 0;
 Serial.println(i);
 digitalWrite(backLight, HIGH);
 while (button != btnLEFT)
 {
  button = read_LCD_buttons();
  lcdClear();
  i++;
  if (i > 50)
  {
   lcdClear();
   lcd.print("Alert Alert");
   lcd.setCursor(0,1);
   lcd.print("    Alert Alert");
   i = 0;
   timedMotor(longmotor,3);
  }
 }
 timerCancelled("Alarm");
 alarm2Set = false;
}
void seeAlarm1()                    //ฟังก์ชันดูการตั้งเวลาปลุกในโหมดระบบเวลา AM/PM
{
 int button = 0;
 seeAl =1;
 char *ampm = "AM";
while (seeAl <= 2)
```

```
 {
  if (seeAl == 1)
  {
   viewAlarm1();
  }
  if (seeAl == 2)
  {
   viewAlarm2();
  }
  return;
 }
}
void viewAlarm1()                                    //ฟังก์ชันดูการตั้งเวลาปลุกนาฬิกาปลุกตัวที่ 1
{
 char *ampm = "AM";
  lcdClear();
  lcd.print("Alarm 1 Set at ");
  lcd.setCursor(0,1);
  delay (500);
   if(alarm1Set)
   {
     lcd.print(alarm1Hours);
     lcd.print(":");
     if (alarm1Minutes < 10)
     lcd.print("0");
     lcd.print(alarm1Minutes);
     lcd.print(" ");
     if (alarm1PM == 1)
     ampm = "PM";
     lcd.print(ampm);
     lcd.setCursor(5,1);
     delay (1000);
```

```
    seeAl++;
   }
   else
   {
    lcd.print("None Set");
    lcd.setCursor(4,1);
    delay(1000);
    seeAl++;
   }
  }
  void viewAlarm2()                    //ฟังก์ชันดูการตั้งเวลาปลุกนาฬิกาปลุกตัวที่ 2
{
  char *ampm = "AM";
  lcdClear();
  lcd.print("Alarm 2 Set at ");
  lcd.setCursor(0,1);
  delay (500);
   if(alarm2Set)
   {
    lcd.print(alarm2Hours);
    lcd.print(":");
    if (alarm2Minutes < 10)
    lcd.print("0");
    lcd.print(alarm2Minutes);
    lcd.print(" ");
    if (alarm2PM == 1)
    ampm = "PM";
    lcd.print(ampm);
    lcd.setCursor(5,1);
    delay(1000);
    seeAl++;
   }
```

```
  else
  {
    lcd.print("None Set");
    lcd.setCursor(4,1);
    delay(1000);
    seeAl++;
  }
 }

////////////////////////////////////////////การทำงานในโหมดเวลา 24 ชั่วโมง////////////////////////////////////////////
  void selectMenu2()                                    //ฟังก์ชันเลือกเมนูการทำงาน
{
 int button = 0;
 menuOption = 1;
 lcdClear();
 lcd.print("Set Date Time");
 while (menuOption <= menuOptions)
 {
  button = read_LCD_buttons();
  if (button == btnSELECT)
  {
   timedBeep(shortBeep,1);
   menuOption++;
   if (menuOption == 2)
   {
    lcdClear();
    lcd.print("Set/Clear Alarm1");
   }
   if (menuOption == 3)
   {
    lcdClear();
    lcd.print("Set/Clear Alarm2");
```

```
  }
 }
 if(button == btnLEFT)
 {
   timerCancelled("");
   return;
 }
 if (button == btnRIGHT)
 {
  if (menuOption == 3)
  {
   timedBeep(shortBeep,1);
   if (alarm4Set)
   {
    clearAlarm4();
   }
   else
   {
    setAlarm4();
   }
   return;
  }
  if (menuOption == 2)
  {
   timedBeep(shortBeep,1);
   if (alarm3Set)
   {
    clearAlarm3();
   }
   else
   {
    setAlarm3();
```

```
      }
      return;
     }
     if (menuOption == 1)
     {
      timedBeep(shortBeep,1);
      setDateTime2();
      return;
     }
    }
   }
  }
void setAlarm3()                          //ฟังก์ชันนาฬิกาปลุกตัวที่ 1 ในระบบเวลา 24 ชั่วโมง
{
 int button = 0;
 alarm3Hours = getTimerMinutes("Set Alarm1 Hour", alarm3Hours, 24);
 if (alarm3Hours > 0 && alarm4Hours < 25)
 {
  alarm3Minutes = getTimerMinutes("Set Alarm1 Minutes", alarm3Minutes, 59);
 if (button == btnRIGHT)
    {
     timedBeep(shortBeep,1);
     alarm3Set = true;
     delay(1000);
     lcdClear();
     lcd.setCursor(0,0);
     lcd.print("   Alarm1 Set ");
     delay(1000);
     return;
    }
    else
    {
```

```
      timerCancelled("Alarm1");
      return;
     }
    }
}
void setAlarm4()
{
 int button = 0;
 alarm4Hours = getTimerMinutes("Set Alarm2 Hour", alarm4Hours, 24);
 if (alarm4Hours > 0 && alarm4Hours < 25)
 {
  alarm4Minutes = getTimerMinutes("Set Alarm2 Minutes", alarm4Minutes, 59);
 if (button == btnRIGHT)
    {
     timedBeep(shortBeep,1);
     alarm4Set = true;
     delay(1000);
     lcdClear();
     lcd.setCursor(0,0);
     lcd.print("   Alarm2 Set ");
     delay(1000);
     return;
    }
    else
    {
     timerCancelled("Alarm2");
     return;
    }
   }
}
void clearAlarm3()                          //ฟังก์ชันยกเลิกการปลุกนาฬิกาปลุกตัวที่ 1
{
```

```
int button = 0;
bool clearIt = true;
char *ampm = "AM";
lcdClear();
lcd.print("Alarm 1 Set For");
lcd.setCursor(0,1);
lcd.print(alarm3Hours);
lcd.print(":");
lcd.print(alarm3Minutes);
lcd.print(" ");
if (alarm1PM == 1)
 ampm = "PM";
lcd.print(ampm);
delay(2000);
lcdClear();
lcd.print("Clear Alarm 1?");
lcd.setCursor(0,1);
lcd.print("Yes");
while (button != btnSELECT)
{
 button = read_LCD_buttons();
 if (button == btnUP)
 {
  timedBeep(shortBeep,1);
  clearIt = !clearIt;
 }
 if (button == btnDOWN)
 {
  timedBeep(shortBeep,1);
  clearIt = !clearIt;
 }
 if (button == btnRIGHT)
```

```
    {
     timedBeep(shortBeep,1);
     alarm3Set = !clearIt;
     if (clearIt)
     {
      lcdClear();
      timedBeep(shortBeep,2);
      lcd.print("Alarm 1 Cleared!");
      delay(2000);
     }
     return;
    }
    lcd.setCursor(0,1);
    if (clearIt)
    {
     lcd.print("Yes");
    }
    else{
     lcd.print("No ");
    }
  }
}
void clearAlarm4()                          //ฟังก์ชันยกเลิกการปลุกนาฬิกาปลุกตัวที่ 2
{
{
 int button = 0;
 bool clearIt = true;
 char *ampm = "AM";
 lcdClear();
 lcd.print("Alarm 2 Set For");
 lcd.setCursor(0,1);
 lcd.print(alarm4Hours);
```

```
lcd.print(":");
lcd.print(alarm4Minutes);
lcd.print(" ");
if (alarm4PM == 1)
  ampm = "PM";
lcd.print(ampm);
delay(2000);
lcdClear();
lcd.print("Clear Alarm 2?");
lcd.setCursor(0,1);
lcd.print("Yes");
while (button != btnSELECT)
{
 button = read_LCD_buttons();
 if (button == btnUP)
 {
  timedBeep(shortBeep,1);
  clearIt = !clearIt;
 }
 if (button == btnDOWN)
 {
  timedBeep(shortBeep,1);
  clearIt = !clearIt;
 }
 if (button == btnRIGHT)
 {
  timedBeep(shortBeep,1);
  alarm4Set = !clearIt;
  if (clearIt)
  {
   lcdClear();
   timedBeep(shortBeep,2);
```

```
      lcd.print("Alarm 2 Cleared!");
      delay(2000);
     }
     return;
    }
    lcd.setCursor(0,1);
    if (clearIt)
    {
     lcd.print("Yes");
    }
    else{
     lcd.print("No ");
    }
   }
}
void setDateTime2()                    //ฟังก์ชันตั้งเวลาตัวเครื่องในโหมดระบบเวลา 24 ชั่วโมง
{
 int button = 0;
 int setMonth = getTimerMinutes("Set Month", lastMonth, 12);
 if (setMonth > 0 && setMonth < 13)
 {
  int setDay = getTimerMinutes("Set Day", lastDay, 31);
  if (setDay > 0 && setDay < 32)
  {
   int setYear = getTimerMinutes("Set Year", lastYear, 2999);
   if (setYear > 2000 && setYear < 3000)
   {
    int thisHour = lastHour;
    int setHour = getTimerMinutes("Set Hour", thisHour, 24);
    if (setHour > 0 && setHour < 25)
    {
     int setMinute = getTimerMinutes("Set Minute", lastMinute, 59);
```

```
      if (button == btnRIGHT)
      {
       timedBeep(shortBeep,1);
       RTC.adjust(DateTime(setYear,setMonth,setDay,setHour,setMinute));
       lcd.setCursor(0,0);
       lcd.print("Saving...    ");
       delay(1000);
       return;
      }
      else
      {
       timerCancelled("");
       return;
      }
     }
     else
     {
      timerCancelled("");
     }
    }
    else
    {
     timerCancelled("");
    }
   }
   else
   {
    timerCancelled("");
   }
  }
 }
 else
```

```
    timerCancelled("");
   }
  }
  void setOffAlarm3()                          //ฟังก์ชันปลุกนาฬิกาปลุกตัวที่ 1
{
  int button = 0;
  int i = 0;
  Serial.println(i);
  digitalWrite(backLight, HIGH);
  while (button != btnLEFT)
  {
   button = read_LCD_buttons();
   lcdClear();
   i++;
   if (i > 50)
   {
    lcdClear();
    lcd.print("Alert Alert");
    lcd.setCursor(0,1);
    lcd.print("    Alert Alert");
    i = 0;
   timedMotor(longmotor,3);
   }
  }
  timerCancelled("Alarm");
  alarm3Set = false;
}
void setOffAlarm4()                            //ฟังก์ชันปลุกนาฬิกาปลุกตัวที่ 2
{
  int button = 0;
  int i = 0;
  Serial.println(i);
```

```
digitalWrite(backLight, HIGH);
while (button != btnLEFT)
{
 button = read_LCD_buttons();
 lcdClear();
 i++;
 if (i > 50)
 {
  lcdClear();
  lcd.print("Alert Alert");
  lcd.setCursor(0,1);
  lcd.print("    Alert Alert");
  i = 0;
  timedMotor(longmotor,3);
 }
}
timerCancelled("Alarm");
alarm4Set = false;
}
void seeAlarm2()                    //ฟังก์ชันดูการตั้งปลุกนาฬิกาปลุกในโหมดระบบเวลา 24 ชั่วโมง
{
 int button = 0;
 seeAl2 =1;
while (seeAl2 <= seeAl2s)
{
 if (seeAl2 == 1)
 {
  viewAlarm3();
 }
 if (seeAl2 == 2)
 {
  viewAlarm4();
```

```
  }
 }
}
void viewAlarm3()                          //ฟังก์ชันดูการตั้งปลุกนาฬิกาปลุกตัวที่ 1
{
   lcdClear();
   lcd.print("Alarm 1 Set at ");
   lcd.setCursor(0,1);
   delay (500);
    if(alarm3Set)
    {
      lcd.print(alarm3Hours);
      lcd.print(":");
      if (alarm3Minutes < 10)
      lcd.print("0");
      lcd.print(alarm3Minutes);
      lcd.print(" ");
      lcd.setCursor(5,1);
      delay (1000);
      seeAl2++;
    }
    else
    {
      lcd.print("None Set");
      lcd.setCursor(4,1);
      delay(1000);
      seeAl2++;
    }
  }
  void viewAlarm4()                        //ฟังก์ชันดูการตั้งปลุกนาฬิกาปลุกตัวที่ 2
{
```

```
lcdClear();
lcd.print("Alarm 2 Set at ");
lcd.setCursor(0,1);
delay (500);
 if(alarm4Set)
 {
   lcd.print(alarm4Hours);
   lcd.print(":");
   if (alarm4Minutes < 10)
   lcd.print("0");
   lcd.print(alarm4Minutes);
   lcd.print(" ");
   lcd.setCursor(5,1);
   delay(1000);
   seeAl2++;
 }
 else
{
   lcd.print("None Set");
   lcd.setCursor(4,1);
   delay(1000);
   secAl2++;
}
}
```

ภาคผนวก ข

คู่มือการทำงานนาฬิกาปลุกคนหูหนวกด้วยระบบไมโครคอนโทรลเลอร์

คู่มือการใช้งาน

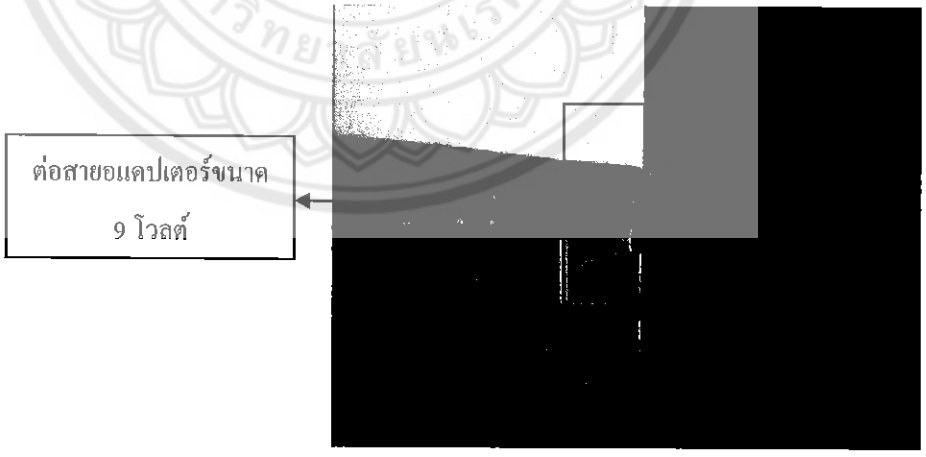
นาฬิกาปลุกคนหูหนวก

คู่มือฉบับนี้จัดทำขึ้นเพื่ออธิบายการใช้งานและวิธีการต่อวงจรไฟเลี้ยงที่ใช้ในกาทำงานของนาฬิกาปลุกคนหูหนวก ซึ่งโหมดเวลาการใช้งานสามารถใช้ได้ 2 โหมด คือ โหมด AM/PM และโหมด 24 ชั่วโมง โดยมีรายละเอียดการใช้งานและวิธีการต่อวงจรไฟเลี้ยงดังต่อไปนี้

1. วิธีการต่อวงจรไฟเลี้ยง
2. รายละเอียดของปุ่มสถานะคำสั่งต่างๆ
3. วิธีการตั้งวัน/เดือน/ปีและเวลาจริง
4. วิธีการตั้งปลุก

1. วิธีการต่อวงจรไฟเลี้ยง

การทำงานของนาฬิกาปลุกคนหูหนวกจะต้องมีไฟเลี้ยงตลอดเวลา เพื่อให้หน้าจอแสดงผลแอลซีดีทำงานตลอด โดยมีการต่อดังนี้

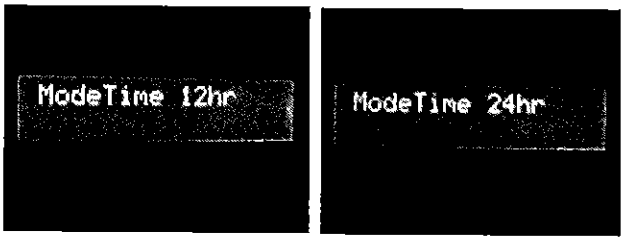


2. รายละเอียดของปุ่มสถานะคำสั่งต่างๆ

ลักษณะการใช้งานปุ่มกดต่างๆ โดยมีการติดตั้งดังรูป

กดปุ่ม (ตรวจสอบเวลาปลุก) การเลือกใช้คำสั่ง (ตรวจสอบเวลาปลุก) ระบบจะทำการส่งข้อมูลชุดคำสั่งไปยังไมโครคอนโทรลเลอร์ เพื่อสั่งการให้หน้าจอแสดงผลแอลซีดีจะแสดงสถานะ (ตรวจสอบเวลาปลุก)

กดปุ่ม (เมนู) การเลือกใช้คำสั่ง (เมนู) ระบบจะทำการส่งข้อมูลชุดคำสั่งไปยังไมโครคอนโทรลเลอร์ เพื่อสั่งการให้หน้าจอแสดงผลแอลซีดีจะแสดงสถานะ (เมนู) โดยจะมีเมนูอยู่ 2 เมนูคือ AM/PM (Mode Time 12 Hour) และ 24 ชั่วโมง (Mode Time 24 Hour)



กดปุ่ม (เพิ่มค่า)	การเลือกใช้คำสั่ง (เพิ่มค่า) ระบบจะทำการส่งข้อมูลชุดคำสั่งไปยังไมโครคอนโทรลเลอร์ เพื่อสั่งการให้เพิ่มค่าตัวเลขโดยจะแสดงผลผ่านหน้าจอแสดงผลแอลซีดี
กดปุ่ม (ลดค่า)	การเลือกใช้คำสั่ง ลดค่า) ระบบจะทำการส่งข้อมูลชุดคำสั่งไปยังไมโครคอนโทรลเลอร์ เพื่อสั่งการให้ลดค่าตัวเลขโดยจะแสดงผลผ่านหน้าจอแสดงผลแอลซีดี
กดปุ่ม (ตกลง)	การเลือกใช้คำสั่ง (ตกลง) ระบบจะทำการส่งข้อมูลชุดคำสั่งไปยังไมโครคอนโทรลเลอร์ เพื่อสั่งการให้ค่าที่ได้ตั้งไว้ได้มีการเก็บข้อมูลไว้และแสดงผลผ่านหน้าจอแสดงผลแอลซีดี

3. วิธีการตั้งวัน/เดือน/ปีและเวลาจริง

1. กดปุ่ม "เมนู" ให้หน้าจอแสดงผลแอลซีดีแสดงข้อความ (Mode Time 12 Hour) และเมื่อกดปุ่ม "เมนู" อีก 1 ครั้งหน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Mode Time 24 Hour) เลือกโหมดที่ต้องการแล้วให้กดปุ่ม "ตกลง" เพื่อเข้าไปยังโหมดนั้น หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Set Date Time) เพื่อให้ผู้ใช้ทราบสถานะ การทำงานเข้าโหมดการตั้งวัน/เดือน/ปีและเวลาจริง

2. กดปุ่ม "ตกลง" เพื่อเข้าไปตั้งเดือน โดยหน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Set Month) และกดปุ่ม "เพิ่มค่า" "ลงค่า" เพื่อเพิ่มหรือลดค่าตัวเลข

3. ตั้งค่าเดือนเสร็จให้กดปุ่ม "ตกลง" หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Set Day) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน เพื่อตั้งวัน แล้วกดปุ่ม "เพิ่มค่า" "ลงค่า" เพื่อเพิ่มหรือลดค่าตัวเลข

4. ตั้งค่าเดือนเสร็จให้กดปุ่ม “ตกลง” หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Set Year) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน เพื่อตั้งปี แล้วกดปุ่ม “เพิ่มค่า” “ลงค่า”เพื่อเพิ่มหรือลดค่าตัวเลข

5. ตั้งค่าปีเสร็จให้กดปุ่ม “ตกลง” หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Set Hour) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน เพื่อตั้งเวลา (ชั่งโมง) แล้วกดปุ่ม “เพิ่มค่า” “ลงค่า”เพื่อเพิ่มหรือลดค่าตัวเลข

6. ตั้งค่าชั่งโมงเสร็จให้กดปุ่ม “ตกลง” หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Set Minute) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน เพื่อตั้งเวลา (นาที) แล้วกดปุ่ม “เพิ่มค่า” “ลงค่า” เพื่อเพิ่มหรือลดค่าตัวเลข

7. ตั้งค่านาทีเสร็จให้กดปุ่ม “ตกลง” หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (ในโหมด AM/PM คือ “Toggle AM/PM”) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน เพื่อตั้งช่วงเวลาก่อนเที่ยง (AM) และหลังเที่ยง (PM) แล้วกดปุ่ม “เพิ่มค่า” เพื่อเลือกช่วงเวลาก่อนเที่ยง (AM) และหลังเที่ยง (PM)

8. ตั้งค่าช่วงเวลาเสร็จให้กดปุ่ม “ตกลง” หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (วัน/เดือน/ปีและเวลา) ตามปกติ

9. ตั้งค่านาทีเสร็จให้กดปุ่ม “ตกลง” หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (ในโหมด 24 ชั่วโมง) เป็นวัน/เดือน/ปีและเวลาปกติ

4. วิธีการตั้งเวลาปลุก

การตั้งเวลาปลุกนี้จะมีการตั้งโหลดละ 2 ครั้งทั้ง AM/PM และ 24 ชั่งโมง โดยแต่ละโหมดจะมีลักษณะวิธีการตั้งที่เหมือนกันแตกต่างตรงที่ AM/PM จะมีคำสั่งเลือกเพิ่มขึ้นมาว่าเป็น AMหรือ PM โดยมีวิธีการตั้งเวลาปลุกดังนี้

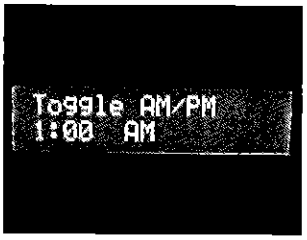
1. กดปุ่ม “เมนู” ให้หน้าจอแสดงผลแอลซีดีแสดงข้อความ (Mode Time 12 Hour) และเมื่อกดปุ่ม “เมนู” อีก 1 ครั้งหน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Mode Time 24 Hour) เลือก

โหมดที่ต้องการแล้วให้กดปุ่ม “ตกลง” เพื่อเข้าไปยังโหมดนั้น หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Set Date Time) จากนั้นกดปุ่ม “เมนู” หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Set /Clear Alarm1) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน

2. กดปุ่ม “ตกลง” จอแสดงผลแอลซีดีจะแสดงข้อความ (Set Alarm1 Hour) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน เพื่อตั้งชั่วโมง แล้วกดปุ่ม “เพิ่มค่า” “ลงค่า” เพื่อเพิ่มหรือลดค่าตัวเลข

3. ตั้งค่าชั่งโมงเสร็จให้กดปุ่ม “ตกลง” จอแสดงผลแอลซีดีจะแสดงข้อความ (Set Alarm1 Minute) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน เพื่อตั้งนาที แล้วกดปุ่ม “เพิ่มค่า” “ลงค่า” เพื่อเพิ่มหรือลดค่าตัวเลข

4. ตั้งค่านาทีเสร็จให้กดปุ่ม “ตกลง” จอแสดงผลแอลซีดีจะแสดงข้อความ (เฉพาะในโหมด 12 ชั่วโมง คือ “Toggle AM/PM”) เพื่อให้ผู้ใช้ทราบสถานะ การทำงาน เพื่อตั้งช่วงเวลาก่อนเที่ยง (Am) และหลังเที่ยง (PM)



5. ตั้งค่านาทีเสร็จให้กดปุ่ม "ตกลง" หน้าจอแสดงผลแอลซีดีจะแสดงข้อความ (Alarm1 Set) แล้วรอสักครู่หน้าจอแสดงผลจะกลับไปสู่หน้าจอปกติทั้ง 2 โหมด (วัน/เดือน/ปีและเวลาจริง) โดยจะมีเครื่องหมายดอกจัน (*) เพื่อให้ผู้ใช้ได้รู้ว่ามีการตั้งเวลาปลุกครั้งที่ 1 ส่วนการตั้งเวลาปลุกครั้งที่ 2 นั้นจะมีเครื่องหมายสี่เหลี่ยม (#) และเมื่อนาฬิกาปลุกทำงานถ้าเราต้องการหยุดการทำงานให้กดปุ่ม "หยุดการปลุก" ด้านบนของตัวเครื่อง

ประวัติผู้ดำเนินโครงงาน

ชื่อ นายณัฐวัตร แย้มศรี

ภูมิลำเนา 29/13หมู่ 5ต.ไร่ขิง อ.สามพราน จ.นครปฐม

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนวัดไร่ขิงวิทยา
- ปัจจุบันกำลังศึกษาระดับปริญญาตรี ชั้นปีที่ 4 สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail :o_y_forever@hotmail.com.th

ชื่อ นายภูวดล แย้มชื่น

ภูมิลำเนา 137/1หมู่ 7ต.สามง่ามอ.สามง่าม จ.พิจิตร

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนสามง่ามชนูปถัมภ์
- ปัจจุบันกำลังศึกษาระดับปริญญาตรี ชั้นปีที่ 4 สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail :ball_puwadol@hotmail.com