สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

เครื่องให้อาหารปลาอัตโนมัติ และสามารถสั่งงานทางโทรศัพท์ได้

AUTOMATIC FISH FEEDING MACHINE AND CONTROLABLE BY TELEPHONE



โดย

นาย ชาญชัย อัศวภูษิตกุล

นาย ณัฐพงษ์ จุลบุตร

ส/ลว.
25486
2547

เลขหมู่.....
เลขทะเบียน 61349
วัน,เดือน,ปี 17 ก.ค. 2549

b 11595413
i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่องให้อาหารปลาอัตโนมัติ และสามารถสั่งงานทางโทรศัพท์ได้

AUTOMATIC FISH FEEDING MACHINE AND CONTROLABLE BY TELEPHONE

โดย

นาย ชาญชัย อัศวภูษิตกุล 44010111

นาย ณัฐพงษ์ จุลบุตร 44010151

อาจารย์ที่ปรึกษา

ผศ.พลผดุง ผดุงกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงงานเรื่อง เครื่องให้อาหารปลาอัตโนมัติ และ สามารถสั่งงานทางโทรศัพท์ได้

AUTOMATIC FISH FEEDING MACHINE AND CONTROLABLE BY TELEPHONE

จัดทำโดย นายชาญชัย อัศวภูษิตกุล รหัส 44010111

นายณัฐพงษ์ จุลบุตร รหัส 44010151

โครงงานนี้ได้รับการตรวจสอบแล้ว พร้อมที่จะทำการสอบได้



.....,

(ผศ.พลผดุง ผดุงกุล)

อาจารย์ที่ปรึกษา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2547

ภาควิชา วิศวกรรมอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง เครื่องให้อาหารปลาอัตโนมัติและสามารถสั่งงานทางโทรศัพท์ได้

ผู้จัดทำ

1. นาย ชาญชัย อัศวภูษิตกุล รหัส 44010111
2. นาย ณัฐพงษ์ จุลบุตร รหัส 44010151



.....(อาจารย์ ที่ปรึกษา)

(ผศ.พลผดุง ผดุงกุล)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่องให้อาหารปลาอัตโนมัติ และ สามารถสั่งงานทางโทรศัพท์ได้

นายชาญชัย อัศวภูษิตกุล
นายณัฐพงษ์ จุลบุตร
ผศ.พลผดุง ผดุงกุล(อาจารย์ที่ปรึกษา)
ปีการศึกษา 2547

บทคัดย่อ

ปริญญานิพนธ์นี้นำเสนอการสร้างเครื่องให้อาหารปลาอัตโนมัติและสั่งงานทางโทรศัพท์ สามารถติดต่อสื่อสารและสั่งงานการทำงานของเครื่องผ่านโทรศัพท์ได้ โดยอาศัยการแปลงสัญญาณ DTMF จากเครื่องโทรศัพท์ส่งให้ไมโครคอนโทรลเลอร์ ในส่วนการทำงานอัตโนมัติที่ตัวเครื่อง ใช้ Keypad ในการติดต่อสื่อสารและสั่งงานกับเครื่อง การแสดงผลใช้ LCD ในการแสดงผลติดต่อกับผู้ใช้งาน และใช้ IC Real Time Clock(RTC) ในการสร้างฐานเวลาจริงส่งให้ไมโครคอนโทรลเลอร์ เครื่องให้อาหารปลาอัตโนมัติมี 2 ช่องทางการให้อาหาร โดยสามารถโปรแกรม จำนวนครั้งต่อวัน เวลา และปริมาณที่จะให้ ตั้งแต่วันเสาร์ถึงวันอาทิตย์ ของทั้ง 2 ช่องทางการให้อาหารได้อย่างอิสระต่อกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

AUTOMATIC FEEDINGFISH MACHINE AND CONTROLABLE BY TELEPHONE

Chanchai Ussawapusitgul

Nuttapong Julabut

Pulpadung Phadungkul(Adviser)

Academic year 2004

Abstract

This thesis present the Automatic Feeding Fish Machine And Controlable By Telephone. It can communicate and work by telephone on principle of DTMF signal from telephone to microcontroler. In automatic mode using keypad to communicate the machine and display to user by LCD. Using IC Real Time Clock(RTC) to establish real time for microcontroler. It can feed in two Channel by programming day to day and define and define the feeding time and quantity in each channel independenly.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

โครงงาน เครื่องให้อาหารปลาอัตโนมัติ และสามารถสั่งงานทางโทรศัพท์นี้จะไม่สามารถสำเร็จลุลวงลงได้เลย ถ้าไม่ได้รับความกรุณาจากท่าน ผศ. พลผดุง ผดุงกุล อาจารย์ที่ปรึกษาที่คอยแนะนำให้คำปรึกษาและความรู้ที่มีประโยชน์ในการโครงงานนี้ อนุญาตให้ใช้เครื่องมือต่างๆ ในห้องของท่านได้อย่างเต็มที่

ขอขอบพระคุณ

ผศ.พลผดุง ผดุงกุล ในฐานะอาจารย์ที่ปรึกษาที่ให้คำแนะนำ และช่วยเหลือในการทำโครงงานด้วยดีเสมอมา

เพื่อนๆ ที่ให้คำแนะนำในด้านเทคนิคต่างๆ

คณะผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เรื่อง	หน้า
กิตติกรรมประกาศ	
บทคัดย่อ	
ABSTRACT	
บทที่ 1 คำนำ	1
1.1 วัตถุประสงค์ของโครงงาน	1
1.2 ขอบเขตของโครงงาน	1
1.3 ความเป็นมาและความสำคัญของโครงงาน	1
บทที่ 2 ทฤษฎี	2
2.1 ส่วนของเครื่องตอบรับและสั่งงานทางโทรศัพท์	2
2.1.1 ส่วนต่อกับคู่สายโทรศัพท์	3
2.1.2 ส่วนประมวลผล	4
2.1.3 ส่วนแสดงผล	4
2.1.4 ไมโครคอนโทรเลอร์	5
2.2 ส่วนของการทำงานอัตโนมัติโดยใช้การตั้งเวลา ซึ่งอาศัยฐานเวลาจริงจาก IC REAL TIME CLOCK	22
2.2.1 การเชื่อมต่อกับคีย์แพดหรือสวิตช์เมตริกซ์ 4x3 จุด กับไมโครคอนโทรลเลอร์	23
2.2.2 การใช้งานโมดูลแสดงผลแบบผลึกเหลว (LCD module)กับ MCS-51	24
2.2.3 ระบบบัส I^2C	31
2.2.4 การใช้งานไอซีสร้างฐานเวลาจริง รีลไทม์คล็อก(RTC)กับ MCS-51	36
บทที่ 3 การศึกษาและการออกแบบสร้างวงจร	43
3.1 การออกแบบเครื่องตอบรับและสั่งงานทางโทรศัพท์	43
3.1.1 วงจรตรวจจับสัญญาณกระดิ่ง และหน่วงเวลา	43
3.1.2 วงจรยกหู/วางหูโทรศัพท์	44
3.1.3 วงจรถอดรหัสความถี่โทรศัพท์	45
3.1.4 ภาคประมวลผล	46

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 การออกแบบส่วนของการทำงานอัตโนมัติโดยใช้
การตั้งเวลาซึ่งอาศัยฐานเวลาจริงจาก IC REAL TIME CLOCK 47
3.3 การออกแบบตัวเครื่องให้อาหาร 49
บทที่ 4 คุณสมบัติและลำดับการใช้งาน 55
4.1 คุณสมบัติและลำดับการใช้งานของภาคสั่งงานทางโทรศัพท์ 56
4.2 คุณสมบัติและลำดับการใช้งานของภาคการทำงานอัตโนมัติ 56
4.3 โฟวชาตการทำงานของเครื่อง 57
บทที่ 5 ผลการทดลอง 69
5.1 กล่าวนำ 69
5.2 ผลการทดลองของส่วนการสั่งงานทางโทรศัพท์ 69
5.3 ผลการทดลองของส่วนการทำงานอัตโนมัติที่ตัวเครื่อง 72
บทที่ 6 สรุปและวิเคราะห์ผลการทดลอง 78
6.1 บทสรุป 78
6.2 วิเคราะห์ผลการทดลอง 78
6.3 แนวทางการพัฒนา 78
บรรณานุกรม
ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

หน้า

- รูปที่ 2.1 ส่วนประกอบผังของเครื่องตอบรับและสั่งงานทางโทรศัพท์ 2
- รูปที่ 2.2 ตำแหน่งของหน่วยความจำทั้งแบบไบต์ และแบบบิต 8
- รูปที่ 2.3 วงจรของสวิตช์แบบต่อเข้ากับไฟเลี้ยงและกราวน์ 22
- รูปที่ 2.4 วงจรของสวิตช์แบบเมตริกซ์หรือคีย์แพด 23
- รูปที่ 2.5 วงจรเชื่อมต่อคีย์แพดเข้ากับไมโครคอนโทรลเลอร์ MCS-51 24
- รูปที่ 2.6 แสดงรูปร่างและการจัดขาของโมดูล LCD 25
- รูปที่ 2.7 การต่อตัวต้านทานพูลอัปบนสายสัญญาณในระบบบัส I^2C 32
- รูปที่ 2.8 วงจรตัวอย่างการต่อไมโครคอนโทรลเลอร์ MCS-51 กับอุปกรณ์ระบบ I^2C 34
- รูปที่ 2.9 การจัดขาของไอซี DS1307 ไอซีสร้างฐานเวลาจริง (RTC) 36
- รูปที่ 2.10 โครงสร้างภายในของ ไอซีรีลไทม์คล็อกเบอร์ DS1307 38
- รูปที่ 2.11 (ก) การจัดสรรหน่วยความจำแรมภายใน 39
  (ข) รายละเอียดของรีจิสเตอร์เก็บค่าเวลาและ
  รีจิสเตอร์ควบคุมของ DS1307 39
- รูปที่ 2.12 รูปแบบของข้อมูลสำหรับติดต่อกับ DS1307 ในโหมดการเขียนข้อมูล 41
- รูปที่ 2.13 รูปแบบของข้อมูลสำหรับติดต่อกับ DS1307 ในโหมดการอ่านข้อมูล 41
- รูปที่ 2.14 แสดงการเชื่อมต่อไมโครคอนโทรลเลอร์ MCS-51 กับไอซีรีลไทม์คล็อก DS1307 42
- รูปที่ 3.1 วงจรตรวจจับสัญญาณกระดิ่ง 43
- รูปที่ 3.2 แผนผังการทำงานของวงจรยกหูโทรศัพท์ 44
- รูปที่ 3.3 วงจรการใช้งานของ MT8870 46
- รูปที่ 3.4 แสดงการต่อวงจรภาคสั่งงานทางโทรศัพท์ และภาคทำงานอัตโนมัติเข้ากับไมโครคอนโทรเลอร์ 48
- รูปที่ 3.5 แสดงรูแบบตัวเครื่องที่ใช้ควบคุมการทำงานและติดต่อกับผู้ใช้งาน 49
- รูปที่ 3.6 แสดงการออกแบบส่วนที่ใช้เป็นช่องทางการให้อาหารที่ 1 50
- รูปที่ 3.7 แสดงส่วนของ มอเตอร์ ชุดเฟือง และลิมิตสวิตช์ที่ใช้งานของช่องทางการให้อาหารที่ 1 50

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- รูปที่ 3.8 แสดงลักษณะของอาหารที่ใช้กับช่องทางการให้อาหารที่ 1 51
- รูปที่ 3.9 แสดงการออกแบบส่วนที่ใช้เป็นช่องทางการให้อาหารที่ 2 52
- รูปที่ 3.10 แสดงส่วนของ มอเตอร์
และลิมิตสวิตช์ที่ใช้งานของช่องทางการให้อาหารที่ 2 52
- รูปที่ 3.11 แสดงลักษณะของอาหารที่ใช้กับช่องทางการให้อาหารที่ 2 53
- รูปที่ 3.12 แสดงลักษะการชื่อมต่อในการใช้ของช่องทางการการให้อาหารทั้งสองกับส่วนควบคุม
การทำงานและติดต่อกับผู้ใช้งาน 54
- รูปที่ 4.1 ตำแหน่งของปุ่มหน้าปัดโทรศัพท์ 55

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

	หน้า
ตารางที่ 2.1 การจำแนกแอดเดรสรีจิสเตอร์	9
ตารางที่ 2.2 การกำหนดค่าของบิตที่อยู่ใน PSW	10
ตารางที่ 2.3 ความสามารถอ้างถึงแบบบิตชิงรีจิสเตอร์	11
ตารางที่ 2.4 ความหมายของโปรแกรม สเตตัสเวิร์ด(PSW)	12
ตารางที่ 2.5 ความหมายของโปรแกรมพาวเวอร์คอนโทรลรีจิสเตอร์(PCON)	13
ตารางที่ 2.6 แสดงความสัมพันธ์ในการทำงานของขา RS,R/WและE ของ LCD	25

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 วัตถุประสงค์ของโครงงาน

วัตถุประสงค์ในการทำปริญญานิพนธ์ฉบับนี้ ผู้เขียนมีวัตถุประสงค์เพื่อสร้างเครื่องให้อาหารปลาอัตโนมัติ ที่สามารถควบคุมเครื่องให้อาหารได้สองแหล่ง โดยที่สามารถตั้งเวลา,จำนวนครั้ง,และปริมาณการให้อาหารที่ต้องการให้ของ 2 แหล่งจ่ายแยกกันอิสระ

การควบคุมเวลา,จำนวนครั้ง,ปริมาณการให้อาหารทำได้ 2 วิธี คือจากที่ตัวเครื่องหรือจากการควบคุมทากโทรศัพท์

เวลาที่ใช้อ้างอิงเป็นค่าฐานเวลาจริง(Real Time) ในขณะที่ไม่มีการตั้งค่าจอ LCD จะแสดง วัน เวลา วันที เดือน และ ปี

1.2 ขอบเขตของโครงงาน

- สามารถตั้งเวลา จำนวนครั้ง ปริมาณการให้อาหารได้ 2 แหล่งจ่ายอาหารแยกจากกันได้อย่างอิสระ ตั้งแต่วันเสาร์ ถึง อาทิตย์
- สามารถรับคำสั่งการควบคุม การให้ หยุดให้ ปริมาณอาหารที่จะให้จากโทรศัพท์ได้
- สามารถ แสดงวัน เวลา วันที่ เดือน ปี ได้ขณะที่ไม่มีการตั้งค่าได้
- สามารถตั้งเวลา วัน วันที่ เดือน ปี ได้จาก Keypad
- มีระบบรักษาความปลอดภัยโดยมีการถาม Admin Password และ Password เพื่อป้องกันผู้ไม่พึงประสงค์มาใช้งาน
- สามารถเปลี่ยนแปลง Password ได้จาก Keypad และ จากทางโทรศัพท์

1.3 ความเป็นมาและความสำคัญของโครงงาน

ในปัจจุบันมีการเลี้ยงปลาเพื่อจำหน่ายกันเป็นจำนวนมากจนทำให้ปลากลายเป็นสัตว์เลี้ยงที่สำคัญชนิดหนึ่ง ซึ่งในการเลี้ยงปลานั้นปัญหาที่สำคัญที่เป็นประเด็นหลักๆ ก็คือ การที่ผู้เลี้ยงให้อาหารปลาในปริมาณที่ไม่เหมาะสม และไม่เป็นเวลามากบ้างน้อยบ้าง หรือหลงลืมการให้อาหาร ส่งผลเสียต่อปลาซึ่งเป็นสัตว์เลี้ยง ทำให้ปลาโตได้ไม่เต็มที่ และไม่แข็งแรงเพราะได้อาหารไม่เพียงพอ

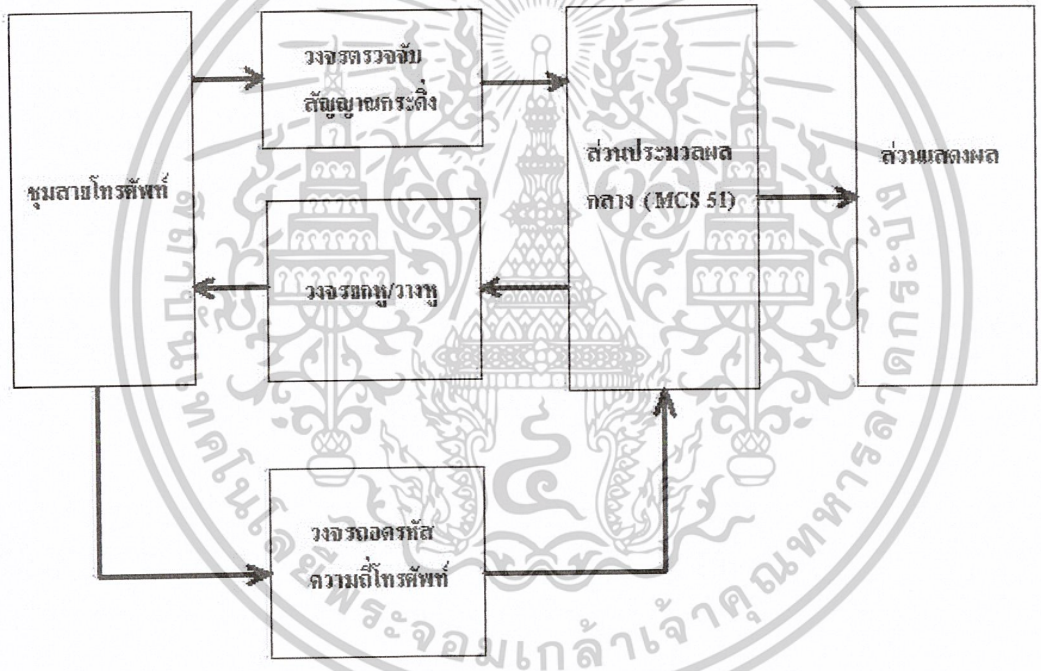
การให้อาหารปลาที่ไม่เป็นเวลา และปริมาณที่ไม่อาจควบคุมได้แน่นอน รวมทั้งผู้เลี้ยงเองอาจไม่มีเวลา หรือต้องไปทำธุระในที่ไกลๆ ดังนั้นจึงได้มีการคิดค้นเครื่องให้อาหารปลาอัตโนมัติที่สามารถสั่งงานทางโทรศัพท์ได้ด้วย ซึ่งเครื่องนี้จะทำงานตามโปรแกรมที่กำหนดโดยที่ผู้ใช้สามารถตั้งจำนวนครั้ง และเวลาการให้อาหารได้เองตามต้องการ ซึ่งเป็นการเพิ่มความสะดวกสบายให้กับผู้เลี้ยงปลาในที่ต่างๆ ได้มากขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎี

2.1 ส่วนของเครื่องตอบรับและสั่งงานทางโทรศัพท์

ในโครงงานนี้ ได้เอาเครื่องโทรศัพท์มาประยุกต์ใช้เป็น เครื่องสั่งงานทางโทรศัพท์ และเครื่องตอบรับโทรศัพท์เพื่อรับการสั่งงานทางโทรศัพท์ซึ่งจะต้องทำตัวเหมือนเครื่องรับโทรศัพท์เครื่องหนึ่ง สามารถติดต่อและเข้าใจสัญญาณต่างๆจากชุมสาย เช่น การรับสัญญาณกระดิ่งจากชุมสาย การยกหูการวางหูของเครื่องสั่งงานทางโทรศัพท์ ซึ่งเครื่องสั่งงานทางโทรศัพท์แบ่งออกเป็น 3 ส่วนใหญ่ๆ ดังรูป 2.1



รูปที่ 2.1 ส่วนประกอบผังของเครื่องตอบรับและสั่งงานทางโทรศัพท์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.1 ส่วนต่อกับคู่สายโทรศัพท์

หน้าที่หลักของส่วนเชื่อต่อกับคู่สายโทรศัพท์คือ การทำให้ส่วนประมวลผลทำงานร่วมกับคู่สายโทรศัพท์ได้ โดยสัญญาณภายในคู่สายโทรศัพท์เป็นสัญญาณอนาลอก (analog signal) และมีความแตกต่างของแรงดันภายในคู่สายโทรศัพท์มาก ได้แก่สัญญาณกระดิ่ง (Tone ringing) สัญญาณแมวกวน (Dial tone) เป็นต้น ในขณะที่สัญญาณของส่วนประมวลผลเป็น สัญญาณลอจิก (Logic signal)

ส่วนเชื่อมต่อกับคู่สายประกอบด้วย วงจรยกหู/วางหู วงจรตรวจจับสัญญาณกระดิ่ง วงจรถอดรหัสสัญญาณเลขหมาย วงจรตอบรับโทรศัพท์ และวงจรสร้างหน่วงเวลา ซึ่งใช้หลักการของโทรศัพท์ดังนี้

1. การยกหู/วางหูของเครื่องโทรศัพท์ ในขณะที่โทรศัพท์วางหูอยู่นั้น แรงดันภายในคู่สายโทรศัพท์มีค่าประมาณ 48 โวลท์ และอิมพิแดนซ์(impedance) ของเครื่องโทรศัพท์มีค่าสูงมาก ส่วนในขณะที่เครื่องโทรศัพท์ยกหูแรงดันคู่สายโทรศัพท์มีค่าประมาณ 6-10 โวลท์ ทั้งนี้เนื่องจากอิมพีแดนซ์ ของโทรศัพท์ขณะยกหูมีค่าประมาณ 600 Ω โดยชุมสายโทรศัพท์จะตรวจการยกหูของเครื่องโทรศัพท์จากการเปลี่ยนแปลงของอิมพีแดนซ์ ของเครื่องโทรศัพท์

2. สัญญาณเสียงพูดในระบบโทรศัพท์ใช้ในช่วงความถี่ 300-3400 Hz (สัญญาณเสียงมนุษย์ได้ยินอยู่ในช่วง 20-20000 Hz) ซึ่งเป็นช่วงความถี่ที่ใช้แล้วมนุษย์สามารถสื่อสารกันด้วยคำพูดที่เข้าใจได้

3. สัญญาณกระดิ่ง (Ring tone) เป็นสัญญาณที่ส่งจากชุมสายโทรศัพท์มายังโทรศัพท์ทางด้านผู้รับ ให้ขับกระดิ่งภายในเครื่องโทรศัพท์ เพื่อบอกให้ฝ่ายรับ ทราบว่ามีการเรียกเข้าโดยมีระดับสัญญาณ 120 โวลท์ ความถี่ 20 Hz โดยส่ง 1 วินาที หยุด 4 วินาที

4. สัญญาณเลขหมายเป็นสัญญาณที่สร้างจากเครื่องโทรศัพท์ เพื่อใช้ส่งเลขหมายของผู้รับ สำหรับสัญญาณของเลขหมายขึ้นอยู่กับชนิดของเครื่องโทรศัพท์และชุมสายโทรศัพท์ที่ใช้คือแบบหมุน (ส่งสัญญาณเป็นพัลส์) และแบบกดปุ่ม (ส่งสัญญาณความถี่คู่) เนื่องจากโทรศัพท์แบบกดปุ่มมีแนวโน้มนำมาใช้แทนโทรศัพท์แบบหมุน ดังนั้นโครงงานชุดนี้จึงพัฒนาให้ใช้กับเครื่องโทรศัพท์ของผู้สั่งงานทางโทรศัพท์เป็นแบบกดปุ่ม ซึ่งมีสัญญาณที่ใช้สั่งเป็นแบบความถี่คู่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.2 ส่วนประมวลผล

ส่วนประมวลผลนี้มีความสำคัญมาก เปรียบเสมือนเป็นสมองของเครื่องสั่งงานทางโทรศัพท์ก็ได้ คือ เป็นส่วนที่ใช้ในการบริหารการทำงานของวงจรภายในเครื่องทำงานทางโทรศัพท์ให้วงจรแต่ละส่วนสามารถรวมกันเป็นระบบใหญ่ ซึ่งจะขอกล่าวถึงการบริหารวงจรของส่วนประมวลผลคร่าวๆ ดังนี้

ในภาวะปกติเครื่องตอบรับและสั่งงานทางโทรศัพท์จะวางหูโทรศัพท์ โดยคู่สายโทรศัพท์ต่อเข้ากับวงจรตรวจจับกระดิ่ง เมื่อมีผู้เรียกเข้ามายังเครื่องโทรศัพท์ส่วนประมวลผลจะส่งสัญญาณไปยังวงจรยกหู/วางหู ให้ทำการยกหูพร้อมทั้งสลับคู่สายโทรศัพท์จากวงจรตรวจจับสัญญาณกระดิ่ง ไปยังวงจรส่วนตอบรับโทรศัพท์ วงจรนี้จะทำงานโดยจะสั่งงานเครื่องใช้ไฟฟ้า

เมื่อคู่สายโทรศัพท์ต่อกับวงจรตอบรับส่วนประมวลผล จะทำให้วงจรตอบรับส่งข้อมูลเสียงแจ้งให้ผู้เรียกทราบว่าไม่มีคนอยู่เพื่อสั่งงานควบคุมอุปกรณ์ไฟฟ้า

การเลือกควบคุมเครื่องใช้ไฟฟ้า ก็จะต้องกดรหัสผ่านเพื่อเข้าสู่ระบบสั่งงานทางโทรศัพท์ ส่วนประมวลผลจะรับข้อมูลเลขหมาย ที่กดโดยผู้สั่งงานทางโทรศัพท์จากการถอดรหัสสัญญาณเลขหมาย

สำหรับส่วนแสดงผล ก็จะได้รับสัญญาณควบคุมเครื่องใช้ไฟฟ้า จากส่วนประมวลผลเช่นกัน สัญญาณที่ใช้ในการติดต่อในส่วนแสดงผล ระหว่างส่วนที่ต่อกับคู่สายโทรศัพท์กับส่วนประมวล เป็นสัญญาณลอจิก คือมีระดับสัญญาณเพียง 2 สถานะ คือสถานะแรงดันสูง กับแรงดันต่ำ

2.1.3 ส่วนแสดงผล

ส่วนแสดงผลนี้มีความสำคัญเทียบเท่ากับส่วนอื่นๆ เพราะจะเป็นตัวแสดงผลเพื่อให้ทราบถึงผลการทดลองที่เกิดขึ้น เปรียบเสมือนกับตัวที่บ่งบอกถึงการทำงานของวงจรทั้งหมด ว่าเป็นไปตามผลที่คาดว่าจะได้รับหรือไม่

จะเห็นได้ว่าเมื่อมีการสั่งงานทางโทรศัพท์ผ่านเข้ามาส่วนประมวลผล ซึ่งเป็นส่วนที่ใช้ในการบริหารการทำงานของวงจรแล้ว โดยส่วนประมวลผลจะรับข้อมูลหมายเลขที่กดโดยผู้สั่งงานทางโทรศัพท์จากวงจรถอดรหัสเลขหมาย

ในส่วนแสดงผลจะเกิดจากผลที่ได้จากส่วนประมวลผลนั้นเอง และแสดงออกทางตัวแสดงผลอย่างชัดเจน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.4 ไมโครคอนโทรเลอร์

นอกจากเครื่องสั่งงานและตอบรับทางโทรศัพท์ จะรับสัญญาณจากผู้ใช้โดยผ่านทางสายโทรศัพท์ และนำไปประมวลผล ซึ่งสัญญาณเหล่านั้นจะมีหลายรูปแบบ เช่น สัญญาณระบบคู่ความถี่ แปลงเป็นสัญญาณ 4 บิต สัญญาณกระดิ่ง แปลงเป็นสัญญาณพัลส์ และจะถูกส่งมายังหน่วยประมวลผลในที่นี้ คือ ซิงเกิลชิพไมโครโปรเซสเซอร์ (Single chip microprocessor) เบอร์ 8051 เพื่อใช้ในการควบคุมอุปกรณ์ต่างๆ

ในการควบคุมกับเครื่องเฉพาะงานนั้น ไม่จำเป็นต้องใช้องค์ประกอบที่จำเป็นเฉพาะในการควบคุมนั้นๆ มารวมเป็นชิพเดียว(Single ship) ทำให้ขนาดลดลงเหลือขนาดเท่ากับไมโครโปรเซสเซอร์ (Microprocessor) ทั่วๆไป ซึ่งจะแสดงองค์ประกอบของซิงเกิลชิพตระกูล MCS-51

2.1.4.1 การจัดขาและลักษณะภายนอกของ 8051

-ขาต่อลงดิน(Ground) คือขา 20 เป็นขาสำหรับต่อลงดิน

-ขา Vcc (ขา40) เป็นขาที่ต่อกับแรงดันไฟตรงขนาด 5 โวลท์

-ขา พอร์ท 0(P0.0-P0.7) คือขา 32-39 เป็นพอร์ท I/O 8 บิตแบบโอเพนเดรนไบไดเรกชันแนล(open drain bi-directional) สามารถที่จะรับโหลดทีทีแอล(TTL) ได้ 8 ตัว การเขียนค่า "1" ลงไปที่พอร์ทนี้ จะเป็นการปล่อยขาลอย (Float) ขาของพอร์ทนี้ ทำให้มันทำงานอินพุทมีอิมพีแดนซ์สูงในการให้พอร์ทนี้บริการแบบ I/O พอร์ท 0 ยังใช้งานเป็นตัวส่งข้อมูล โดยเมื่อใช้บริการทางด้านการตรวจสอบโปรแกรม (ROM) ภายใน ถ้าใช้งานในลักษณะนี้การพูลอัพ (Pull up) จากภายนอกต้องต่อด้วยค่า 10 kΩ

-ขา พอร์ท 1 (P1.0-P1.7) คือขา 1-8 เป็นพอร์ท I/O แบบ 8 บิตแบบโอเพนเดรนไบไดเรกชันแนลพร้อมด้วยการพูลอัพภายในถ้าเป็นพอร์ทเอาท์พุทบัฟเฟอร์ สามารถขับโหลดทีทีแอล ได้ 4 ตัว พอร์ท 1 เมื่อถูกเขียนค่า "1" ด้วยโปรแกรมจะมีสถานะสูงด้วยการพูลอัพภายใน การใช้สาถานะเช่นนี้จะเป็นการเริ่มต้น การใช้งานพอร์ทนี้เป็นอินพุทการให้สัญญาณลงต่ำ จะเป็นการจ่ายกระแสออกเนื่องจากการพูลอัพภายใน

-ขา พอร์ท 2 (P2.0-P2.7) คือขา 21-28 เป็นพอร์ท I/O แบบ 8 บิตแบบโอเพนเดรนไบไดเรกชันแนล ด้วยการพูลอัพภายใน พอร์ท 2 ที่ทำหน้าที่เป็นบัฟเฟอร์เอาท์พุท (Output buffer) สามารถจ่ายโหลดทีทีแอลได้ 4 ตัว พอร์ทจะถูกใช้งานเป็นตัวส่งแอสเดรสไบท์สูงด้วย เมื่อใช้งานร่วมกับหน่วยความจำภายนอกเพื่อใช้แอสเดรสได้ถึง 16 บิต

-ขา พอร์ท 3 (P3.0-P3.7) คือขา 10-17 เป็นพอร์ท I/O แบบ 8 บิตแบบ พูลอัพภายใน นอกจากจะทำตัวเป็นพอร์ท I/O รับโหลด ทีทีแอล 4 ตัวแล้ว ยังใช้งานเป็นพิเศษสำหรับ MCS-51 ตามรายการข้างล่างนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขาพอร์ท	ขา	การทำงานตามฟังก์ชันพิเศษ	
P3.0	10	RxD	พอร์ทอนุกรมอินพุท
P3.1	11	TxD	พอร์ทอนุกรมอินพุท
P3.2	12	INT0	อินเตอร์รัปต์ภายนอกตัวที่ 0
P3.3	13	INT1	อินเตอร์รัปต์ภายนอกตัวที่ 1
P3.4	14	T0	สัญญาณกระตุ้นเข้าที่ตัวนับ 0
P3.5	15	T1	สัญญาณกระตุ้นเข้าที่ตัวนับ 1
P3.6	16	WR	สัญญาณควบคุมการเขียน
P3.7	17	RD	สัญญาณควบคุมการอ่าน

การที่จะทำงานตามฟังก์ชันข้างบนจะต้องเริ่มโปรแกรมโดยการส่งค่า "1" ไป แลทซ์ ไว้ก่อนที่จะให้ทำงานตามฟังก์ชันข้างบน

-ขารีเซต (RST) คือขา 9 ต้องคงสถานะค่าสูงเป็นเวลาอย่างน้อยสองวัฏจักร ระหว่างที่ออสซิเลเตอร์ทำงาน ขณะที่ต้องการรีเซตทั้งระบบ

-ขา ALE/PROG คือขา 30 เป็นขาแอดเดรสแลทซ์อีนาเบล (Address latch enable) ด้วยการส่งพัลส์ ออกไปใช้สำหรับแลทซ์ ค่าแอสเดรสไบต์ต่ำจากพอร์ท 0 ในระหว่างการเข้าถึงข้อมูลจากหน่วยความจำภายใน ALE จะถูกส่งสัญญาณนาฬิกาออกมาในอัตราความเร็วคงที่ ที่ 1/8 ของความถี่ออสซิลเลเตอร์ตลอดเวลา แม้ว่าจะไม่มีการเข้าถึงข้อมูลภายใน ดังนั้นจึงสามารถที่จะใช้สัญญาณจากขานี้เป็นตัวตั้งสัญญาณภายนอก หรือเป็นความถี่สัญญาณนาฬิกา แต่อย่างไรก็ตามความถี่สัญญาณนี้ จะลดความถี่ช้าลงไปเท่าหนึ่งระหว่างการทำงานแบบการภายใน ดังนั้นจึงสามารถที่จะใช้สัญญาณจากขานี้เป็นตัวตั้งสัญญาณภายนอก หรือเป็นความถี่สัญญาณนาฬิกา แต่อย่างไรก็ตามความถี่สัญญาณนี้ จะลดความถี่ไปเท่าหนึ่งระหว่างการทำงาน แบบการเข้าภายนอก ขานี้ยังเป็นสัญญาณพัลส์ เข้าสำหรับการควบคุมโปรแกรม EPROM ภายในชิพ

-ขา PSEN (Program Storage Enable) คือขา 29 เป็นสโตรบ(Strop) สำหรับอ่านข้อมูลจากโปรแกรมหน่วยความจำภายนอก เมื่อชิพทำงานด้วยโปรแกรมจากหน่วยความจำภายนอก ขา PSEN จะสร้างสโตปต่อสองครั้งภายใน 1 แมชีนไซเคล (machine cycle)

-ขา EA/Vpp คือขา 31 มีสถานะสูง ตัว CPU ทำงานตามหน่วยความจำภายนอก ซึ่งขยายโปรแกรมได้ยาวถึง 64 กิโลไบต์

-ขา XTAL1 คือขา 19 ใช้เป็นตัวอินพุทเข้าสู่ตัวออสซิลเลเตอร์ขยายแบบ อินเวอท (invert)

-ขา XTAL2 คือขา 18 ใช้เป็นตัวเอาท์พุทจากตัวออสซิลเลเตอร์ขยายแบบ อินเวอท (invert)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.4.2 การจัดหน่วยความจำและการเชื่อมต่อ

หน่วยความจำไมโครคอนโทรลเลอร์ 8051 แบ่งเป็น 2 ประเภท คือ

- 1.หน่วยความจำโปรแกรม (Program memory) หน่วยความจำประเภทนี้คือ ROM ใช้เก็บโปรแกรมที่ใช้ควบคุมระบบ ซึ่งเป็นหน่วยความจำประเภท non-volatile
- 2.หน่วยความจำข้อมูล (Data memory) หน่วยความจำประเภทนี้ได้แก่ RAM

1. หน่วยความจำโปรแกรม

MCS-51 เบอร์ 8051 จะมีหน่วยความจำในชิพ และสามารถสร้างหน่วยความจำภายนอกได้มากที่สุด และอ้างหน่วยความจำข้อมูลภายนอกได้มากที่สุด 64 กิโลไบต์ สำหรับหน่วยความจำ RAM จะประกอบไปด้วยพื้นที่ใช้งานทั่วไป,รีจิสเตอร์แบงค์,พื้นที่ใช้งานระดับบิต และรีจิสเตอร์ฟังก์ชันพิเศษ ใน 8051 จะมีหน่วยความจำภายในตั้งแต่ตำแหน่ง 00H ถึง FFH และสามารถอ้างหน่วยความจำภายนอกได้ 64000 ตำแหน่ง ถ้าอ่านข้อมูลจากหน่วยความจำโปรแกรมขา PSEN จะแอกทีฟ นอกจากนี้ 8051 สามารถอ้างถึงหน่วยความจำภายนอกได้ 64000 ตำแหน่ง โดยการติดต่อกับหน่วยความจำนี้ขา RD และ WR จะแอกทีฟ สำหรับหน่วยความจำข้อมูลภายในจะแบ่งออกได้ดังนี้

- 1) ชุดรีจิสเตอร์ 4 ชุดแต่ละชุดเรียกว่ารีจิสเตอร์แบงค์ ที่ตำแหน่ง 00H-1FH โดยแต่ละชุดประกอบด้วยรีจิสเตอร์ R0-R7
- 2) หน่วยความจำสามารถเข้าถึงข้อมูลระดับบิตได้ ตำแหน่ง 30H-7FH
- 3) หน่วยความจำใช้งานทั่วไปตำแหน่ง 80H-FFH
- 4) รีจิสเตอร์ฟังก์ชันพิเศษตำแหน่ง 80H-FFH

แผนผังการจัดหน่วยความจำข้อมูลภายในแสดงได้ดังรูป2.2 ซึ่งการอ้างถึงตำแหน่งจะอ้างได้สองแบบ คือการอ้างไปที่ตำแหน่งของไบต์ หรืออ้างไปที่ตำแหน่งของบิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

	D7	D6	D5	D4	D3	D2	D1	D0	
7FH	Scratch Pad Area								
30H									
2FH	7F	7E	7D	7C	7B	7A	79	78	
2EH	77	76	75	74	73	72	71	70	
2DH	6F	6E	6D	6C	6B	6A	69	68	
2CH	67	66	65	64	63	62	61	60	
2BH	5F	5E	5D	5C	5B	5A	59	58	
2AH	57	56	55	54	53	52	51	50	
29H	4F	4E	4D	4C	4B	4A	49	48	
28H	47	46	45	44	43	42	41	40	
27H	3F	3E	3D	3C	3B	3A	39	38	
26H	37	36	35	34	33	32	31	30	
25H	2F	2E	2D	2C	2B	2A	29	28	
24H	27	26	25	24	23	22	21	20	
23H	1F	1E	1D	1C	1B	1A	19	18	
22H	17	16	15	14	13	12	11	10	
21H	0F	0E	0D	0C	0B	0A	09	08	
20H	07	06	05	04	03	02	01	00	
1FH	R0-R7								รีจีสเตอร์ แบงค์ 3
18H									
17H	R0-R7								รีจีสเตอร์ แบงค์ 2
10H									
0FH	R0-R7								รีจีสเตอร์ แบงค์ 1
08H									
07H	R0-R7								รีจีสเตอร์ แบงค์ 0
00H									

รูปที่ 2.2 ตำแหน่งของหน่วยความจำทั้งแบบไบต์ และแบบบิต

ใน 8051 จะแบ่งหน่วยความจำประเภทนี้ออกเป็น 2 ประเภท หน่วยความจำโปรแกรมภายนอก(external memory) กล่าวคือ ROM ที่มาต่อภายนอกตัว 8051 ส่วนอีกประเภทหนึ่ง คือ หน่วยความจำภายใน (internal memory) ได้แก่ ROM ภายในตัวไมโครคอนโทรเลอร์เอง การเลือกใช้หน่วยความจำโปรแกรมภายนอกหรือหน่วยความจำโปรแกรมภายในทำได้โดยการให้สัญญาณไฟฟ้าที่ขา EA/ โดย

-สัญญาณไฟฟ้าที่ขา EA/ เป็นลอจิก 0 หมายถึงหน่วยความจำโปรแกรมภายนอก

-สัญญาณไฟฟ้าที่ขา EA/ เป็นลอจิก 1 หมายถึงหน่วยความจำโปรแกรมภายใน

*หมายเหตุ หน่วยความจำโปรแกรมนั้น ไมโครคอลโทรเลอร์ตระกูล 8051 สามารถอ้างข้อมูลได้ 64 กิโลไบต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. หน่วยความจำข้อมูล

หน่วยความจำข้อมูลมีหน้าที่สำหรับเก็บข้อมูล หรือตัวแปรที่เกิดขึ้นในขณะประมวลผลโปรแกรมไว้เป็นการชั่วคราว โดยพื้นฐานแล้วหน่วยความจำข้อมูลจัดเป็นหน่วยความจำ RAM แบบสแตติก ดังนั้นเมื่อไม่มีการจ่ายไฟให้ระบบ ก็จะมีผลทำให้มูลที่จัดเก็บไว้ภายในหน่วยความจำนี้สูญหายไป

พื้นที่ของหน่วยความจำข้อมูล ของ 8051 สามารถมีได้สูงสุดไม่เกิน 64 กิโลไบต์ และแยกประเภทออกเป็นสองลักษณะตามตำแหน่งที่ตั้งของหน่วยความจำนั้น

-หน่วยความจำข้อมูลภายใน (internal data memory) ซึ่งเป็นหน่วยความจำข้อมูล ที่อยู่ภายในตัวไอซีไมโครคอนโทรลเลอร์เอง

-หน่วยความจำข้อมูลภายนอก(external data memory) ซึ่งเป็นการใช้ไอซีหน่วยความจำข้อมูลมาเพิ่มเข้าไปในวงจร

-หน่วยความจำข้อมูลภายใน

ใน 8051 มีทั้งหมด 256 ไบต์ โดยจำแนกออกเป็นสองลักษณะ คือพื้นสำหรับหน่วยประมวลผลกลาง หรือเรียกว่า รีจิสเตอร์ R0-R7 และพื้นที่ใช้งานทั่วไปสำหรับโปรแกรมใช้งานที่ผู้ใช้สร้างขึ้นมา

หน่วยความจำขนาด 128 ไบต์แรก

บริเวณแอดเดรส 00H-1FH จำนวน 32 ไบต์จำแนกออกเป็นกลุ่ม(Blank) ได้ 4 กลุ่ม กลุ่มละ 8 ไบต์ ดังตารางต่อไปนี้

แอดเดรส	รีจิสเตอร์แบงค์	ชื่อรีจิสเตอร์ที่ต้องการ
00H-07H	1	R0-R7
08H-0FH	2	R0-R7
10H-17H	3	R0-R7
18H-1FH	4	R0-R7

ตารางที่ 2.1 การจำแนกแอดเดรสรีจิสเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเห็นได้ว่าชื่อรีจีสเตอร์ไม่ว่าจะอยู่ในรีจิสเตอร์แบงค์ใด ก็จะมีชื่อ R0 ถึง R7 เหมือนกัน ดังนั้นในการใช้งานผู้ใช้จะต้องระวังในการเลือกใช้รีจิสเตอร์ จากแบงค์ต่างๆ โดยจะต้องทำการเลือกแบงค์รีจีสเตอร์ด้วยการกำหนดค่าบิตที่อยู่ภายในรีจีสเตอร์ PSW เท่ากับตามตารางที่อยู่ข้างล่างนี้

รีจิสเตอร์แบงค์	บิต RSO	บิต RS1	ตำแหน่ง หน่วยความจำ
0	0	0	0000H
1	0	1	0008H
2	1	0	0010H
3	1	1	0018H

ตารางที่ 2.2 การกำหนดค่าของบิตที่อยู่ภายในรีจิสเตอร์ PSW

อย่างไรก็ตามโดยทั่วไปมักจะมีการใช้งานรีจิสเตอร์ R0-R7 เฉพาะในแบงค์ 0 เท่านั้น ดังนั้นพื้นที่ของแบงค์อื่นๆ ที่เหลือก็สามารถนำมาใช้เป็นหน่วยความจำปกติด้วยการอ้างถึงหมายเลขของแอสเดรสนั้น ๆ โดยตรง

บริเวณแอดเดรส 20H-2FH จำนวน 16 ไบต์ เป็นส่วนที่สามารถอ้างถึงข้อมูลได้ถึงระดับบิต โดยจะมีตำแหน่งระดับบิตทั้งหมดถึง 128 บิต

บริเวณแอดเดรส 30H-7FH เป็นบริเวณที่สามารถนำไปใช้งานได้อย่างอิสระ โดยสามารถอ้างถึงได้เฉพาะในลักษณะไบต์ข้อมูลตามปกติเท่านั้น

หน่วยความจำข้อมูลขนาด 128 ไบต์ ถัดไป คือ แอดเดรสตั้งแต่ 80H-FFH เป็นบริเวณที่มีการใช้งานเฉพาะ 8051 เท่านั้น โดยจะนำมาใช้งานเป็นตำแหน่งของรีจิสเตอร์หน้าที่พิเศษ(Special-Function Register) จำนวน 20 ตำแหน่ง สำหรับไมโครคอนโทรลเลอร์เบอร์ 8051 จะมีหน่วยความจำข้อมูลภายในเพิ่มขึ้นอีก 128 ไบต์ โดยจะอยู่บริเวณช่วงแอดเดรส 80H-FFH เช่นกัน ซึ่งแม้ว่าจะมีหมายเลขแอดเดรสที่เดียวกันกับรีจีสเตอร์หน้าที่พิเศษ แต่ในความเป็นจริงแล้วจะเป็นพื้นที่หน่วยความจำอีกส่วนหนึ่ง ซึ่งมีการซ้อนเกย(overlap) กัน ซึ่งหากว่าผู้ใช้ต้องการใช้ข้อมูลบริเวณนี้ก็จะต้องอ้างถึงหน่วยความจำแบบโดยอ้อมเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชื่อรีจิสเตอร์	การอ้างถึงแบบบิต	ชื่อรีจิสเตอร์	การอ้างถึงแบบบิต
ACC	ได้	IE	ไม่ได้
B	ได้	TMOD	ไม่ได้
PSW	ได้	TCON	ไม่ได้
SP	ได้	TH0	ไม่ได้
DPTR	ได้	TL0	ไม่ได้
P0	ได้	TH1	ไม่ได้
P1	ได้	TL1	ไม่ได้
P2	ได้	SCON	ไม่ได้
P3	ได้	SBUF	ไม่ได้
IP	ได้	PCON	ไม่ได้

ตารางที่ 2.3 ความสามารถอ้างถึงแบบบิตของรีจิสเตอร์

รีจิสเตอร์หน้าที่พิเศษ

รีจิสเตอร์หน้าที่พิเศษ (SFR) เป็นรีจิสเตอร์สำหรับควบคุมหน้าที่และการทำงานของอุปกรณ์ หรือพอร์ทของ 8051 ทั้งหมดโดยมีตำแหน่งอยู่บริเวณแอดเดรส 80H-FFH การใช้งานรีจิสเตอร์หน้าที่พิเศษเหล่านี้ สามารถทำได้ทั้งการระบุถึงชื่อของรีจิสเตอร์ หรือตำแหน่งแอดเดรสที่เป็นของรีจิสเตอร์นั้นก็ได้

ในตาราง2.3 แสดงเห็นลักษณะการจัดพื้นที่หน่วยความจำ สำหรับรีจีสเตอร์หน้าที่พิเศษเหล่านี้โดยมีข้อสังเกตว่ารีจีสเตอร์ที่อยู่ตำแหน่งแอดเดรสที่เป็นจำนวนทวีคูณของค่า 8 จะสามารถอ้างถึงในระดับบิตได้ด้วย (นั้นคือ แอดเดรส 80H,88H,90H,98H,A0H,A8H,B0H,B8H,D0H,E0H และ F0H)

-แอกคิวมูเลเตอร์ (Accumulator) หรือ Acc

เป็นรีจิสเตอร์ขนาด 8 บิต ทำหน้าที่เก็บข้อมูลที่จะส่งให้กลับหน่วยความจำภายใน ซีพียู และเก็บผลลัพธ์ที่ได้จากการทำงานนั้น การใช้งานภายในจะเรียกว่ารีจิสเตอร์ A

-รีจิสเตอร์ B

เป็นรีจิสเตอร์ที่ใช้ทำงานคำสั่งการคูณ และการหารตัวเลข ในกรณีที่ไม่ได้ใช้งานทางคณิตศาสตร์ก็สามารถนำไปใช้งานเช่นเดียวกับรีจิสเตอร์ทั่วไปได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-โปรแกรมเคาน์เตอร์ (Program Counter)

เป็นรีจีสเตอร์ที่ใช้ทำงานในการเป็นตัวชี้ตำแหน่งแอดเดรสของหน่วยความจำโปรแกรม ซึ่งจะต้องไปทำงานในลำดับถัดไป ในการใช้งานภายในจะเรียกว่า รีจิสเตอร์ PC

-สแต็กพอยน์เตอร์ (Stack Pointer)

เป็นรีจีสเตอร์ขนาด 8 บิต ทำหน้าที่เก็บตำแหน่งของตัวชี้ หรือพอยน์เตอร์(Pointer) ของบริเวณ สแต็ก(Stack) สำหรับเก็บข้อมูลแอกคิวมูเลเตอร์ รีจิสเตอร์ต่างๆ รวมทั้งข้อมูลจากโปรแกรม ในการใช้งานภายในจะเรียกว่า รีจิสเตอร์ SP

-ตัวชี้ข้อมูลหรือดาต้าพอยน์เตอร์ (Data pointer)

เป็นรีจิสเตอร์ขนาด 16 บิต ซึ่งเรียกว่า DPTR และสามารถใช้งานแยกออกเป็นรีจิสเตอร์ขนาด 8 บิต 2 ตัว คือ DPH และ DPL เพื่อเก็บค่าของแอดเดรสของหน่วยความจำที่ต้องที่ต้องใช้งานภายในโปรแกรมหรืออาจจะเป็นแอดเดรสของอุปกรณ์ภายนอก ซึ่งกำหนดให้ติดต่อกันโดยใช้ตำแหน่งของหน่วยความจำภายในโปรแกรม

-โปรแกรมสเตตัสเวิร์ด (PSW)

รีจิสเตอร์นี้ทำหน้าที่บอกถึงแฟล็กสภาวะการทำงานต่างๆ รวมทั้งบิตสำหรับการกำหนดเลือกแบงก์ (Bank) ของรีจีสเตอร์ที่ใช้งานด้วย ดังแสดงในตาราง 2.4

*ชื่อบิต : PSW ตำแหน่ง : D0H ค่าบิตเริ่มต้น : 0000 0111

ชื่อบิต	ตำแหน่ง	ความหมาย
CY	PSW.7	Carry Flag
AC	PSW.6	Auxiliary Carry Flag
F0	PSW.5	Flag 0
RS1	PSW.4	Select Blank bit 1
RS0	PSW.3	Select Blank bit 0
OV	PSW.2	Overflow Flag
-	PSW.1	
P	PSW.0	Parity Flag

ตารางที่ 2.4 ความหมายของโปรแกรม สเตตัสเวิร์ด (PSW)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-รีจิสเตอร์ที่เกี่ยวข้องกับพอร์ท (Port Register)

รีจิสเตอร์เหล่านี้จะมีความเกี่ยวข้องกับการทำงานของพอร์ทอินพุท/เอาท์พุทโดยตรงซึ่งแต่ละตัวจะเป็นรีจิสเตอร์ขนาด 8 บิต สามารถใช้งานได้ทั้งในลักษณะของการอินพุท หรือเอาท์พุทข้อมูลได้ การดำเนินการใดๆ ที่เกี่ยวกับพอร์ททั้งสี่นี้จะมีผลทำให้ข้อมูลที่ตำแหน่งของพอร์ทเหล่านี้เปลี่ยนแปลงไปเช่นกัน

-รีจิสเตอร์ SBUF

เป็นบัฟเฟอร์ขนาด 8 บิต สำหรับการสื่อสารแบบอนุกรมทั้งการรับและการส่งข้อมูลซึ่งตามความเป็นจริง แล้วบัฟเฟอร์นี้มีอยู่ด้วยกันสองชุดและแยกจากกันอย่างชัดเจน สำหรับการส่งและการรับ โดยซีพียูจะจัดการเลือกบัฟเฟอร์ที่ถูกต้องโดยอัตโนมัติ

-รีจีสเตอร์ PCON

เป็นรีจีสเตอร์ที่ใช้ในการควบคุมหน้าที่การทำงานอยู่สามลักษณะ ซึ่งได้แก่ การควบคุมการทำงานของโปรเซสเซอร์ (บิต IDL และ PD 1) การกำหนดอัตราเร็วในการสื่อสารข้อมูลอนุกรม (บิต SMOD) และแฟล็กสภาวะสำหรับการทำงานทั่วไป (บิต GR0 และ GR1)

*ชื่อบิต : PCON ตำแหน่ง : 97H ค่าบิตเริ่มต้น :0xxx 0000

ชื่อบิต	ตำแหน่ง	ความหมาย
SMOD	PCON.7	บิตทวีคูณของอัตราบวกปกติ
-	PCON.6	-
-	PCON.5	-
-	PCON.4	-
GF1	PCON.3	แฟล็กสำหรับให้ผู้ใช้ ใช้งานทั่วไป Flag 0
GF0	PCON.2	แฟล็กสำหรับให้ผู้ใช้ ใช้งานทั่วไป Flag 1
PD	PCON.1	บิตสำหรับการกำหนด Power down
IDL	PCON.0	บิตสำหรับการกำหนด Ideal โหมด

ตารางที่ 2.5 ความหมายของโปรแกรมพาวเวอร์คอนโทรลรีจิสเตอร์ (PCON)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บิต PD (Power down) เป็นการกำหนดให้ลดกำลังไฟฟ้าที่จ่ายให้กับส่วนของโปรเซสเซอร์ภายในลงโดยยังคงมีการจ่ายไฟฟ้าให้กับส่วนหน่วยความจำข้อมูลภายในผ่านทางขาสัญญาณ RST วิธีการนี้มักนำมาใช้ในกรณีที่มีการตรวจสอบการไม่มีกำลังไฟฟ้า (Power failure) โดย วงจรตรวจสอบภายนอกจะต้องมีการอินเตอร์รัปต์เข้ามา เพื่อทำการเก็บข้อมูลที่กำลังประมวลผลอยู่ก่อนและเมื่อมีกระแสจ่ายให้เป็นปกติแล้ว จึงค่อยนำข้อมูลนั้นมาประมวลผลต่อไป

บิต IDL (Idle Mode) เป็นการกำหนดให้โปรเซสเซอร์หยุดการทำงานชั่วขณะ (sleep) และจะกลับอยู่ในสภาพปกติอีกครั้งเมื่อ ทำการรีเซตทางฮาร์ดแวร์ หรือมีการอินเตอร์รัปต์อย่างใดอย่างหนึ่งเกิดขึ้นการทำงานในลักษณะนี้สามารถเกิดขึ้นได้ ก็เนื่องจากสภาวะการหยุดการทำงานชั่วขณะนั้น เป็นเพียงการห้ามไม่ให้มีสัญญาณนาฬิกาจ่ายในส่วนของโปรเซสเซอร์เท่านั้น ส่วนของวงจรอินเตอร์รัปต์พอร์ทอนุกรมและวงจรนับ/จับเวลา ยังคงมีสัญญาณนาฬิกาอยู่เป็นปกติ

-รีจีสเตอร์ IP,IE,TMOD,SCON

เป็นกลุ่มรีจีสเตอร์ที่ทำหน้าที่กำหนดการควบคุม และการทำงานอินเตอร์รัปต์ต่างๆ ของ 8051

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.4.3 การอินเตอร์รัปต์ (interrupt) ใน 8051

ประเภทของการอินเตอร์รัปต์

1.อินเตอร์รัปต์ภายนอก การตรวจสอบสัญญาณอินเตอร์รัปต์ นี้ จะสามารถกำหนดให้มีการตรวจสอบในลักษณะเมื่อได้มีการเปลี่ยนแปลงระดับสัญญาณ (Level-sensitive) ไปแล้ว หรือในช่วงเวลาขณะเริ่มมีการเปลี่ยนแปลงสัญญาณจาก ลอจิกสูงไปต่ำ (Edge-sensitive)

2.อินเตอร์รัปต์ภายใน แหล่งกำเนิดสัญญาณนี้จะอยู่ภายในไมโครคอนโทรลเลอร์เอง เช่น วงจรนับจับเวลา เป็นต้น

โครงสร้างการ อินเตอร์รัปต์ เกิดได้ 5 ลักษณะ คือ

-INT0 สัญญาณอินเตอร์รัปต์จากภายนอก ทางขาสัญญาณ P3.2 โดย 8051 จะทำการสุ่มตัวอย่าง สัญญาณเมื่อสิ้นสุดทุก Machine Cycle

-INT1 สัญญาณอินเตอร์รัปต์จากภายนอก ทางขาสัญญาณ P3.3 โดย 8051 จะทำการสุ่มตัวอย่าง สัญญาณเมื่อสิ้นสุดทุก แมชีนไซเคิล

-Timer0 สัญญาณการเกิด โอเวอร์โฟลว์ (Overflow) ของไทม์เมอร์ 0

-Timer0 สัญญาณการเกิด โอเวอร์โฟลว์ (Overflow) ของไทม์เมอร์1

-Serial Port การเกิดอินเตอร์รัปต์ที่เกิดจากการรับ/ส่งข้อมูลอนุกรม ทำให้มีผลต่อแฟลกอินเตอร์รัปต์ RI และ TI ตามลำดับ

การกำหนดให้ 8051 สามารถตอบรับการอินเตอร์รัปต์ แต่ละประเภท ทำได้โดยกำหนดบิต (bit ของข้อมูลที่เกี่ยวข้อง ซึ่งมักอยู่ในหน่วยความจำ TCON และ SCON) หากได้ว่ามีการกำหนดค่าของ บิต ซึ่งอยู่ในหน่วยความจำ IE (interrupt enable register) ด้วยแล้วก็ สามารถตอบรับการอินเตอร์รัปต์ ของสัญญาณนั้นๆ ได้ นอกจากนี้สัญญาณอินเตอร์รัปต์ แต่ละประเภทยังสามารถกำหนด ไพออริตี้ (piority) ของการอินเตอร์รัปต์ได้ 2 ลักษณะคือ ไพออริตี้ต่ำ หรือสูง

ตามโครงสร้างที่ด้านการจัดการ อินเตอร์รัปต์ ของ 8051 สามารถกำหนดเรียกเพื่อยินยอม หรือไม่ยินยอม (Enable/Disable) ให้มีการ อินเตอร์รัปต์ แต่ละสัญญาณได้ โดยวิธีการกำหนดค่าของบิตภายในหน่วยความจำ IE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.4.4 พอร์ทอินพุท/เอาต์พุทของ 8051

พอร์ท มีความหมายถึงแอดเดรสหนึ่งที่ได้รับการกำหนดไว้ เพื่อการโอนย้ายข้อมูลระหว่างไมโครคอนโทรลเลอร์กับอุปกรณ์ภายนอก การกำหนดประเภทของการติดต่อขึ้นอยู่กับทิศทางการไหลของข้อมูลเมื่อพิจารณาจากไมโครคอนโทรลเลอร์เป็นหลัก ดังนั้นการนำเข้าข้อมูลจากวงจรภายนอกจึงเรียกว่า การอินพุท และในกรณีตรงข้ามเพื่อ ส่งออกข้อมูลออกจะเรียกว่า การเอาต์พุท

เมื่อพิจารณาถึงวิธีการส่งข้อมูลภายในพอร์ท จะสามารถแยกประเภทของพอร์ทออกได้เป็น สองลักษณะ คือพอร์ทแบบขนาน (Parallel port) ซึ่งทำการส่งจำนวนบิตข้อมูลทั้งหมดออกมาหรือ นำเข้าไปในคราวเดียว และพอร์ทอนุกรม (Serial port) ซึ่งทำการโอนย้ายข้อมูลที่ละบิต จนครบจำนวน

<u>พอร์ทแบบขนานของ 8051</u>

8051 มีโครงสร้างของพอร์ทที่สามารถใช้งานแบบขนานได้จำนวนทั้งหมดสี่พอร์ทเรียกชื่อเรียงตามลำดับว่า พอร์ท 0,1,2 และ 3 และเป็นพอร์ทขนาด 8 บิตทั้งหมด การใช้งานพอร์ทสามารถใช้ได้ทั้งในลักษณะของเส้นสัญญาณเดี่ยวๆ หรือกลุ่มของสัญญาณได้ นอกจากนี้พอร์ท 0,2 และ 3 ยังสามารถนำไปใช้งานอื่นๆ ที่ไม่ใช่เป็นพอร์ท ได้โดยพอร์ท 0 จะทำหน้าที่มัลติเพล็กซ์ ระหว่างบัสแอดเดรสไบต์ต่ำและบัสข้อมูลสำหรับการติดต่อกับวงจรประกอบร่วมข้อมูลบัสแอดเดรสไบต์สูงซึ่งจะส่งออกทางพอร์ท 2 สำหรับพอร์ท 3 นั้น นอกเหนือไปจากความสามารถเช่นพอร์ทปกติแล้วสามารถนำไปเป็นขาสัญญาณอินเตอร์รัปต์ต่างๆ ซึ่งรวมทั้งการสร้างสัญญาณ ควบคุม RD\ และ WR\ เพื่อทำหน้าที่อ่านหรือเขียนหน่วยความจำข้อมูลภายนอกด้วย การใช้งานพอร์ทลักษณะงานแบบ อื่นๆที่ไม่ใช่เป็นพอร์ทเอาต์พุทนี้จะดำเนินการโดย 8051 เองโดยอัตโนมัติ

<u>โครงสร้างการทำงานของพอร์ท 8051</u>

จากลักษณะโครงสร้างของแต่ละบิต ภายในพอร์ททั้งหมดของ 8051 ซึ่งได้แสดงไว้ในรูปนั้นจะเห็นว่ามีความคล้ายคลึงกันตามลักษณะโครงสร้างที่เรียกว่า พอร์ทควอซิไบไดเร็กชันแนล (Quasi-bidirectional port) ยกเว้นพอร์ท 0 ซึ่งเพียงแต่ไม่มีตัวต้านทานทำหน้าที่ พูลอัพสัญญาณไว้ภายในเท่านั้น วงจรประกอบอื่นภายในยังมีฟลิปฟลอปแบบ D ซึ่งมีผลทำให้พอร์ทสามารถแลตช์หรือค้างสภาวะของสัญญาณได้ นอกจากนี้ในส่วนเอาต์พุทของฟลิปฟลอปเฉพาะของพอร์ท 0 และพอร์ท 2 จะมีโครงสร้างที่ทำหน้าที่คล้ายสวิตช์เพิ่มเติมขึ้น เพื่อควบคุมให้เอาต์พุทนี้ต่อเข้ากับส่วนของ ทรานซิสเตอร์ในระหว่างที่ไม่มีการทำงานในลักษณะบัสแอดเดรส หรือบัสข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง_พระจอมเกล้าลาดกระบัง

ด้วย สำหรับบัฟเฟอร์สองตัวของทุกบิตในพอร์ทนั้นมีการทำงานแยกกัน โดยอิสระโดยตัวที่อยู่ด้านบนจะยอมให้สัญญาณผ่านได้ก็ต่อเมื่อมีการอ่านค่าข้อมูลที่ค้างไว้ ส่วนอีกตัวหนึ่งซึ่งอยู่ด้านล่างจะถูกใช้งานเฉพาะเมื่อได้มีการอ่านสถานะของขาสัญญาณเท่านั้น

การใช้งานพอร์ทเป็นอินพุท

การใช้งานพอร์เป็นการอินพุท ข้อมูลจะต้องเริ่มต้นด้วยการส่งข้อมูลที่มีค่าเป็น 1 ออกมาทางบิตของพอร์ทนั้นก่อนเป็นลำดับแรกเพื่อหยุดการทำงานของทรานซิสเตอร์ที่ทำหน้าที่ ขับสัญญาณเอาต์พุท ของบิตนั้นๆ ทำให้ขาสัญญาณของบิต ถูกต่อเข้ากับตัวต้านทานที่ทำหน้าที่พูลอัพภายในซึ่งมีผลให้บิตนั้นๆ ของพอร์ท 1,2 และ 3 เป็น สภาวะของลอจิกสูง ตัวต้านทานนี้มีค่าประมาณ 50 kΩ ซึ่งเป็นค่าที่สูงมาก และทำให้อุปกรณ์ภายนอกสามารถขับสัญญาณของพอร์ทเหล่านี้เป็นลอจิกต่ำได้ง่าย สำหรับบิตของพอร์ท 0 นั้น แม้ว่าจะมีหลักการทำงานที่คล้ายคลึงกันกับบิตของพอร์ทอื่นๆ แต่เนื่องจากการที่ไม่มีตัวต้านทานทำหน้าที่พูลอัพภายใน ทำให้เมื่อทรานซิสเตอร์ที่ทำหน้าที่ ขับสัญญาณเอาต์พุทนั้นหยุดการทำงาน ก็จะเป็นผลให้ขาสัญญาณนี้อยู่ในสภาวะอิมพีแดนซ์สูงแทน

การใช้งานพอร์ทเป็นเอาต์พุท

เมื่อมีการส่งข้อมูลที่มีค่าเป็น 0 ให้กับแต่ละบิตของพอร์ททุกพอร์ท ข้อมูลนี้จะถูกส่งให้กับฟลิปฟลอปซึ่งจะค้างค่านี้ไว้ และมีผลทำให้ทรานซิสเตอร์ที่ทำหน้าที่ขับสัญญาณเอาต์พุทนั้นทำงาน ดังนั้นขาสัญญาณก็จะมีสภาวะ ลอจิกเป็นลอจิกต่ำส่วนข้อมูลที่มีค่าเป็น 1 ออกมานั้น ในกรณีที่เป็นการทำงานที่แต่ละบิตของพอร์ท 1,2 หรือ 3 จะทำให้ทรานซิสเตอร์ที่ทำหน้าที่ขับสัญญาณเอาต์พุทนั้นหยุดการทำงาน มีผลให้ขาของสัญญาณมีลอจิกสูงด้วยตัวต้านทานพูลอัพที่อยู่ภายในนั้น แต่สำหรับการทำงานในแต่ละบิตของพอร์ท 0 นั้นจะมีผลที่แตกต่างออกไป โดยขาสัญญาณจะเป็นสภาวะอิมพีแดนซ์สูงแทน เนื่องจากไม่มีตัวต้านทานภายในเชื่อมต่ออยู่นั้นเอง ดังนั้นในการใช้งานพอร์ท 0 เป็นเอาต์พุทข้อมูล จึงจำเป็นต้องใช้ตัวต้านทานภายนอก พูลอัพสัญญาณไว้กับลอจิกสูงแทน ความสามารถอีกอย่างหนึ่งเกี่ยวกับพอร์ทอินพุท/พอร์ทเอาต์พุทของ 8051 เป็นวิธีการอ่านลอจิกจากพอร์ท ซึ่งมีได้สองวิธีคือ การอ่านค่าลอจิกที่ขาสัญญาณ (Port pin) และวิธีการอ่านลอจิกของการแลตช์ที่พอร์ท (Port latch) ดังจะสังเกตได้จาก วิธีการอ่านค่าจากพอร์ททั้งสองแบบนี้จะช่วยให้ระบบทำงานได้ถูกต้องมากขึ้น

61349
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลักษณะสมบัติของพอร์ทอินพุท/เอาร์พุท

ดังได้กล่าวแล้วว่าพอร์ท 1,2 และ 3 ของ 8051 มีตัวต้านทาน (ซึ่งสร้างขึ้นจาก Fet) ทำหน้าที่พูลอัพสัญญาณไว้และมีค่าประมาณ 50 kΩ ซึ่งถือว่ามีค่าสูงมาก เป็นผลให้การเปลี่ยนแปลงระดับสัญญาณลอจิก จากสูงไปต่ำทำได้อย่างรวดเร็ว แต่ในกรณีตรงข้ามจะใช้เวลานานกว่ามาก ทั้งนี้เนื่องจากว่ากระแสไหลผ่านตัวต้านทานนี้ได้น้อยมาก ดังนั้นในการแก้ปัญหาจึงมีการออกแบบตัวต้านทานเพิ่มอีกหนึ่งตัว ขนานไว้โดยมีค่าประมาณ 1 k เรียกว่าความต้านทานสปีดอัพ (speed up) ซึ่งยอมให้กระแสไหลผ่านได้มากขึ้นประมาณ 50 – 100 เท่า และจะมีการเชื่อมต่อตัวต้านทานนี้เฉพาะเมื่อมีการเปลี่ยนจากลอจิกต่ำไปสูงเท่านั้น โดยใช้เวลาประมาณ 2 คล็อกไซเคิล คำสั่งการใช้งานพอร์ทอินพุท/เอาต์พุท เนื่องจาก 8051 ใช้หลักการที่ว่ากำหนดหน่วยความจำ (Memory mapped system) กล่าวคือ การอ้างถึงพอร์ทรีจิสเตอร์ หรืออุปกรณ์ต่างๆ ภายในระบบ จะเป็นการติดต่อกับหน่วยความจำตำแหน่งหนึ่งเท่านั้น ดังนั้นในการดำเนินการเพื่อนำ เข้าหรือส่งออกข้อมูลกับพอร์ท จึงใช้คำสั่งการอ่านค่าจากหน่วยความจำซึ่งถูกออกแบบให้เป็นตำแหน่งของพอร์ทหรือ คำสั่งการเขียนค่าข้อมูลไปยังตำแหน่งหน่วยความจำ ดังนั้นจะสังเกตเห็นได้ว่าในตารางชุดคำสั่งของ 8051 จะไม่มีคำสั่งที่เกี่ยวกับการทำงานพอร์ทแต่ประการใด เช่น คำสั่ง IN (นำเข้าข้อมูลจากพอร์ท) หรือคำสั่ง Out (ส่งข้อมูลออกจากพอร์ท)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.4.5 วงจรนับ/จับเวลา

8051 ประกอบด้วยหน่วยความจำ ขนาด 16 บิต จำนวน 2 ตัว คือ T0 (Timer0)และ T1(Timer1) ซึ่งสามารถนำไปใช้งานได้อย่างอิสระ โดยสามารถควบคุมให้ทำหน้าที่เป็นตัวจับเวลาเพื่อนับจำนวน พัลส์ สัญญาณนาฬิกาภายใน หรือควบคุมให้ทำหน้าที่เป็นตัวนับ(Counter)เพื่อนับจำนวนพัลส์ ของระบบได้ภายในหน่วยความจำ แต่ละตัวยังสามารถแยกออกได้เป็นหน่วยความจำขนาด 8 บิต คือ TH0,TL0,TH1 และTL1 โดยการทำงานของหน่วยความจำ ทั้ง 2 ตัวนี้ผลมาจากการกำหนดค่าของ บิต ที่อยู่ภายใน TMOD (Timer mode control register) และ TCON (Timer/Counter control register)

<u>การอินเตอร์รัปต์ วงจรนับตรวจจับเวลา</u>

จากกระบวนการทำงานของวงจรนับ/จับเวลาของ 8051 จำเป็นต้องกำหนดค่าเริ่มต้นให้กับหน่วยความจำ T0 หรือT1 ค่านี้เป็นจำนวน plusภายใน ที่จะต้องนับหรือค่าของจำนวน plus ภายนอก ที่เข้ามาทางขาสัญญาณ T0 หรือT1 ค่าตัวเลขภายในหน่วยความจำนี้จะต้องให้มีค่าน้อยกว่า ค่าที่ต้องการอยู่หนึ่งค่า ทั้งนี้เนื่องจากการทำงานของหน่วยความจำ จะเพิ่มค่าจากที่กำหนดไปเรื่อยๆ จนถึงค่าสูงสุดของหน่วยความจำ และกลับไปเป็นค่า 0 เมื่อมีการเกิดโอเวอร์โฟลว์เกิดขึ้น ทำให้เกิดการกำหนดค่าแฟลก เพื่อแจ้งให้ CPU ได้รับทราบ ดังนั้นโปรแกรมทั่วไปจึงมักใช้สภาวะของแฟลก (TF0 และTF1) ซึ่งเป็นบิตอยู่ในหน่วยความจำ TCON เพื่อตรวจสอบว่ากระบวนการนั้นได้เสร็จสิ้นลงแล้ว หรือใช้เพื่อทำการอินเตอร์รัปต์โปรแกรมเพื่อไป ส่วนควบคุมการทำงานของวงจรนับ/จับเวลา ซึ่งประกอบด้วยส่วนของการกำหนดสัญญาณไทม์เมอร์ หรือเคาท์เตอร์ และบิต หรือขาสัญญาณสำหรับการหยุดหรือทำงานของวงจรนับ

<u>การทำงานเป็นตัวจับเวลา</u>

- การจับเวลาในโหมด 0 การทำงานในโหมด 0 วงจรนับจับเวลาจะทำหน้าที่เป็นตัวนับขนาด 13 บิต (โดยใช้หน่วยความจำ TH0 หรือTH1 เป็นตัวนับขนาด 8 บิต และหน่วยความจำ TL0 หรือ TL1 มีขนาด 5 บิต)

- การจับเวลาในโหมด 1 การทำงานในโหมด 1 มีความคล้ายคลึงกันในโหมด 0 แต่แตกต่างกันที่โหมด 1 เป็นตัวนับขนาด 16 บิตเต็ม

- การจับเวลาในโหมด 2 การทำงานในโหมด 2 ของวงจรนับ/จับเวลาแตกต่างกันออกไปเพียงใช้ หน่วยความจำ TL0 (TL1) เป็น ตัวนับขนาด 8 บิต ส่วนหน่วยความจำ TH0 (TH1) เก็บค่าเริ่มต้นของการนับไว้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การจับเวลาในโหมด 3 การทำงานในโหมด 3 จะสามารถใช้ได้เฉพาะกับไทม์เมอร์ 0 เท่านั้น หากว่านำไปกำหนดให้กับ ไทม์เมอร์1 จะทำให้หยุดการทำงานไป เมื่อไทม์เมอร์ 0 ได้รับการกำหนดการทำงานในโหมด 3 จะมีผลทำให้หน่วยความจำของมันแยกการทำงานออกเป็นอิสระ โดยหน่วยความจำ TL0 จะถูกควบคุมจากบิตภายในหน่วยความจำ TCON และขาสัญญาณ INT0 และเมื่อมีการเกิดโอเวอร์โฟลว์ เกิดขึ้น จากค่า 0FFH เป็น 00H ก็จะมีผลให้แฟลก TF0 มีการเปลี่ยนแปลงเกิดขึ้น สำหรับหน่วยความจำ TH0 จะถูกกำหนดให้ทำงานในแบบของตัวจับเวลาภายใต้การควบคุมของ บิต TR1 ในหน่วยความจำ TCON เท่านั้น และหากเกิดโอเวอร์โฟลว์ จะมีผลเฉพาะต่อแฟลก TF1 ในส่วนไทม์เมอร์ 1 ขณะไทม์เมอร์ 0 กำหนดให้ทำงานในโหมด 3 ก็ยังจะสามารถทำงานในโหมดอื่นๆ ที่ไม่ใช่โหมด 3 ได้ เช่นเดิม ยกเว้นจะไม่มีการอินเตอร์รัปต์เกิดขึ้นเท่านั้น (เนื่องจากแฟลก TF1 ถูกใช้โดย ไทม์เมอร์ 0 ไปแล้ว)

<u>การทำงานเป็นตัวนับสัญญาณ</u>

การใช้งานในลักษณะตัวนับ (counter) โดยหลักแล้วจะเหมือนกับการทำงานเป็นตัวจับเวลา (Timer) แต่ต่างกันโดย ตัวนับแทนที่จะนับพัลส์สัญญาณภายในและผ่านวงจรหาร 12 มาเป็นการนับ พัลส์สัญญาณทางขาสัญญาณ T0(P3.4) ให้กับ ไทม์เมอร์ 0 หรือขาสัญญาณ T1(P3.5) ให้กับไทม์เมอร์ 1 เท่านั้น นอกจากนี้ก่อนจะเริ่มต้นใช้งานจะต้องกำหนดค่าของบิต C/T ภายในหน่วยความจำ TCON ให้มีค่าเป็น 1 เสียก่อน

<u>วงจรนับ/จับเวลา2 (Timer2)</u>

โหมด การทำงานของไทม์เมอร์ 2 ประกอบด้วย

-Capture โหมด สามารถเลือกใช้งานได้ 2 ลักษณะ ด้วยการกำหนดให้กับบิต EXEN2 ของหน่วยความจำ T2CON ดังนี้

1. เมื่อกำหนดบิต EXEN2 เป็น 0 ไทม์เมอร์ 2 ยังทำงานเป็นตัวนับ/ตรวจจับเวลา เมื่อมีการเกิดโอเวอร์โฟลว์ขึ้น บิตในหน่วยความจำ TF2 จะถูกเซต และสามารถนำไปสร้างการอินเตอร์รัปต์ขึ้นได้

2. เมื่อกำหนดค่า EXEN2 เป็น 1 การทำงานจะคลอบคลุมลักษณะการทำงานข้างต้นแต่จะเพิ่มเมื่อมีการเปลี่ยนแปลงระดับสัญญาณทางขาสัญญาณ T2EX จากลอจิกสูงไปเป็นลอจิกต่ำ จะมีผลทำให้ค่าข้อมูลภายในหน่วยความจำ ของไทม์เมอร์ 2 คือ TL2 และ TH2 ถูกนำไปใส่(Capture) ให้กับหน่วยความจำ RCAP 2L และ RCAP2H ซึ่งเป็นหน่วยความจำหน้าที่พิเศษ หรือ SFR ที่มีในไมโครคอนโทรลเลอร์เบอร์ 8052 เท่านั้น นอกจากนี้ยังมีผลให้บิต EXF2 ภายในหน่วยความจำ T2CON มีค่าเป็น 1 สามารถนำไปใช้งานในการอินเตอร์รัปต์ได้เช่นกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-Auto-reload โหมด สามารถทำงานได้ 2 ลักษณะเช่นเดียวกัน

Baud rate Generetor ของไทม์เมอร์ 2 จะมีความแตกต่างจากไทม์เมอร์ 1 โดยวงจรรับและการส่งสามารถเป็นค่าที่ต่างกันได้ขึ้นอยู่กับการกำหนดค่าของบิต TCLK และ RCLK ของไทม์เมอร์ 2

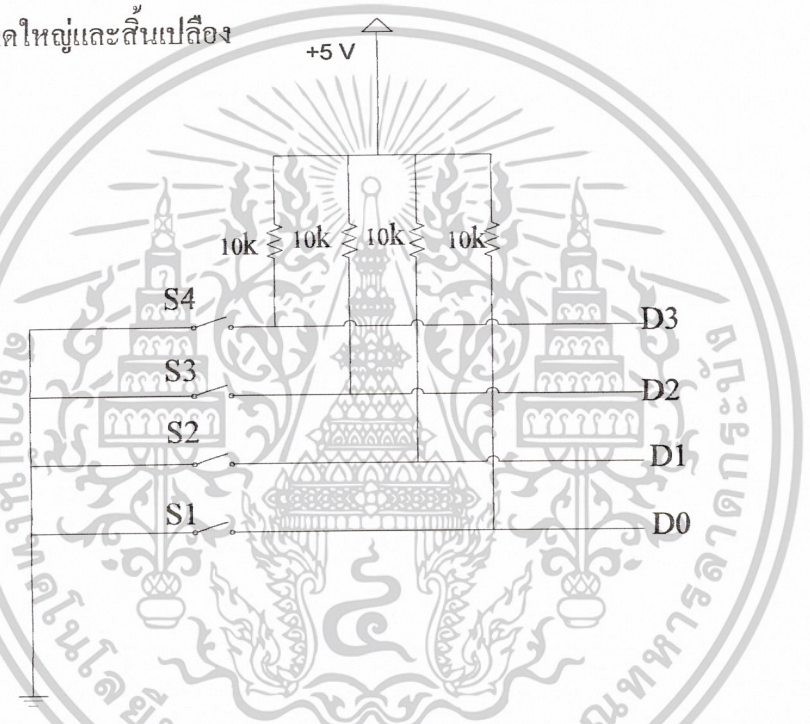
การทำงานของโหมดนี้คล้ายคลึงกับการทำงานใน ออโตรีโหลด (Auto-reload) โหมดดังกล่าวคือค่าในหน่วยความจำ TH2 เปลี่ยนแปลงจากค่า 0FFH ไปเป็นค่า 0 หรือที่เรียกว่า โอเวอร์โฟลว์ จะมีผลให้มีการโหลดข้อมูลขนาด 16 บิต จากหน่วยความจำ RCAP2H และ RCAP2L ซึ่งมีการเตรียมค่าล่วงหน้าแล้วโดยอัตโนมัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 ส่วนของการทำงานอัตโนมัติโดยใช้การตั้งเวลาซึ่งอาศัยฐานเวลาจริงจาก IC REAL TIME CLOCK

2.2.1 การเชื่อมต่อกับคีย์แพดหรือสวิตช์เมตริกซ์ 4x3 จุด กับไมโครคอนโทรลเลอร์

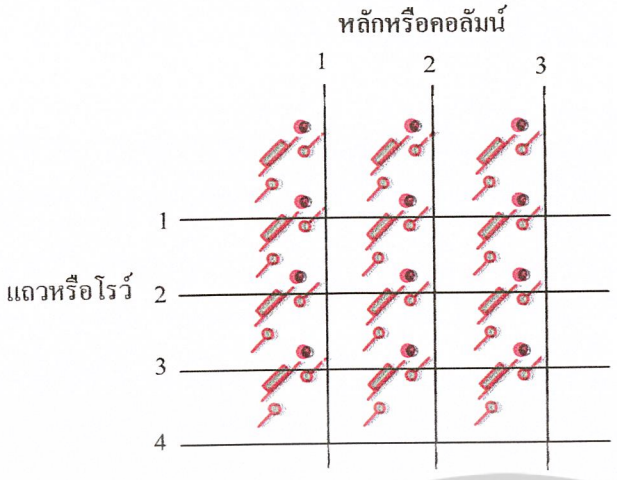
การอ่านค่าหรือรับค่าการกดสวิตช์เป็นอีกงานหนึ่งที่ไมโครคอนโทรลเลอร์ต้องสามารถรองรับและเชื่อมต่อใช้งานร่วมด้วยได้ วงจรการเชื่อมต่อของสวิตช์มีด้วยกัน 2 ลักษณะใหญ่ คือการต่อเข้ากับไฟเลี้ยงหรือกราวน์โดยตรง เมื่สวิตช์ตัวใดต่อวงจรสามารถอ่านค่าได โดยตรง ดังแสดงในรูป่ 2.3 วงจรในลักษณะนี้ไม่มีความซับซ้อน สามารถอ่านค่าของสวิตช์ได้ง่ายและรวดเร็ว แต่มีข้อเสียคือ ถ้าหากจำนวนของสวิตช์มีมากๆจำนวนของสายข้อมูลก็จะมีมากตาม ทำให้ระบบหรือวงจร โดยรวมมีขนาดใหญ่และสิ้นเปลือง



รูปที่ 2.3 วงจรของสวิตช์แบบต่อเข้ากับไฟเลี้ยงและกราวน์

วงจรของสวิตช์อีกลักษณะหนึ่งคือ การต่อวงจรแบบเมตริกซ์(matrix switch) ดังในรูปที่ 2.4 สวิตซ์จะถูกต่อกันในแนวแกนตั้งและแกนนอน จะเรียกแนวตั้งว่า หลักหรือคอลัมน์(column) ในขณะที่แนวนอนจะเรียกว่า แถวหรือโรว์(row) ดังนั้นค่าของสวิตซ์จะต้องประกอบด้วย ตำแหน่งในแนวหลักและแถว กระบวนการที่จะทำให้ได้มาซึ่งค่าของสวิตซ์มีขั้นตอนซับซ้อนพอสมควร แต่วงจรสวิตซ์แบบนี้มีข้อดีคือ สามารถรองรับการเพิ่มของสวิตซ์ได้อย่างสะดวก เพียงเพิ่มจำนวนสวิตซ์และแก้ไขซอฟต์แวร์อีกเล็กน้อยเท่านั้น ทำให้วงจรสวิตซ์แบบเมตริกซ์เป็นที่นิยมใช้มากในระบบควบคุมอัตโนมัติหรือกึ่งอัตโนมัติที่มีจำนวนสวิตซ์มากกว่า 8 ตัว ในการใช้งานทั่วไปจะเรียกสวิตซ์แบบนี้ว่า คีย์แพด(keypad)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

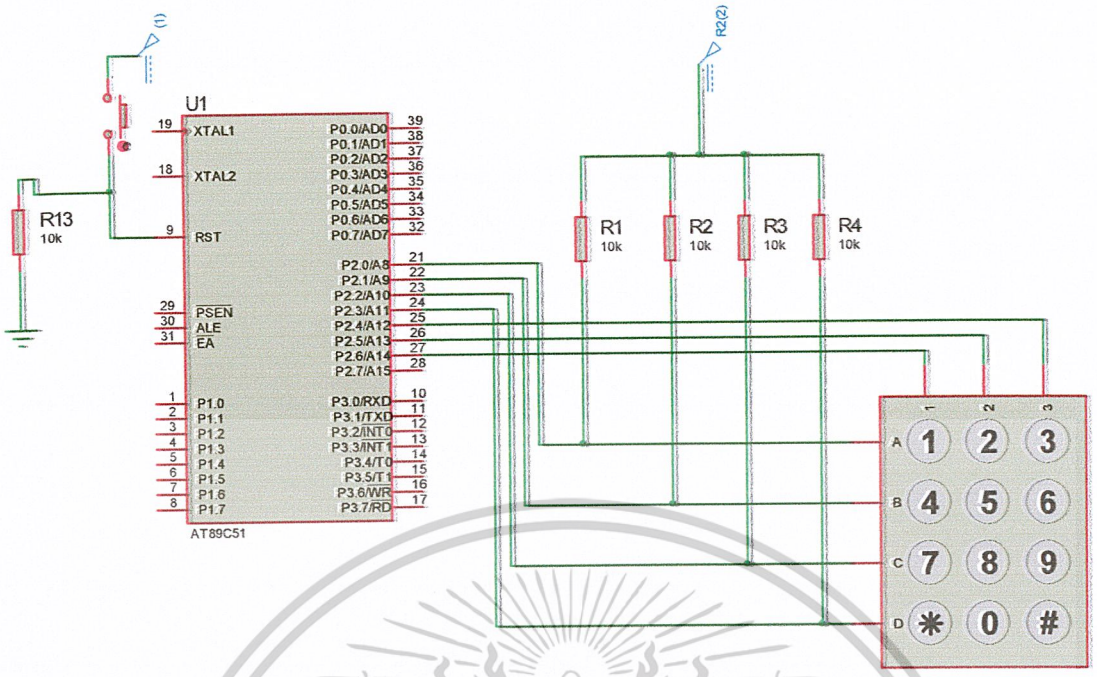


รูปที่ 2.4 วงจรของสวิตซ์แบบแมตริกซ์หรือคีย์แพด

การเชื่อมต่อคีย์แพดเข้ากับไมโครคอนโทรลเลอร์ MCS-51

วงจรเชื่อมต่อคีย์แพดเข้ากับไมโครคอนโทรลเลอร์แสดงในรูปที่ 2.5 จากรูปวงจรตัวอย่างจะใช้พอร์ต 2 ของไมโครคอนโทรลเลอร์เชื่อมต่อกับคีย์แพดทั้ง 7 เส้นคือ สายของคอลัมน์หรือหลัก 3 สาย C0-C2 และสายทางโรว์หรือแถวอีก 4 สาย R0-R3 โดยเฉพาะที่ขาพอร์ต P2.0-P2.3 จะต้องต่อตัวต้านทานพูลอัพอัปไว้เพื่อกำหนดสภาวะเริ่มต้นที่ไม่มีการกดคีย์ โดยไมโครคอนโทรลเลอร์จะทำการส่งข้อมูล “0” ไปยัง P2.6,P2.5 และ P2.4 ตามลำดับ ในทุกครั้งที่มีการส่งข้อมูลไปยังสายคอลัมน์ของคีย์แพด ไมโครคอนโทรลเลอร์จะทำการอ่านค่าที่ P2.0-P2.3 เข้ามาด้วย หากไม่มีการกด ค่าของ P2.0-P2.3 ก็จะเป็น “1”ทั้งหมด ถ้าหากมีการกดคีย์ ค่าของ P2.0-P2.3 ก็จะไม่เป็น 1111 อีกต่อไป เป็นการแจ้งให้ทราบว่า มีการกดคีย์แพดขึ้นแล้ว จากนั้นไมโครคอนโทรลเลอร์ก็จะทำการค้นหาตำแหน่งต่อไป โดยการค้นหาตำแหน่งสิ่งที่จะได้มาอย่างแรกคือ ค่าตำแหน่งของคีย์นั้น จากนั้นก็จะนำค่าตำแหน่งนั้นไปเปิดตารางข้อมูล เพื่อที่จะได้หมายเลขของคีย์ที่กดอย่างแท้จริง ยกตัวอย่างจากวงจรในรูปที่ 2.5 ถ้าหากคีย์ “0” ถูกกดไมโครคอนโทรลเลอร์จะได้ค่าตำแหน่งของคีย์ 0 เป็น 0BH จากนั้นจะนำค่า 0BH นี้ไปเปิดตารางก็จะได้ข้อมูลเป็น 0 หมายถึงคีย์ 0 ซึ่งกระบวนการหลังนี้จะเป็นหน้าที่ของซอฟต์แวร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.5 วงจรเชื่อมต่อคีย์แพดเข้ากับไมโครคอนโทรลเลอร์ MCS-51

2.2.2 การใช้งานโมดูลแสดงผลแบบผลึกเหลว(LCD module)กับ MCS-51

<u>รายละเอียดเกี่ยวกับโมดูล LCD</u>

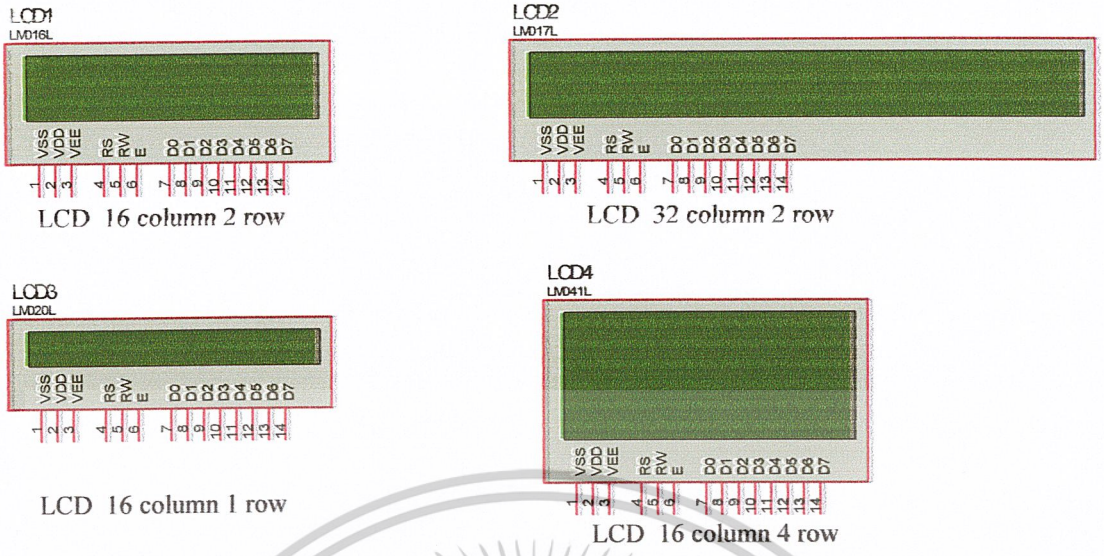
ในโมดูล LCD จะมีส่วนประกอบหลักๆ 3 ส่วน ดังนี้

ตัวแสดงผล(display) ภายในเป็นผลึกเหลวที่สามารถแสดงผลให้เห็น โดยอาศัยแสงจากภายนอก ดังนั้นจึงต้องมีมุมในการมองข้อมูลที่แสดงผลบนจอ LCD

ตัวควบคุม(controller) เป็นตัวรับข้อมูลจากอุปกรณ์ภายนอกมาควบคุมการทำงานของโมดูล LCD เช่น ลบจอภาพ แสดงตัวอักษร หรือเลื่อนเคอร์เซอร์ เป็นต้น

ตัวขับ(driver)เป็นตัวรับสัญญาณจากตัวควบคุมมาขับให้ตัวแสดงผลแสดงข้อมูลตามที่กำหนด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.6 แสดงรูปร่างและการจัดขาของโมดูล LCD

RS	R/W	E	การทำงาน
0	0		เขียนคำสั่ง
0	1		อ่านสถานะของโมดูลLCD
1	0		เขียนข้อมูล
1	1		อ่านข้อมูล

ตารางที่ 2.6 แสดงความสัมพันธ์ในการทำงานของขา RS,R/W และ E ของโมดูล LCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โมดูล LCD มีขาต่อใช้งานทั้งสิ้น 14 ขา มีการจัดขาดังแสดงรูปที่ 2.6 สำหรับรายละเอียดการทำงานของแต่ละขามีดังนี้

Vss(ขา1) : ต่อกราวน์

Vdd(ขา2): ต่อไฟเลี้ยง +5 V

Vo(ขา3) : เป็นขาอินพุทรับแรงดันเพื่อปรับความเข้มของการแสดงผล

Rs(ขา4) : เป็นขาอินพุทใช้ในการแยกชนิดของข้อมูลที่ทำการประมวลผลในขณะนั้นว่าเป็นคำสั่งสำหรับรีจิสเตอร์ คำสั่ง(IR) หรือเป็นข้อมูลสำหรับรีจิสเตอร์ข้อมูล(DR) โดยถ้าขานี้เป็น “0” ข้อมูลที่ส่งมาจะเป็นคำสั่ง แต่ถ้าขานี้เป็น “1” ข้อมูลที่ส่งมาจะเป็นข้อมูลสำหรับแสดงผล

R/W(ขา5) : เป็นขาที่ใช้เลือกการอ่านหรือเขียนข้อมูลกับโมดูล LCD ถ้าเป็น “0”เป็นการกำหนดให้เขียนข้อมูล แต่ถ้าเป็น “1” จะเป็นการอ่านข้อมูล

E(ขา6) : เป็นขาสำหรับรับสัญญาณพัลส์เอ็นเอเบิลโมดูล LCD ให้ทำงาน

D0-D7 : เป็นขาที่ใช้เป็นทางผ่านของข้อมูลระหว่าง LCD กับอุปกรณ์ภายนอกขนาด 8 บิต

*อนึ่งขา RS,R/W และ E จะใช้งานร่วมกัน โดยมีความสัมพันธ์ดังแสดงในตารางที่ 2.6

<u>คำสั่งควบคุมโมดูล LCD</u>

ในการเขียนคำสั่งลงในตัวควบคุม ต้องกำหนดให้ขา RS และ R/W เป็น “0” แล้วเขียนคำสั่งตามไป คำสั่งควบคุมโมดูล LCD ที่สำคัญมีคำสั่งดังนี้

1.คำสั่งเคลียร์ตัวแสดงผล(clear display)

มีคำสั่งเป็น 01H เป็นคำสั่งที่ใช้เขียนข้อมูลช่องว่าง หรือ space เข้าไปใน DDRAM ทั้งหมด เมื่อเคลียร์จอแสดงผลแล้ว แคอร์เซอร์จะกลับไปอยู่ที่ตำแหน่งซ้ายมือสุดของจอแสดงผล

2.คำสั่ง return home

ต้องกำหนดให้บิต 1 ของข้อมูลเป็น “1” เป็นคำสั่งให้เคอร์เซอร์เคลื่อนที่กลับไปยังตำแหน่งแรกสุดของจอแสดงผล แต่ข้อมูลบนจอแสดงผลไม่เปลี่ยนแปลง นั่นคือ ข้อมูลคำสั่งของคำสั่งนี้จะเป็น 02H หรือ 03H ก็ได้

3.คำสั่งเลือกโหมดการป้อนข้อมูล

มีรายละเอียดของรูปแบบข้อมูลคำสั่งดังนี้

บิต7	บิต6	บิต5	บิต4	บิต3	บิต2	บิต1	บิต0
0	0	0	0	0	1	I/D	S

บิต S เป็นบิตที่ใช้ในการกำหนดลักษณะของการแสดงผล เมื่อมีการป้อนข้อมูล ถ้าหากบิต S เป็น “1” เมื่อเกิดข้อมูลใหม่บนจอแสดงผล ตัวเคอร์เซอร์จะกระพริบอยู่กับที่ แต่ตัวอักษร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลเดิมจะถูกดันไปทางซ้าย แต่ถ้าหากบิตนี้เป็น “0” เมื่อเกิดข้อมูลใหม่เคอร์เซอร์จะเลื่อนไปทางขวามือ

บิต I/D เป็นบิตที่ใช้กำหนดว่า เมื่อเขียนหรืออ่านข้อมูลแล้ว แอสเดรสของ DDRAM เพิ่มขึ้นหรือลดลงหนึ่งแอสเดรส โดยถ้าบิตนี้เป็น “1” แอสเดรสของ DDRAM จะเพิ่มขึ้น แต่ถ้าเป็น “0” แอสเดรสจะลดลง

ดังนั้น ข้อมูลคำสั่งที่เกิดขึ้นสำหรับคำสั่งนี้ได้แก่ 04H-07H(4 ข้อมูลคำสั่ง) และที่ใช้บ่อยคือ 06H หมายถึง กำหนดให้เมื่อเกิดข้อมูลใหม่ เคอร์เซอร์จะเลื่อนไปทางขวามือ และแอสเดรสของ DDRAM เพิ่มขึ้น

4.คำสั่งควบคุมการแสดงผล

มีรายละเอียดของรูปแบบข้อมูลคำสั่งดังนี้

บิต7	บิต6	บิต5	บิต4	บิต3	บิต2	บิต1	บิต0
0	0	0	0	1	D	C	B

บิต D ใช้ควบคุมการเปิดปิดจอแสดงผล ถ้าบิตนี้เป็น “1”จะเป็นการเปิดจอแสดงผล ถ้าเป็น “0” จะเป็นการปิดจอแสดงผล

บิต C ใช้ควบคุมการแสดงตัวเคอร์เซอร์บนจอแสดงผล ถ้าต้องให้มีเคอร์เซอร์แสดงผลบนจอแสดงผล ต้องกำหนดให้บิตนี้เป็น “1” ถ้ากำหนดให้เป็น “0” จะเป็นการปิดเคอร์เซอร์หรือไม่แสดงเคอร์เซอร์

บิต B ใช้ควบคุมการกระพริบของเคอร์เซอร์ ถ้าบิตนี้เป็น “1” เคอร์เซอร์จะกระพริบ

ดังนั้นจะมีข้อมูลคำสั่งได้ตั้งแต่ 08H-0FH(8รูปแบบคำสั่ง) ที่ใช้บ่อยคือ 0CH เป็นการสั่งให้เปิดจอแสดงผล แต่ไม่ต้องแสดงเคอร์เซอร์ และ 0FH เป็นการสั่งให้เปิดจอแสดงผล แสดงเคอร์เซอร์ และสั่งให้เคอร์เซอร์กระพริบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. คำสั่งควบคุมการเลื่อนเคอร์เซอร์และข้อมูลตัวอักษร

มีรายละเอียดของข้อมูลคำสั่งดังนี้

บิต7	บิต6	บิต5	บิต4	บิต3	บิต2	บิต1	บิต0
0	0	0	1	S/C	R/L	*	*

การควบคุมการเลื่อนเคอร์เซอร์และตัวอักษรบนจอแสดงผล ขึ้นอยู่กับการกำหนดบิต S/C และ R/L ซึ่งสามารถสรุปได้ดังนี้

S/C	R/L	ลักษะการเลื่อน	ข้อมูลคำสั่ง
0	0	เลื่อนเคอร์เซอร์ไปทางซ้าย	10H-13H
0	1	เลื่อนเคอร์เซอร์ไปทางขวา	14H-17H
1	0	เลื่อนตัวอักษรใหม่ไปทางซ้าย	18H-1BH
1	1	เลื่อนตัวอักษรใหม่ไปทางขวา	1CH-1FH

6.คำสั่งกำหนดฟังชั่นการทำงาน

มีรายละเอียดของรูปแบบข้อมูลคำสั่งดังนี้

บิต7	บิต6	บิต5	บิต4	บิต3	บิต2	บิต1	บิต0
0	0	1	DL	N	F	*	*

บิต DL ใช้กำหนดจำนวนบิตที่ใช้ติดต่อส่งผ่านข้อมูล ถ้าบิตนี้เป็น “0” จะเป็นการติดต่อแบบ 4 บิต แต่ถ้าเป็นหนึ่งจะเป็นแบบ 8 บิต

บิต N ใช้กำหนดจำนวนบรรทัดของการแสดงผล ถ้าเป็น “0”จะแสดงผล 1 บรรทัด ถ้าเป็น “1” จะแสดง 2 บรรทัด ในกรณีที่จอแสดงผลแสดงมากกว่า 2 บรรทัด และต้องการให้แสดงผลมากกว่า 2 บรรทัด แต่ในกรณีของ โมดูล LCD แบบ 16 ตัวอักษร แบบ 1บรรทัด แม้จะมีบรรทัดการแสดงผลเป็น 1บรรทัดแต่ต้องกำหนดบิต N ให้เป็น “1” เนื่องจากแอสเดรส DDRAM แบ่งเป็น 2 ช่องคือ 00H และ 40H

บิต F ใช้เลือกความละเอียดของตัวอักษรในการแสดงผล ถ้าบิตนี้เป็น “0” จะแสดงผลแบบ 5x7 จุด และถ้าเป็นหนึ่งจะแสดงแบบ 5x10 จุด

ข้อมูลคำสั่งที่ใช้บ่อยคือ 38H เป็นการกำหนดให้ โมดูล LCD ทำงานแบบ 8 บิต แสดงผล 2 บรรทัดและเลือกความละเอียดเป็น 5x7 จุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.คำสั่งเลือกแอสเดรสของ CGRAM

เมื่อต้องการกำหนดแอสเดสของ CGRAM ต้องกำหนดให้บิต 7 เป็น “0” บิต 6 เป็น “1” ส่วนอีก 6 บิตที่เหลือจะแทนด้วยค่าแอสเดรสของ CGRAM จะต้องทำการกำหนดแอสเดรสด้วยคำสังนี้ ก่อนที่จะอ่านหรือเขียนข้อมูลให้ CGRAM โดยแอสเดรสของ CGRAM อยู่ระหว่าง 00H-3FH

8.คำสั่งเลือกแอสเดรสของ DDRAM

ใช้ในการเลือกแอสเดรสของ DDRAM ก่อนท่จะทำการอ่านหรือเขียนข้อมูล โดยบิต 7 ต้องเป็น “1” และข้อมูลอี 7 บิตที่เหลือจะเป็นค่าแอสเดรสของ DDRAM ซึ่งแอสเดรสของ DDRAM จะอยู่ระหว่าง 8CH-0FFH

9.คำสั่งอ่านแฟลก BUSY และแอสเดรส

มีรายละเอียดข้อมูลคำสั่งดังนี้

บิต7	บิต6	บิต5	บิต4	บิต3	บิต2	บิต1	บิต0
BF	A	A	A	A	A	A	A
	แอสเดรสไบต์สูง			แอสเดรสไบต์ต่ำ			

เป็นคำสั่งที่ใช้อ่านแฟลก BUSY(BF) โดยแฟลกนี้จะเป็นตัวบอกสถานะของตัวควบคุม LCD ว่าพร้อมจะรับข้อมูลอยู่หรือไม่ ถ้าหากบิต BF เป็น “0” แสดงว่าตัวควบคุม LCD พร้อมจะรับข้อมูลหรือคำสั่ง แต่ถ้าเป็น “1” แสดงว่าขณะนี้ตัวควบคุม LCD ยังอยู่ในกระบวนการทำงานภายในหรือกำลังประมวลผลข้อมูลอยู่ ยังไม่พร้อมรับข้อมูลหรือคำสั่ง เมื่อต้องการอ่านแฟลกต้องกำหนดให้ขา R/W เป็น “1” ด้วย แต่สัญญาณที่ RS ยังต้องเป็น “0” อยู่เพราะข้อมูลนี้เป็นชุดคำสั่ง นอกจากนี้ยังใช้เป็นยังใช้เป็นคำสั่งอ่านข้อมูลแอสเดรสของ CGRAM และ DDRAM ด้วย โดยบิต 0 – 6 เป็นค่าของข้อมูลแอสเดรสที่ต้องการอ่าน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเขียนคำสั่งและข้อมูลให้แก่โมดูล LCD

ในการเขียนข้อมูลเพื่อควบคุมให้โมดูล LCD แสดงผลตามที่ผู้ใช้งานต้องการ ต้องส่งคำสั่งแล้วกำหนดโหมดการทำงานให้แก่โมดูล LCD ก่อน จากนั้นจึงค่อยส่งข้อมูลที่ต้องการแสดงผล เนื่องจากบัสข้อมูลของโมดูล LCD มี 8 เส้น คือ D0-D7 และใช้เป็นทางผ่านของทั้งคำสั่งและข้อมูล ดังนั้นในการส่งคำสั่ง ดังนั้นในการส่งคำสั่งและข้อมูลจึงต้องอาศัยการกำหนดสัญญาณลอจิกที่ขา RS ถ้าหากที่ขา RS ได้ลอจิก “0” หมายความว่าข้อมูลที่ป้อนให้แก่โมดูล LCD ขณะนั้นเป็นคำสั่ง ในทางตรงกันข้าม หากขา RS ได้รับลอจิก “1” ข้อมูลที่ป้อนให้ขณะนั้นเป็นข้อมูลที่ใช้ในการแสดงผล

เมื่อต้องการเขียนหรืออ่านข้อมูลใน CGRAM และ DDRAM เริ่มต้นต้องกำหนดแอสเดรสที่ต้องการอ่าน หรือเขียนก่อน โดยใช้คำสั่งเลือกแอสเดรส จากนั้นกำหนดให้ขา RS เป็น “1”เพื่อแจ้งให้ตัวควบคุมภายในโมดูล LCD ทราบว่าข้อมูลข้อมูลที่ปรากฏต่อไปนี้เป็นข้อมูลปกติไม่ใช่คำสั่ง

ในกรณีที่ต้องการอ่านข้อมูลต้องกำหนดให้ขา R/W เป็น “1” ข้อมูลขนาด 8 บิต(หรือ 4 บิต) ก็จะปรากฏบนบัสข้อมูล ดยข้อมูลท่อ่านได้เป็นข้อมูลจากแอสเดรสของ CGRAM หรือ DDRAM ตามที่ต้องการ

ในกรณีที่ต้องการเขียนข้อมูล เมื่อกำหนดแอสเดรสและป้อนลอจิก “1” ให้ขา RS แล้ว ต้องกำหนดให้ขา R/W เป็น “0” ข้อมูลที่อยู่บนบัสข้อมูลจะถูกเขียนลงในรีจิสเตอร์ DR จากนั้นจึงถ่ายทอดลงใน DDRAM ต่อไป

จังหวะการทำงานของโมดูล LCD

ในการติดต่อกับโมดูล LCD จะต้องมีการหน่วงเวลาหลังจากที่มีการส่งรหัสคำสั่งหรือข้อมูล เนื่องจากต้องรอให้คอนโทรลเลอร์ภายในโมดูล LCD แปลความหมายของรหัสคำสั่งและทำงานตามคำสั่งให้เรียบร้อยก่อนจากนั้นจึงจะรับข้อมูลดำเนินการต่อไป

ดังนั้น ในการใช้งานโมดูล LCD ต้องมีโปรแกรมเพื่อหน่วงเวลารอให้โมดูล LCD พร้อมทำงานด้วย โดยเมื่อเริ่มจ่ายไฟให้กับโมดูล LCD ต้องรอประมาณ 10 มิลลิวินาที เพื่อให้โมดูล LCD ทำการเตรียมความพร้อม หลังจากนั้นก็จะกำหนดลอจิกให้แก่ขา RS ของโมดูล LCD แล้วต้องหน่วงเวลาอีกประมาณ 2 มิลลิวินาทีเพื่อให้คอนโทรลเลอร์ในโมดูล LCD แปลความหมายของลอจิกที่ขา RS ว่า ข้อมูลต่อไปที่จะได้รับเป็นรหัสคำสั่งหรือหรือข้อมูลที่ต้องการแสดงผล จากนั้นจะเป็นการส่งข้อมูลมารอที่บัสข้อมูล D0-D7(กรณีทำงานในโหมด 8 บิต)ขั้นต่อไปเป็นการส่งสัญญาณพัลส์ไปที่ขา E เพื่อเอนเนเบิลโมดูล LCD ให้รับข้อมูลจากบัสข้อมูลเข้าไป โดยพัลส์ที่ป้อนเข้าที่ขา E ของโมดูล LCD ต้องเป็นพัลส์ขอบขาขึ้น จากนั้นทำการหน่วงเวลา 2 มิลลิวินาที ทั้งหมดที่กล่าวมานี้เป็นการทำงานหนึ่งรอบของ โมดูล LCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.3 ระบบบัส I^2C

ความรู้เบื้องต้นเกี่ยวกับ I^2C

I^2C ย่อมาจาก Inter-IC Communication หมายถึง การติดต่อสื่อสารระหว่างไอซี โดยบัส I^2C ได้รับการพัฒนาขึ้นโดยฟิลิปส์(Philips) ด้วยจุดมุ่งหมายหลักคือ ต้องการให้ไอซีหรือโมดูลสามารถติดต่อ สั่งงานและควบคุมโดยใช้สายสัญญาณเพียง 2 เส้น สายหนึ่งคือ สายข้อมูล อีกเส้นหนึ่งคือสายสัญญาณนาฬิกาที่ใช้ในการกำหนดจังหวะการทำงาน การต่อร่วมกันของอุปกรณ์บนบัส I^2C ทำได้ง่ายมาก เพียงต่อสายข้อมูลและสายสัญญาณนาฬิกา ของอุปกรณ์แต่ละตัวขนานหรือพ่วงกันไป ส่วนการกำหนดแอสเดรสหรือตำแหน่งสำหรับติดต่ออุปกรณ์แต่ละตัว จะใช้รหัสข้อมูลและการกำหนดสภาวะลอจิกที่ขาแอสเดรสของอุปกรณ์แต่ละตัว

สายข้อมูลบน I^2C มีชื่อเรียกอย่างเป็นทางการว่า สายข้อมูลอนุกรมหรือ SDA(Serial Data Line) ส่วนสายสัญญาณนาฬิกามีชื่อเรียกว่า สายสัญญาณนาฬิกาอนุกรมหรือ SCL(Serial Clock Line)

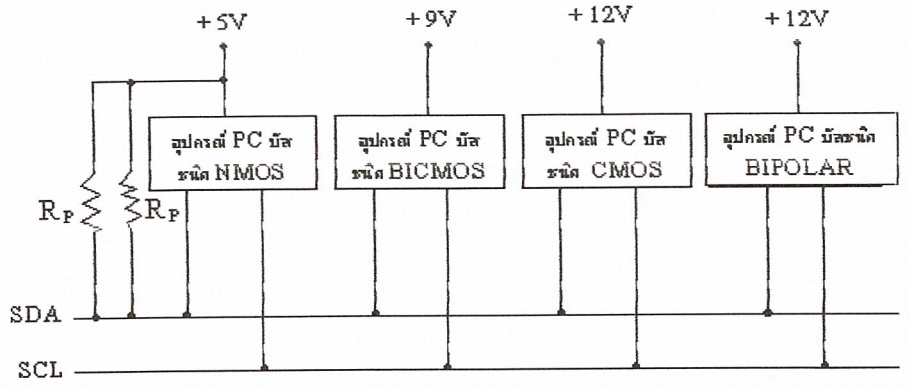
คุณสมบัติโดยทั่วไปของบัส I^2C

สาย SDA และ SCL เป็นสายสัญญาณ 2 ทิศทาง ต้องมีการต่อตัวต้านทานพูลอัปกับแรงดัน +5V ไว้ตลอดเวลา เพื่อให้สายมีสถานะลอจิกสูง ในขณะที่ไม่มีการติดต่อใช้งาน ทั้งยังช่วยป้องกันสัญญาณรบกวนที่อาจมีเข้ามาในสายสัญญาณทั้งสอง

อัตราการถ่ายทอดข้อมูลบนบัส I^2C สูงถึง 100 กิโลบิตต่อวินาทีในโหมดปกติ และสูงถึง 400 กิโลบิตต่อวินาทีในโหมดความเร็วสูง อุปกรณ์ที่ต่อร่วมอยู่บนบัส I^2C จะต้องมีค่าความจุไฟฟ้ารวมที่เกิดขึ้นระหว่างสาย SDA และ SCL ไม่เกิน 400 pF การเข้าถึงอุปกรณ์บนบัส I^2C ใช้ข้อมูลสำหรับการเข้าถึง 2 ค่าคือ 7 บิต หรือ 10 บิต

ข้อเด่นอีกประการหนึ่งของการเชื่อมต่อแบบบัส I^2C คือสามารถเชื่อมต่ออุปกรณ์ที่ใช้ไฟเลี้ยงไม่เท่ากันให้สามารถติดต่อสื่อสารถึงกันได้ โดยอุปกรณ์บนบัส I^2C ตัวหนึ่งอาจใช้ไฟเลี้ยง +5 V ในขณะที่อีกตัวหนึ่งใช้ไฟเลี้ยง +12 V การต่อร่วมกันบนบัส I^2C สามารถกระทำได้เหมือนกับในกรณีที่อุปกรณ์ทั้งสองมีไฟเลี้ยงเท่ากัน กล่าวคือ ให้ต่อสาย SDA และ SCL ของอุปกรณ์แต่ละตัวเข้าด้วยกัน และต้องต่อตัวต้านทานพูลอัป เข้ากับแรงดัน +5 V ไว้ด้วยเสมอ ดังแสดงในรูปที่ 2.7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.7 การต่อตัวต้านทานพูลอัปบนสายสัญญาณในระบบบัส I^2C

<u>หลักการของบัส I^2C</u>

บัส I^2C ประกอบด้วยสายสัญญาณ 2 เส้น ดังที่ได้กล่าวมาแล้ว คือ SDA และ SCL อุปกรณ์ที่ต่อพ่วงบนบัสสามารถมีได้มากมาย ดังนั้นจึงต้องมีการกำหนดรูปแบบของการติดต่อบนบัส หรือเรียกว่า โปรโตคอล เพื่อให้ผู้ใช้งานทราบว่า ขณะนี้อุปกรณ์ใดติดต่อกันอยู่ และอุปกรณ์ใดเป็นตัวรับหรือตัวส่ง

ลักษณะหน้าที่ และ นิยามของอุปกรณ์ที่ต่ออยู่บนบัส I^2C

อุปกรณ์ที่เป็นผู้สร้างข้อมูลหรือส่งข้อมูล เรียกว่า ตัวส่ง(transmitter)

อุปกรณ์ที่เป็นผู้รับข้อมูล เรียกว่า ตัวรับ อุปกรณ์บนบัส I^2C สามารถเป็นได้ทั้งตัวรับหรือตัวส่ง บางอุปกรณ์ทำหน้าที่เป็นตัวรับอย่างเดียว จะไม่มีอุปกรณ์ใดบนบัส I^2C ที่ทำหน้าที่เป็นตัวส่งเพียงอย่างเดียว

อุปกรณ์ที่ทำหน้าที่ควบคุมจังหวะในการติดต่อบนบัส I^2C เรียกว่า มาสเตอร์

อุปกรณ์ที่ถูกควบคุมหรืออุปกรณ์ที่ต่อพ่วงเข้าไปบนบัส I^2C เรียกว่า สเลฟ

ข้อกำหนด 2 ประการสำคัญของการติดต่อบนบัส I^2C คือ

1. การถ่ายทอดข้อมูลจะเกิดขึ้นได้เมื่อบัสว่างเท่านั้น

2. ในระหว่างการถ่ายทอดข้อมูล เมื่อใดก็ตามที่สาย SCL มีสถานะเป็นลอจิกสูง สายข้อมูลต้องรักษาข้อมูลไว้ อย่าให้เกิดการเปลี่ยนแปลงขึ้นเด็ดขาด มิฉะนั้น สัญญาณที่เกิดขึ้นจะได้รับการแปลความหมายเป็นสัญญาณควบคุมแทน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สภาวะที่เกิดขึ้นบนบัส I^2C

มี 5 สภาวะดังนี้

1. บัสว่าง(Bus not busy) สภาวะนี้เกิดขึ้นเมื่อสถานะลอจิกบนสาย SDA และ SCL เป็นลอจิกสูงทั้งคู่ นั่นหมายความว่า การถ่ายทอดข้อมูลสามารถเริ่มต้นขึ้นได้

2. เริ่มต้นการถ่ายทอดข้อมูล(start data transfer) เกิดขึ้นเมื่อสาย SDA มีการเปลี่ยนแปลงลอจิกจากสูงไปต่ำ ในขณะที่สาย SCL มีสถาวะลอจิกสูง เรียกสภาวะที่เกิดขึ้นนี้ว่า สภาวะเริ่มต้น (start)

3. หยุดการถ่ายทอดข้อมูล(stop data transfer) เกิดขึ้นเมื่อสาย SDA มีการเปลี่ยนแปลงสถานะลอจิก จากต่ำไปสูง ในขณะที่สาย SCL มีสถานะลอจิกสูง เรียกสภาวะที่เกิดขึ้นนี้ว่า สภาวะหยุด(stop)

4. ข้อมูลดำรงอยู่บนบัส(data valid) สภาวะนี้เกิดขึ้นถัดจากสภาวะเริ่มต้น โดยสถานะที่ลอจิกที่เกิดบนสาย SDA ก็คือข้อมูลที่ทำการถ่ายทอด เมื่อสาย SCL เป็นลอจิกสูง สถานะที่สาย SDA ต้องคงที่ เพื่อให้อุปกรณ์รับรู้ข้อมูลในขณะนั้นว่า เป็น "0" หรือ "1" ข้อมูลอาจเกิดการเปลี่ยนแปลงได้ในขณะที่สาย SCL เป็นลอจิกต่ำ แต่เมื่อใดก็ตามที่ต้องการให้เกิดการถ่ายทอดข้อมูลที่สมบูรณ์ สถานะลอจิกที่สาย SDA ต้องคงที่ตลอดช่วงเวลาที่สาย SCL มีสถานะลอจิกสูง หากเกิดการเปลี่ยนแปลงสถานะลอจิกในขณะที่สาย SCL มีลอจิกสูงอยู่นั้น อุปกรณ์มาสเตอร์ที่ทำการควบคุมการถ่ายทอดข้อมูลจะแปลความหมายเป็นสภาวะหยุดหรือสภาวะเริ่มต้นก็ได้ ทำให้ข้อมูลที่ทำการถ่ายทอดนั้นเกิดความผิดพลาดขึ้น

5. รับรู้ข้อมูล(acknowledge) เกิดขึ้นหลังจากที่การถ่ายทอดข้อมูลจากตัวส่งมายังตัวรับเกิดขึ้นอย่างสมบูรณ์ โดยตัวส่งจะทำการส่งข้อมูลมา 1 บิต เรียกว่า บิตรับรู้ (acknowledge bit) มีสถานะเป็นลอจิกสูง หลังจากส่งข้อมูลมาครบถ้วน ส่วนอุปกรณ์มาสเตอร์จะทำการส่งสัญญาณรับรู้พิเศษ ซึ่งสัมพันธ์กับสัญญาณนาฬิกาเพื่อตอบสนองบิตรับรู้ที่ส่งมาจากตัวส่ง ทางด้านตัวรับจะส่งบิตรับรู้ที่มีสถานะลอจิกต่ำลงบนบัส อุปกรณ์เสลฟที่ถูกอ้างถึงในการติดต่อหรือกำลังติดต่ออยู่ในขณะนั้นก็จะกำเนิดบิทรับรู้เพื่อตอบสนองให้ทราบว่าได้รับข้อมูลในแต่ละไบต์เรียบร้อยแล้ว

การทำงานบนบัส I^2C

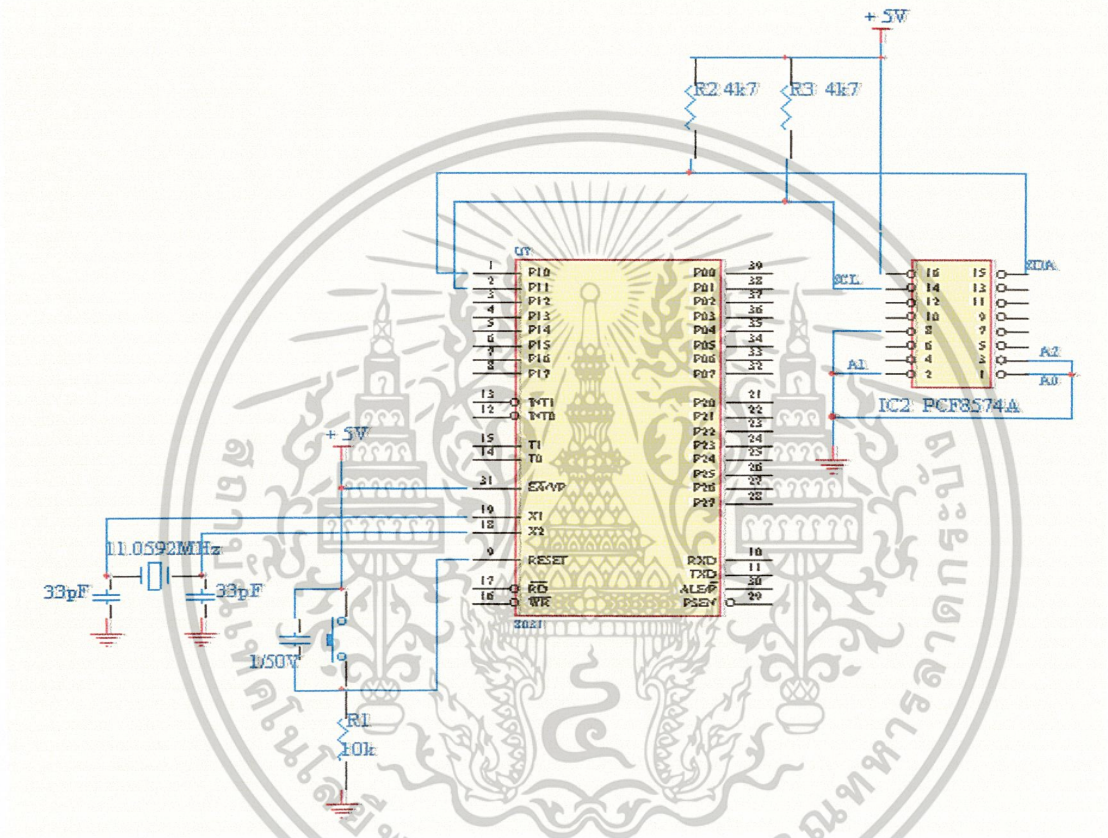
ก่อนเริ่มต้นการถ่ายทอดข้อมูลระหว่างอุปกรณ์ต่างๆที่อยู่บนบัส ต้องมีการอ้างถึงก่น โดยการอ้างถึงอุปกรณ์บนบัส I^2C นั้นจะใช้การอ้างถึงแบบ 7 บิต หรือ 10 บิต ในกรณีที่อุปกรณ์ต่ออยู่บนบัสไม่มาก ใช้การอ้างถึงแบบ 7 บิต ก็เพียงพอ แต่ถ้ามีอุปกรณ์มากกว่า 127 แอสเดรส จำเป็นต้องใช้การอ้างถึงแบบ 10 บิต หลังจากที่ติดต่ออุปกรณ์แต่ละตัวได้เรียบร้อยแล้ว ก็จะเริ่มต้นถ่ายทอดข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การต่ออุปกรณ์ระบบบัส I^2C กับไมโครคอนโทรลเลอร์ MCS-51

สามารถทำได้ง่ายมาก เพียงใช้ขาพอร์ต 2 ขา โดยกำหนดให้ขาหนึ่งเป็น SDA อีกขาหนึ่งเป็น SCL และต่อตัวต้านทานค่าประมาณ 4.7k พูลอัปที่ขาพอร์ตทั้งสองขา เพียงเท่านี้ก็สามารถติดต่อกับอุปกรณ์ระบบบัส I^2C ได้แล้ว

ในรูปที่ 2.8 เป็นวงจรตัวอย่างการเชื่อมต่อไมโครคอนโทรลเลอร์ MCS-51 เข้ากับระบบบัส I^2C จากวงจรจะใช้ขาพอร์ต P1.0 เป็นขา SDA และ P1.1 เป็นขา SCL



รูปที่ 2.8 วงจรตัวอย่างการต่อไมโครคอนโทรลเลอร์ MCS-51 กับอุปกรณ์ระบบ I^2C

การเขียนโปรแกรมติดต่อบัส I^2C

เริ่มต้นด้วยการสร้างสภาวะมาตรฐานของบัส I^2C อันประกอบด้วย สภาวะเริ่มต้น, สภาวะสิ้นสุดการส่งข้อมูล ,สภาวะหยุด , สัญญาณนาฬิกาบนขา SCL การเขียนและอ่านข้อมูลกับอุปกรณ์บนระบบบัส I^2C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<u>การสร้างสภาวะเริ่มต้น</u>

1.เมื่อต้องการติดต่อกับบัส I^2C สิ่งแรกที่ต้องทำสำหรับไมโครคอนโทรลเลอร์ซึ่งถือว่าเป็นอุปกรณ์มาสเตอร์ คือ การทำให้บัสว่างด้วยการกำหนดให้ขา SCL และขา SDA มีลอจิกเป็นหนึ่งทั้งคู่

2.จากนั้นทำให้ขา SDA เป็น “0” โดยที่ขา SCL ยังเป็น “1” อยู่

3.กำหนดให้ขา SCL มีลอจิกเป็น “0” ถึงตอนนี้ทั้ง SCL และ SDA จะมีลอจิกเป็น “0” ทั้งคู่พร้อมที่จะติดต่อได้แล้ว

<u>การสร้างสภาวะหยุด</u>

1.เมื่อต้องการหยุดส่งข้อมูลจะต้องส่งสภาวะหยุดออกไป โดยในตอนแรกต้องกำหนดให้ขา SCL และ SDA เป็นลอจิก “0” ทั้งคู่ก่อน

2.กำหนดให้ขา SCL มีลอจิกเป็น “1” โดย SDA ยังคงมีลอจิกเป็น “0”

3.จากนั้นทำให้ขา SDA เป็น “1” ซึ่งจะทำให้ระบบบัสกลับเข้าสู่บัสว่างอีกครั้ง พร้อมที่จะรับข้อมูลต่อไป

<u>การส่งข้อมูลลอจิก “0” และลอจิก “1”</u>

หลังจากที่ทำการส่งบิตเริ่มต้นแล้ว ลำดับต่อไปคือ จะต้องส่งข้อมูลควบคุมซึ่งจะเป็นขบวนการของลอจิก “0” และ ลอจิก “1”

-สำหรับการส่งข้อมูลลอจิก “0”ต้องดำเนินการดังนี้

1.ทำให้ขา SDA เป็น “0” สำหรับส่งข้อมูลลอจิก “0”

2.ทำให้ขา SCL เป็น “1” สำหรับการส่งสัญญาณนาฬิกา ในขณะที่ขา SDA ยังคงเป็น “0” อยู่

3.จากนั้นทำให้ขา SCL กลับมามีสถานะเป็นลอจิก “0” เหมือนเดิม

-สำหรับการส่งข้อมูลลอจิก “1” มีขั้นตอนดังนี้

1.ทำให้ขา SDA มีลอจิกเป็น “1”

2.ทำให้ขา SCL เป็น “1” สำหรับการส่งสัญญาณนาฬิกา ในขณะที่ SDA ยังคงเป็น “1” อยู่

3.จากนั้นทำให้ขา SCL กลับมามีสถานะเป็นลอจิก “0” เหมือนเดิม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

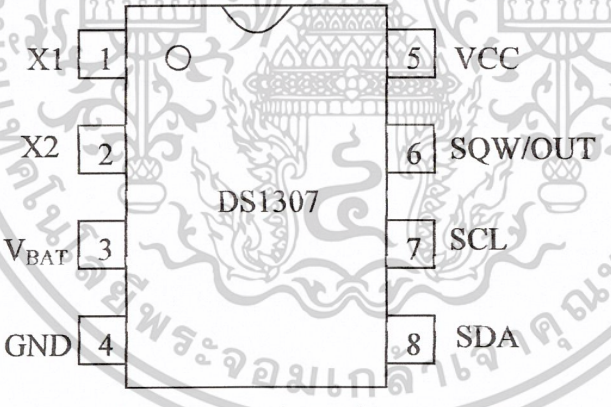
2.2.4 การใช้งานไอซีสร้างฐานเวลาจริงหรือรีลไทม์คล็อก(RTC)กับ MCS-51

แนะนำ DS1307 ไอซีสร้างฐานเวลาจริงหรือรีลไทม์คล็อก(RTC)

ผู้ผลิตคือ ดัลลัสเซมิคอนดัคเตอร์(Dallas semiconductor) มีหน้าที่สร้างฐานเวลาจริงให้แก่ระบบไมโครคอนโทรลเลอร์ โดย DS1307 จะให้ข้อมูลเกี่ยวกับวันเวลาทั้งหมด ไม่ว่าจะเป็นค่าของเวลาที่ละเอียดถึงหลักวินาที,นาที,ชั่วโมง,วันที่(date),วันในสัปดาห์(day),เดือน และ ปี โดยสามารถปรับวันเดือนปีให้ตรงตามปฏิทินได้อย่างถูกต้อง รวมทั้งกำหนดวันในปีอธิกสุรทินด้วย คุณสมบัติทางเทคนิคที่สำคัญมีดังนี้

- เป็นไอซีรีลไทม์คล็อกให้ข้อมูลตั้งแต่วินาทีจนถึงปี สามารถให้ข้อมูลเวลาได้อย่างเที่ยงตรงถึงปีคริสตศักราช 2100
- มีหน่วยความจำแรม 56 ไบต์อยู่ภายในสามารถใช้เก็บข้อมูลทั่วไปได้
- ใช้การเชื่อมต่อแบบระบบบัส I^2C
- มีวงจรตรวจจับไฟเลี้ยงต่ำหรือหายไปอย่างอัตโนมัติและสามารถรักษาข้อมูลเวลาไว้ได้แม้ไม่มีไฟเลี้ยงไอซี

รายละเอียดของขาของ DS1307(RTC)



รูปที่ 2.9 การจัดขาของไอซี DS1307 ไอซีสร้างฐานเวลาจริง (RTC)

ในรูปที่ 2.9 แสดงการจัดขาของ DS1307แต่ละขามีหน้าที่การและใช้งานดังนี้

V_{CC} GND (ขา 8, 4) ต่อกับไฟเลี้ยง +5V

V_{BAT} (ขา 3) ใช้ต่อกับแบตเตอร์รี่ 3V เพื่อรักษาการทำงานของวงจรสร้างฐานเวลาของ DS1307 ให้คงอยู่ต่อไปแม้ว่าไม่มีไฟเลี้ยงจ่ายให้แก่ DS1307 ชนิดของแบตเตอร์รี่ที่เหมาะสมคือแบบเตอร์รี่แบบลิเธียม ซึ่งมีความจุ 40mAhr หรือมากกว่าจะสามารถรักษาข้อมูลได้นาน 10 ปีที่อุณหภูมิ 25 องศาเซลเซียส

SDA, SCL (ขา 5, 6) เป็นขาสำหรับเชื่อมต่อกับไมโครคอนโทรเลอร์บนระบบ I^2C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$\overline{\text{SQW OUT}}$ (ขา 7) ที่ขานี้จะมีสัญญาณรูปสี่เหลี่ยมส่งออกมา โดยสามารถเลือกความถี่ได้ 1Hz, 4.096kHZ, 8.192kHz และ 32kHz ในการใช้งานต้องต่อตัวต้านทาน 1k พูลอัปที่ขานี้ด้วย

X1, X2 (ขา1 และ 2) ใช้ต่อกับคลิสตอลความถี่มาตราฐาน 32.768kHz เพื่อใช้เป็นฐานเวลาในการสร้างค่าเวลาจริง ในการใช้งานต้องต่อคลิสตอลเข้ากับขาทั้งสองนี้ด้วยและที่แต่ละขาต้องต่อตัวเก็บปรุจุค่าต่ำๆ ประมาณ 15pF คร่อมกับขากราวด์ด้วย

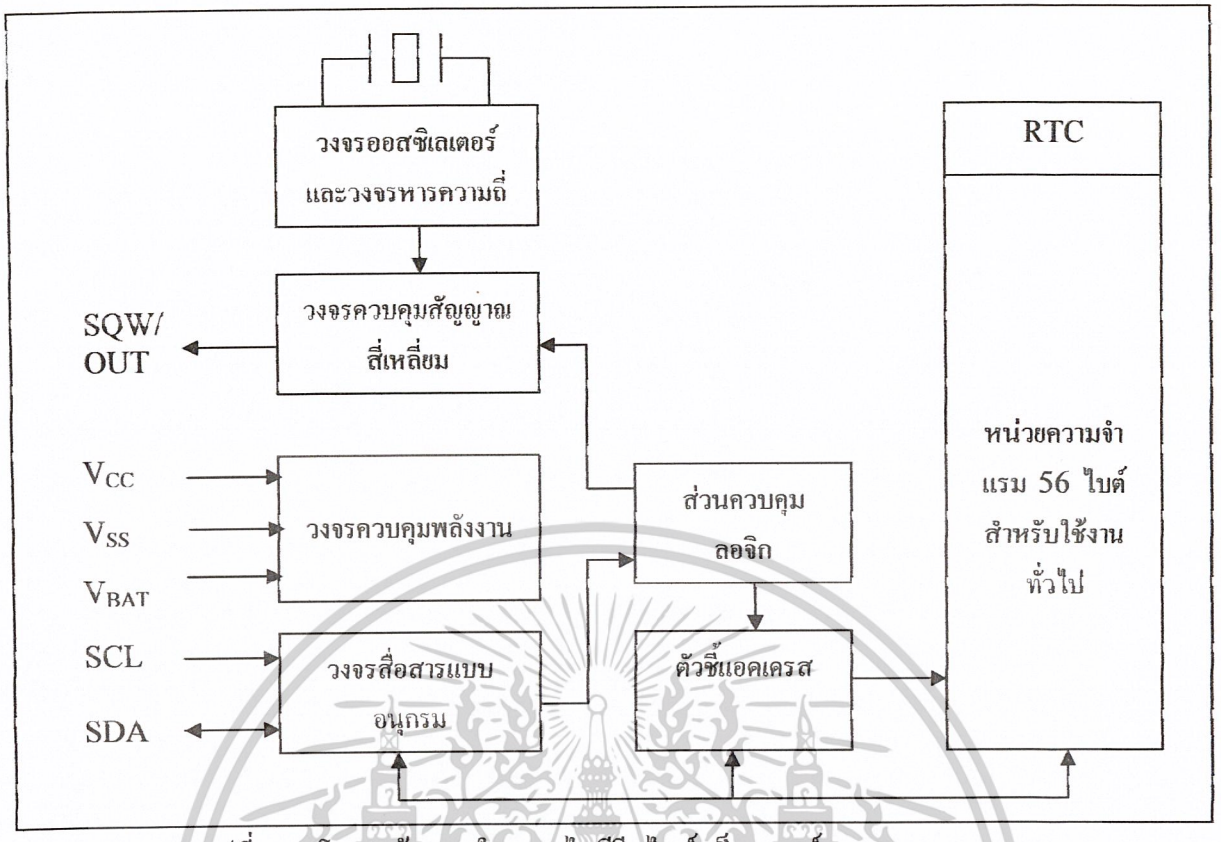
การทำงานของ DS1307

ในไอซี DS1307 จัดการเชื่อมต่อในระบบบัส I^2C โดยจะทำงานเป็นอุปกรณ์สเลฟเสมอ ดังนั้นการติดต่อเพื่อใช้งานจึงต้องกำหนดรูปแบบตามที่กำหนดไว้ในการติดต่อแบบ I^2C ในรูปที่ 2.10 แสดงส่วนประกอบหลักที่สำคัญและไดอะแกรมการทำงานของ DS1307 วงจรออสซิเลเตอร์ถือเป็นหัวใจหลักของไอซีเนื่องจากเป็นจุดเริ่มต้นของการสร้างข้อมูลเวลาจริง ในขณะที่ DS1307 ทำงานที่ขา SQW/OUT จะมีสัญญาณพัลส์สี่เหลี่ยมส่งออกมาตลอดเวลาในกรณีที่มีการอีนาเบิลวงจรกำเนิดสัญญาณพัลส์ที่รีจิสเตอร์ควบคุมค่าความถี่ของสัญญาณที่สามารถเลือกได้ 4 ค่าคือ 1Hz, 4.096kHz, 8.192kHz และ32kHz พร้อมกันนั้นก็จะมีการเก็บค่าของเวลาไว้ในหน่วยความจำนอนโวลาไทม์แรม ซึ่งมีขนาดรวม 64 ไบต์ แต่จัดสรรให้ใช้เก็บข้อมูลเวลา 8 ไบต์ และเป็นหน่วยความจำสำหรับเก็บข้อมูลทั่วไปสำหรับผู้ใช้งานอีก 56 ไบต์

วงจรควบคุมพลังงานไฟฟ้าจะคอยตรวจสอบสภาวะของไฟเลี้ยงไอซี หากไฟเลี้ยงต่ำกว่า $1.25xV_{BAT}$ หรือประมาณ 3.75V ในกรณีที่ใช้ V_{BAT} เท่ากับ 3V ถ้าหากไฟเลี้ยงมีค่าต่ำกว่า V_{BAT} ไอซี DS1307 จะเข้าสู่โหมดสำรองข้อมูลกระแสต่ำทันที จะไม่มีการส่งสัญญาณพัลส์ออกมาที่ขา SQW/OUT แต่วงจรสร้างฐานเวลายังคงทำงานเพื่อให้ค่าของเวลาเดินไปอย่างไม่ผิดพลาด เมื่อมีไฟเลี้ยงปรากฏขึ้นอีกครั้ง DS1307 ก็จะสามารถให้ค่าของเวลาที่เป็นจริงแก่ผู้ใช้งานได้ต่อไป

วงจรสื่อสารอนุกรมภายใน DS1307 ได้รับการกำหนดการทำงานตามรูปแบบของบัส I^2C เป็นช่องทางการสื่อสารระหว่าง DS1307 กับอุปกรณ์มาสเตอร์ ผู้ใช้งานสามารถเข้าถึงหน่วยความจำที่ใช้เก็บค่าเวลาและหน่วยความจำใช้งานทั่วไปได้โดยการเขียนข้อมูลตามรูปแบบที่กำหนดในระบบบัส I^2C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.10 โครงสร้างภายในของ ไอซีรีลไทม์คล็อกเบอร์ DS1307

การจัดสรรหน่วยความจำใน DS1307

ในรูปที่ 2.11(ก) แสดงการจัดสรรพื้นที่ของหน่วยความจำภายใน DS1307 พื้นที่ 7 ไบต์แรกตั้งแต่แอดเดรส 00H-06H เป็นพื้นที่ของรีจิสเตอร์ค่าเวลาใช้เก็บข้อมูลเกี่ยวกับเวลา ไบต์ต่อมาที่แอดเดรส 07H เป็นพื้นที่ของรีจิสเตอร์ควบคุมการทำงานของ DS1307 ในรูปที่ 2.11 (ข) แสดงรายละเอียดของรีจิสเตอร์ค่าเวลา และรีจิสเตอร์ควบคุมของ DS1307

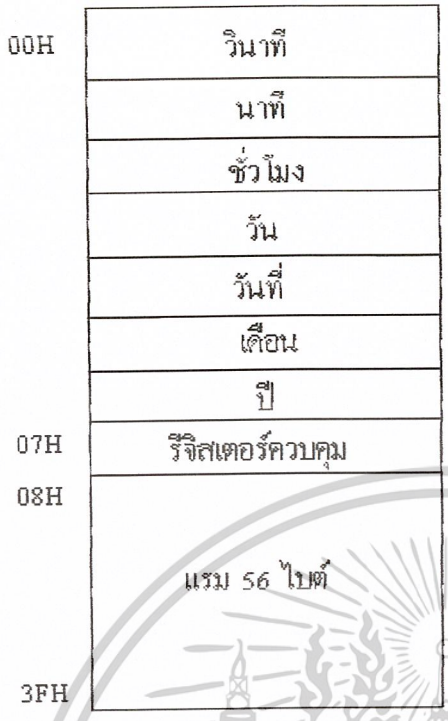
ด้วยการจัดสรรพื้นที่แบบนี้ทำให้ผู้ใช้งานสามารถเรียกข้อมูลเวลาออกมาได้ตามที่ต้องการ โดยไม่จำเป็นต้องอ่านออกมาทั้งหมดก็ได้ ค่าเวลาทั้งหมดจะอยู่ในรูปฐานสิบ สำหรับการแสดงเวลาในรูปของชั่วโมงสามารถเลือกได้ว่าต้องการแบบ 12 หรือ 24 ชั่วโมง โดยกำหนดที่บิต 6 แอดเดรส 02H และเมื่อเลือกแบบ 12 ชั่วโมง ที่บิต 5 ในแอดเดรสเดียวกันจะใช้ในการแสดงค่า AM/PM โดยถ้าบิตนี้เป็น “1” หมายถึงค่าเวลาในขณะนั้นเป็นช่วงเวลาหลังเที่ยงวัน ในกรณีที่เป็นแบบ 24 ชั่วโมง บิตนี้จะใช้ในการแสดงค่า 2 ของหลักสิบในหน่วยชั่วโมง

รีจิสเตอร์ควบคุม

มีแอดเดรสอยู่ที่ 07H มีรายละเอียดของแต่ละบิตดังนี้

OUT (Output control) : ใช้ในการควบคุมระดับลอจิกที่ขา $\overline{\text{SQW OUT}}$ ในกรณีที่ดิสเอเบิลการกำหนดสัญญาณสี่เหลี่ยมโดยถ้าบิตนี้เป็น “1” ที่ขา $\overline{\text{SQW OUT}}$ ก็จะเป็น “1” ถ้าบิตนี้เป็น “0” ที่ขา $\overline{\text{SQW OUT}}$ ก็จะเป็น “0”

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(ก)

	บิต 7	บิต 6	บิต 5	บิต 4	บิต 3	บิต 2	บิต 1	บิต 0	ค่าของข้อมูล
CH	ข้อมูลวินาที (หลักสิบ)				ข้อมูลวินาทีหลักหน่วย				00-59
X	ข้อมูลนาที (หลักสิบ)				ข้อมูลนาทีหลักหน่วย				00-59
X	12 ชั่วโมง 24 ชั่วโมง	ชั่วโมง (หลักสิบ)		ข้อมูลชั่วโมง (หลักสิบ)	ข้อมูลชั่วโมงหลักหน่วย				01-12 00-23
X	X	X		X	X	ข้อมูลวันในสัปดาห์			1-7
X	X	ข้อมูลวันที่ (หลักสิบ)			ข้อมูลวันที่ (หลักหน่วย)				01-28/29 01-30 01-31
X	X	X		ข้อมูลเดือน (หลักสิบ)	ข้อมูลเดือน (หลักหน่วย)				01-12
ข้อมูลปีหลักสิบ					ข้อมูลปี (หลักหน่วย)				00-99
OUT	X	X		SQWE	X	X	RS1	RS0	

(ข)

รูปที่ 2.11 (ก) การจัดสรรหน่วยความจำแรมภายใน

(ข) รายละเอียดของรีจิสเตอร์เก็บค่าเวลาและรีจิสเตอร์ควบคุมของ DS1307

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SQWE (Square Wave Enable) : ใช้ในการเอ็นเอเบิลวงจรกำเนิดสัญญาณสี่เหลี่ยมที่ขา SQW OUT ถ้าต้องการให้สัญญาณสี่เหลี่ยมออกให้กำหนดบิตนี้เป็น "1"

RS1, RS0 (Rate Select) : ใช้ในการเลือกความถี่ของสัญญาณสี่เหลี่ยมออกจากขา SQW OUT ดังรายละเอียดต่อไปนี้

RS1	RS2	ค่าความถี่ของสัญญาณสี่เหลี่ยม
0	0	1kHz
0	1	4.096kHz
1	0	8.192kHz
1	1	32.768kHz

โหมดการทำงานของ DS1307

มีด้วยกัน 2 โหมดคือ โหมดเขียนข้อมูลและโหมดอ่านข้อมูล ในการใช้งาน DS1307 ตามปกติจะใช้งานเฉพาะโหมดอ่านข้อมูลเท่านั้น เนื่องจากไมโครคอนโทรเลอร์จะติดต่อกับ DS1307 เพื่ออ่านข้อมูลของเวลาไปใช้งาน โหมดการเขียนข้อมูลจะถูกใช้งานก็ต่อเมื่อต้องการตั้งค่าเวลาใหม่และต้องการเขียนข้อมูลลงในหน่วยความจำใช้งานทั่วไป อย่างไรก็ตามเมื่อเริ่มต้นติดต่อกับ DS1307 จำเป็นอย่างยิ่งที่จะต้องเข้าสู่โหมดการเขียนข้อมูลก่อนเพื่อกำหนดแอดเดรสที่ต้องการอ่านข้อมูล จากนั้นจึงเปลี่ยนโหมดการทำงานมาเป็นโหมดการอ่านข้อมูล

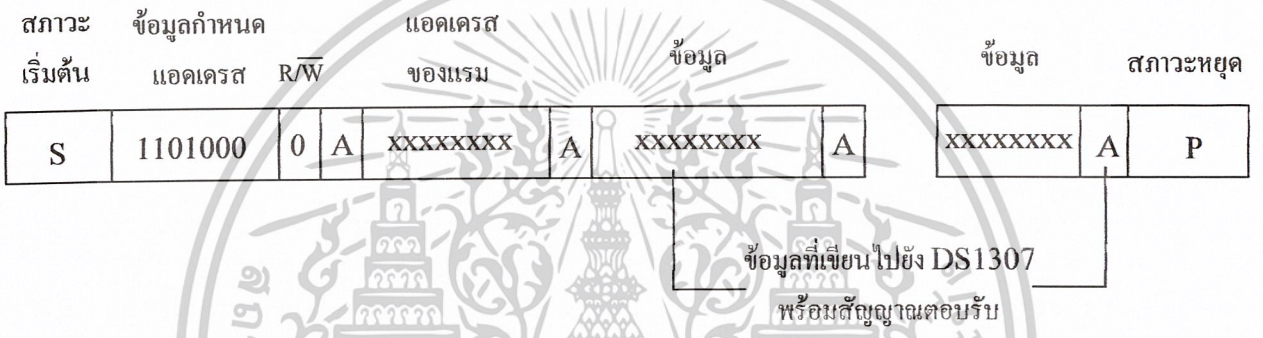
โหมดการเขียนข้อมูล

มีรูปแบบดังในรูปที่ 2.12 เริ่มต้นเมื่อไมโครคอนโทรเลอร์กำหนดสภาวะเริ่มต้น (START :S) จากนั้นส่งข้อมูลกำหนดแอดเดรส 1101000 ตามด้วยข้อมูลเลือกการเขียน นั้นคือค่า 0 จากนั้นจะรอการตอบรับจาก DS1307 ขั้นตอนต่อมาคือ ส่งข้อมูลเพื่อเลือกแอดเดรสที่ต้องการเขียน จากนั้นรอการตอบรับจาก DS1307 เมื่อมีการตอบรับมาเรียบร้อย ก็เริ่มทยอยเขียนข้อมูลลงไปครั้งละแอดเดรส หลังจากเขียนข้อมูลลงไปในแต่ละแอดเดรส จะต้องหยุดรอการตอบรับจาก DS1307 ทุกครั้ง จึงจะสามารถเขียนข้อมูลต่อไปได้ เมื่อเขียนเรียบร้อยแล้วให้ส่งสภาวะหยุด (STOP : P) เป็นอันสิ้นสุดกระบวนการเขียนข้อมูล

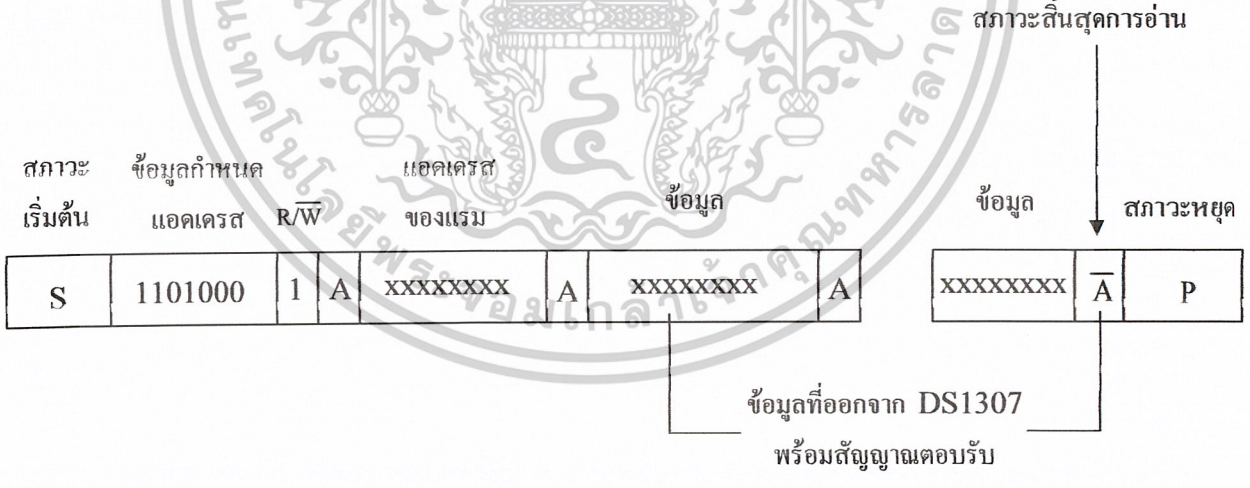
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โหมดการอ่านข้อมูล

รูปแบบดังในรูปที่ 2.13 เริ่มต้นเมื่อไมโครคอนโทรเลอร์กำหนดสภาวะเริ่มต้น (START :S) จากนั้นส่งข้อมูลกำหนดแอดเดรส ตามด้วยข้อมูลเลือกการอ่าน นั้นคือค่า 1 จากนั้นจะรอการตอบรับจาก DS1307 เมื่อรอรับเรียบร้อย DS1307 ทยอยส่งข้อมูลออกไปครั้งละแอดเดรส โดยแอดเดรสที่เลือกอ่านข้อมูลจะต้องมีการกำหนดมาก่อนล่วงหน้าด้วยโหมดการเขียนข้อมูลวิธีการง่ายๆ คือ เข้าสู่โหมดการเขียนข้อมูลก่อน เมื่อถึงจังหวะที่ต้องการเขียนข้อมูลให้ทำการสร้างสภาวะเริ่มต้นและส่งข้อมูลกำหนดแอดเดรสใหม่อีกครั้ง ตามด้วยเลือกโหมดการอ่านข้อมูล ข้อมูลที่ออกมาจาก DS1307 ก็จะเป็นข้อมูลที่แอดเดรสที่กำหนดไว้ก่อนหน้านี้



รูปที่ 2.12 รูปแบบของข้อมูลสำหรับติดต่อกับ DS1307 ในโหมดการเขียนข้อมูล

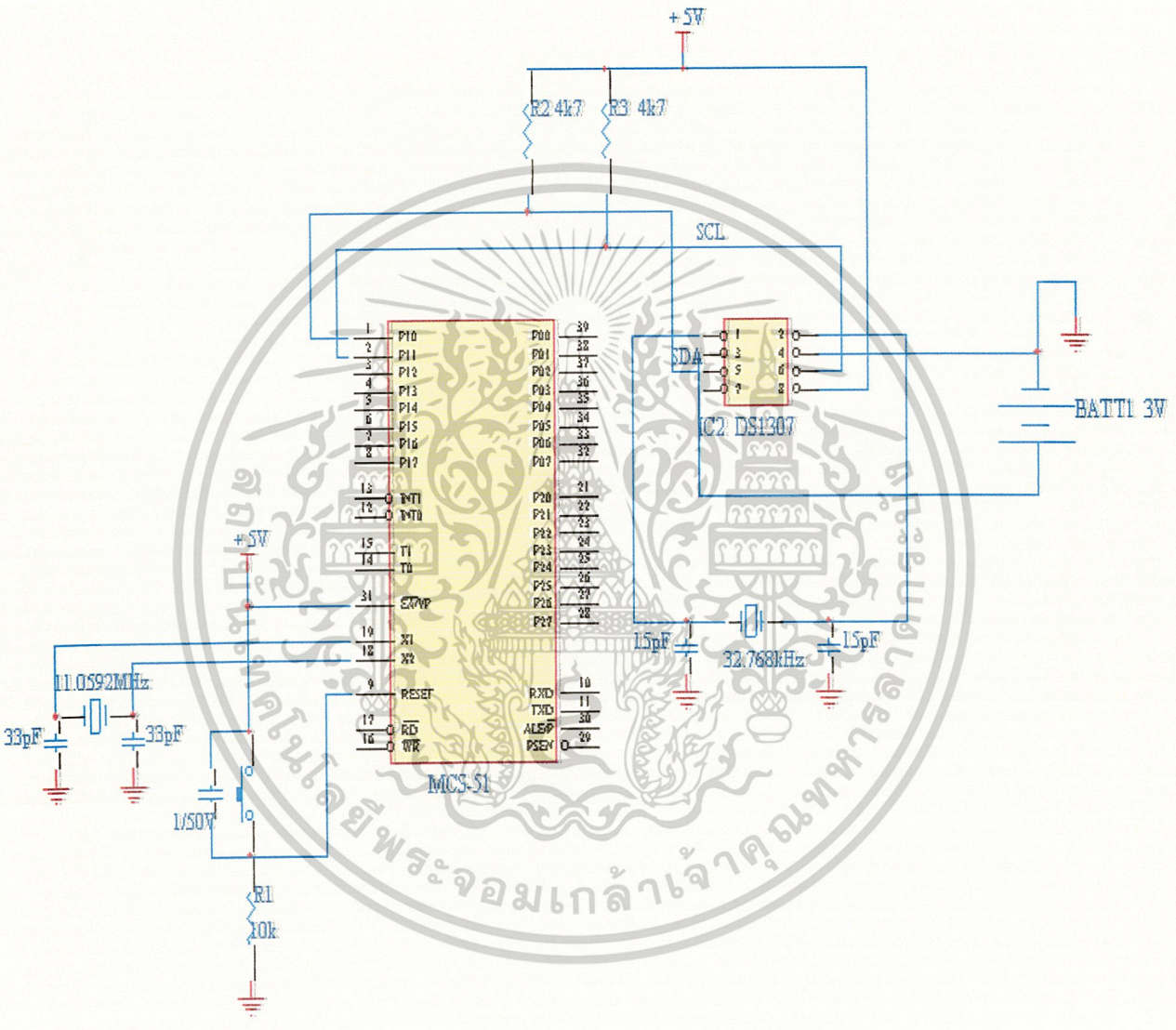


รูปที่ 2.13 รูปแบบของข้อมูลสำหรับติดต่อกับ DS1307 ในโหมดการอ่านข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเชื่อมต่อ DS1307 กับไมโครคอนโทรเลอร์ MCS-51

มีตัวอย่างวงจรแสดงดังในรูปที่ 2.14 จะเห็นว่ามีลักษณะการต่อเหมือนกับอุปกรณ์บนระบบบัส I^2C ตัวอื่นๆ ทุกประการ และสามารถที่จะส่งต่อไอซีทั้งหมดร่วมกันบนสาย SDA และSCL ได้เป็นการย้ำให้เห็นถึงความสามารถพิเศษของระบบบัส I^2C ที่สามารถเชื่อมต่ออุปกรณ์ที่มีความแตกต่างกันในหน้าที่การทำงานบนสายสัญญาณเดียวกันได้ โดยใช้สายสัญญาณเพียง 2 เส้น



รูปที่ 2.14 แสดงการเชื่อมต่อไมโครคอนโทรลเลอร์ MCS-51 กับไอซีรีลไทม์คล็อก DS1307

จากวงจรในรูปที่ 2.14 ไอซี DS1307 จำเป็นจะต้องต่อแบตเตอร์รี่ไว้ตลอดเวลาไม่ว่าจะใช้งานหรือไม่ทั้งนี้เพื่อรักษาการทำงานของวงจรภายใน DS1307 ให้ยังคงทำงานต่อเนื่องไป เมื่อใดที่ไมโครคอนโทรเลอร์ต้องการอ่านข้อมูล ก็จะได้ข้อมูลเวลาที่เป็นจริงตลอดเวลา

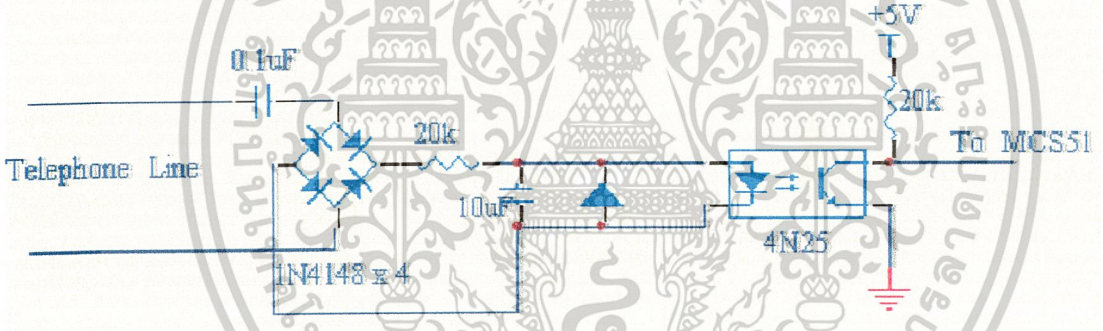
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
การศึกษาและการออกแบบสร้างวงจร

3.1 การออกแบบเครื่องตอบรับและสั่งงานทางโทรศัพท์

3.1.1 วงจรตรวจจับสัญญาณกระดิ่ง และหน่วงเวลา

หลักการทำงานของวงจรคือ เมื่อมีสัญญาณโทรศัพท์เข้ามาซึ่งจะมีแรงดันประมาณ 100 Vp-p 20 Hz วงจรรับสัญญาณที่เข้ามาแล้วแปลงเป็นระดับบิต (0 โวลท์ เมื่อมีสัญญาณ 12 โวลท์ เมื่อไม่มีสัญญาณดัง) เพื่อเป็นระดับสัญญาณที่เหมาะสมสำหรับป้อนให้แก่วงจรหน่วงเวลา โดยใช้วงจรบริดจ์เรคติไฟเออร์ (Bridge Rectifier) แปลงให้เป็นสัญญาณไฟกระแสตรงแล้วแบ่งแรงดันด้วยความต้านทาน เพื่อป้อนให้ทรานซิสเตอร์เป็นส่วนสวิตช์ซึ่งเพื่อทำหน้าที่สร้างพัลส์ลบ (Negative pulse) ให้วงจรประมวลผลหน่วงเวลาและทำงานต่อไป ซึ่งในการทดลองจะต่อวงจรดังรูปที่ 3.1



รูปที่ 3.1 วงจรตรวจจับสัญญาณกระดิ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

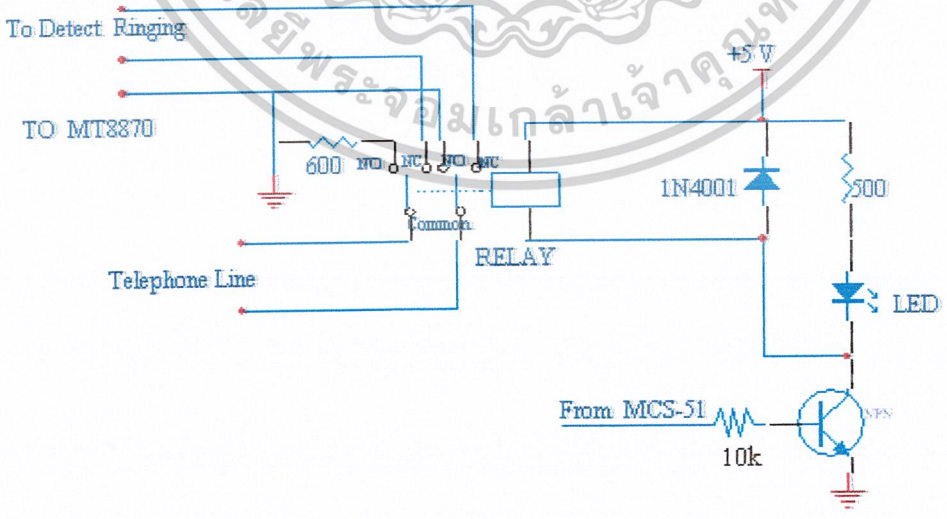
3.1.2 วงจรยกหู/วางหูโทรศัพท์

ขั้นตอนการทำงานในส่วนที่ทำงานกับสัญญาณโทรศัพท์คือส่วนตรวจจับสัญญาณกระดิ่งซึ่งจะทำหน้าที่สร้างพัลส์ลบ (Negative pulse) ทุกครั้งที่มีสัญญาณกระดิ่งเข้ามา และส่งพัลส์ลบให้ไมโครหน่วงเวลา และส่งสัญญาณไปกระตุ้นส่วนควบคุมการยกหู และว่างหู

จากหลักการลดค่าอิมพีแดนซ์ในการยกหู ได้นำมาใช้กับวงจรนี้โดยทำให้เกิดสภาวะความต้านทานต่ำขึ้นระหว่างคู่สาย ก็เหมือนกับเป็นการยกหู

การยกหูโทรศัพท์นี้มิใช้เป็นการยกหูโดยตรง แต่ใช้ความรู้พื้นฐานทางโทรศัพท์ คือ ในสภาวะสายว่างคู่สายโทรศัพท์จะมีแรงดันประมาณ 48 โวลท์ ซึ่งจ่ายมาจากชุมสายโทรศัพท์และเมื่อมีผู้เรียกเข้ามาทางชุมสายจะจ่ายสัญญาณกระดิ่งมาเป็นแรงดันกระแสสลับที่มีแรงดันประมาณ 100 Vp-p เป็นเวลา 2 วินาที และหยุดเป็นเวลา 3 วินาที เป็นจังหวะสัญญาณ ซึ่งแรงดันนี้จะทำให้กระดิ่งภายในตัวโทรศัพท์ทำงาน และทางชุมสายจะรับทราบการยกหู จากวงจรภายในตัวโทรศัพท์จะทำการต่อคู่สายเข้ากับวงจรภายในที่มีความต้านทานกระแสตรงต่ำเมื่อเรายกหูจะเกิดครบวงจรขึ้นทำให้แรงดันลดลงเหลือประมาณ 6 – 10 โวลท์ เมื่อชุมสายโทรศัพท์รับรู้แล้วจะต่อคู่สายของคุณเข้ากับผู้เรียก ทำให้ติดต่อกันได้

จากหลักการที่กล่าวมาข้างต้นได้นำมาใช้กับวงจรนี้โดยทำให้เกิดสภาวะความต้านทานต่ำขึ้นระหว่างคู่สาย ก็เหมือนกับการยกหูได้เหมือนกัน โดยจะรับสัญญาณจากไมโครคอนโทรลเลอร์มาควบคุมรีเลย์ให้ต่อส่วนถ่ายทอดสัญญาณออกสู่คู่สายโทรศัพท์ซึ่งให้วงจรตอบรับข้อความเริ่มทำงานต่อไป ในส่วนแผนผังของวงจรแสดงได้ดังรูปที่ 3.2



รูปที่ 3.2 แผนผังการทำงานของวงจรยกหูโทรศัพท์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.3 วงจรถอดรหัสความถี่โทรศัพท์

การถอดรหัสความถี่โทรศัพท์ หมายถึง การแปลงสัญญาณความถี่ซึ่งเกิดจากการกดปุ่มตัวเลขของโทรศัพท์ชนิดกดปุ่ม (ชนิด Tone หรือ DTMF) ให้เป็นระบบตัวเลขทางดิจิตอลซึ่งไอซี MT8870 ใช้แปลงความถี่โทรศัพท์ให้เป็นเลขฐานสอง ขนาด 8 บิต

คุณสมบัติของ MT8870

-เป็นตัวรับและถอดรหัสความถี่ (DTMF receiver)

-กินไฟน้อย ใช้ไฟเลี้ยงระดับเดียวกับ TTL

-สามารถตั้งอัตราขยายภายในตัวไอซีได้

-สามารถปรับการ์ดไทม์ (Guard time)

-โครงสร้างของ MT8870

โครงสร้างภายในของ MT8870 ประกอบด้วยวงจรกรองความถี่และวงจรถอดรหัสฟังก์ชั่นทางดิจิตอลเป็นไอซีที่สร้าง โดยเทคโนโลยีไอเอสโอทู-ซีมอส (ISO^2-CMOS) ในส่วนวงจรกรองความถี่ใช้เทคนิคของสวิตช์คาปาซิเตอร์ฟิลเตอร์ สำหรับกรองความถี่สูงและต่ำ ส่วนวงจรถอดรหัสใช้เทคนิคการนับทางดิจิตอลเพื่อตรวจจับและถอดรหัสทั้ง 16 ความถี่ออกเป็นเลขฐานสองขนาด 4 บิต และเช็คช่วงเวลาที่สัญญาณเข้ามา ส่วนภาคอินพุทเป็นออปแอมป์ซึ่งปรับอัตราขยายได้โดยอุปกรณ์ภายนอก เอาท์พุทเป็นวงจรแลตช์ 3 สถานะ

-ฟังก์ชั่นการทำงานภายใน MT8870

ภายใน MT8870 ประกอบไปด้วยส่วยสำคัญ 5 ส่วน คือ

1. ภาคกรองความถี่ (filter section)

ในส่วนนี้จะแยกสัญญาณความถี่คู่ของโทรศัพท์ ที่เข้ามาออกเป็น 2 กลุ่มความถี่ คือ ช่วงความถี่สูง และช่วงความถี่ต่ำโดยใช้วงจรกรองความถี่ระดับ 6 ชนิด สวิตช์คาปาซิเตอร์ (six-order switched capacitor band filter) ซึ่งความถี่แยกได้มี 2 ช่วง คือ ความถี่สูงและความถี่ต่ำ

2. ภาคถอดรหัส (decoder section)

สัญญาณความถี่คู่ของโทรศัพท์ ซึ่งถูกกรองเรียบร้อยแล้วจะผ่านเข้าวงจรถอดรหัสความถี่ออกเป็นตัวเลข โดยใช้เทคนิคการนับแบบดิจิตอล (Digital) และมีการตรวจสอบความถี่ที่เข้ามาว่าเป็นความถี่มาตรฐาน สัญญาณความถี่คู่ของโทรศัพท์หรือไม่ เพื่อป้องกันความถี่อื่นเข้ามาผสมเมื่อตรวจสอบว่าความถี่นั้นถูกต้อง สัญญาณที่ขา ESt (early steering) ก็จะแอคทีฟสำหรับค่าที่ถอดรหัสได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.ภาคตรวจสอบสัญญาณ (steering circuit)

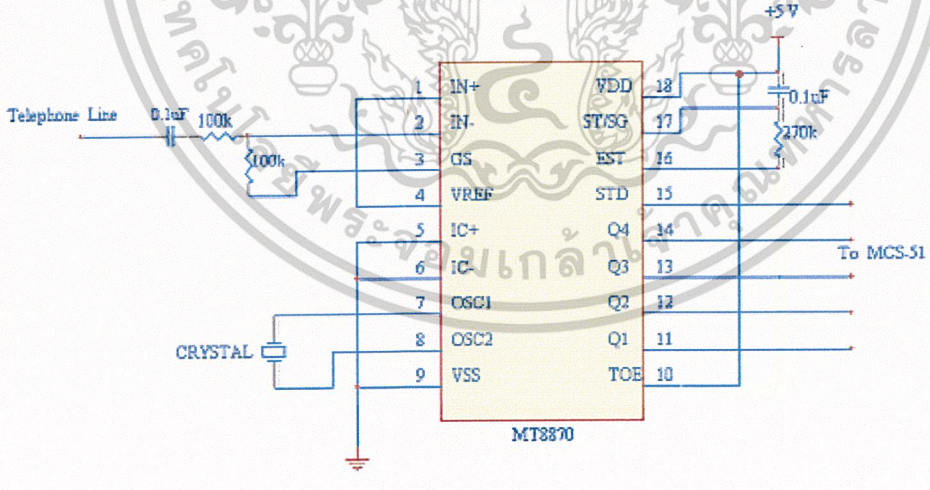
ก่อนที่จะมีการถอดรหัสความถี่ไปที่เอาท์พุท จะมีการตรวจสอบช่วงความถี่ที่เข้ามาว่ามีระยะเวลาตามที่กำหนดหรือไม่ โดยสังเกตุจากระยะเวลาการกดปุ่มโทรศัพท์ ซึ่งต้องกดปุ่มให้มีความถี่ออกมาเป็นช่วงเวลาพอสมควรมิฉะนั้นวงจรส่วนนี้จะไม่รับ โดยถือว่าวงจรนั้นไม่ถูกต้อง ส่วนเวลายาวเท่าใดสามารถตั้งได้โดยใช้ RC ต่อภายนอก สัญญาณที่ขา ESt จะมีสัญญาณบิตสูง นานใกล้เคียงกับระยะเวลาที่มีความถี่โทรศัพท์เข้ามาเมื่อขา ESt มีสัญญาณบิตสูง ทำให้ Vc สูงขึ้นจนถึงค่าเทรชโฮลด์ (Trach hold) วงจรถอดรหัสออกเป็นตัวเลขขนาด 4 บิต

4. ภาคขยายสัญญาณความต่าง (differential input)

วงจรส่วนอินพุทของ MT8870 เป็นภาคขยายออปแอมป์ (Op-Amp) ที่สามารถเปลี่ยนอัตราขยาย โดยต่อวงจรภาคนอกเพิ่มเข้าไปได้โดยดูจากข้อมูลใน Data sheet

5. ภาคกำเนิดความถี่ (Oscillator)

ในภาคนี้ภายในไอซีจะมีวงจรเวลาอยู่ภายในเพียงต่อ คริสตรอล(Crystal) ขนาด 3.5 MHz ก็สามารถใช้งานได้ทันที



รูปที่ 3.3 วงจรการใช้งานของ MT8870

3.1.4 ภาคประมวลผล

ใช้ไมโครคอนโทรลเลอร์ตระกูล MCS – 51 ในการประมวลผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 การออกแบบส่วนของการทำงานอัตโนมัติโดยใช้การตั้งเวลาซึ่งอาศัยฐานเวลาจริงจาก IC REAL TIME CLOCK

ส่วนของการทำงานอัตโนมัติโดยใช้การตั้งเวลาซึ่งอาศัยฐานเวลาจริงจาก RTC ประกอบด้วย

1. ส่วนของการป้อนข้อมูล ใช้ KEYPAD และ BUTTON SWITCH ในการติดต่อระหว่างผู้ใช้กับไมโครคอนโทรลเลอร์

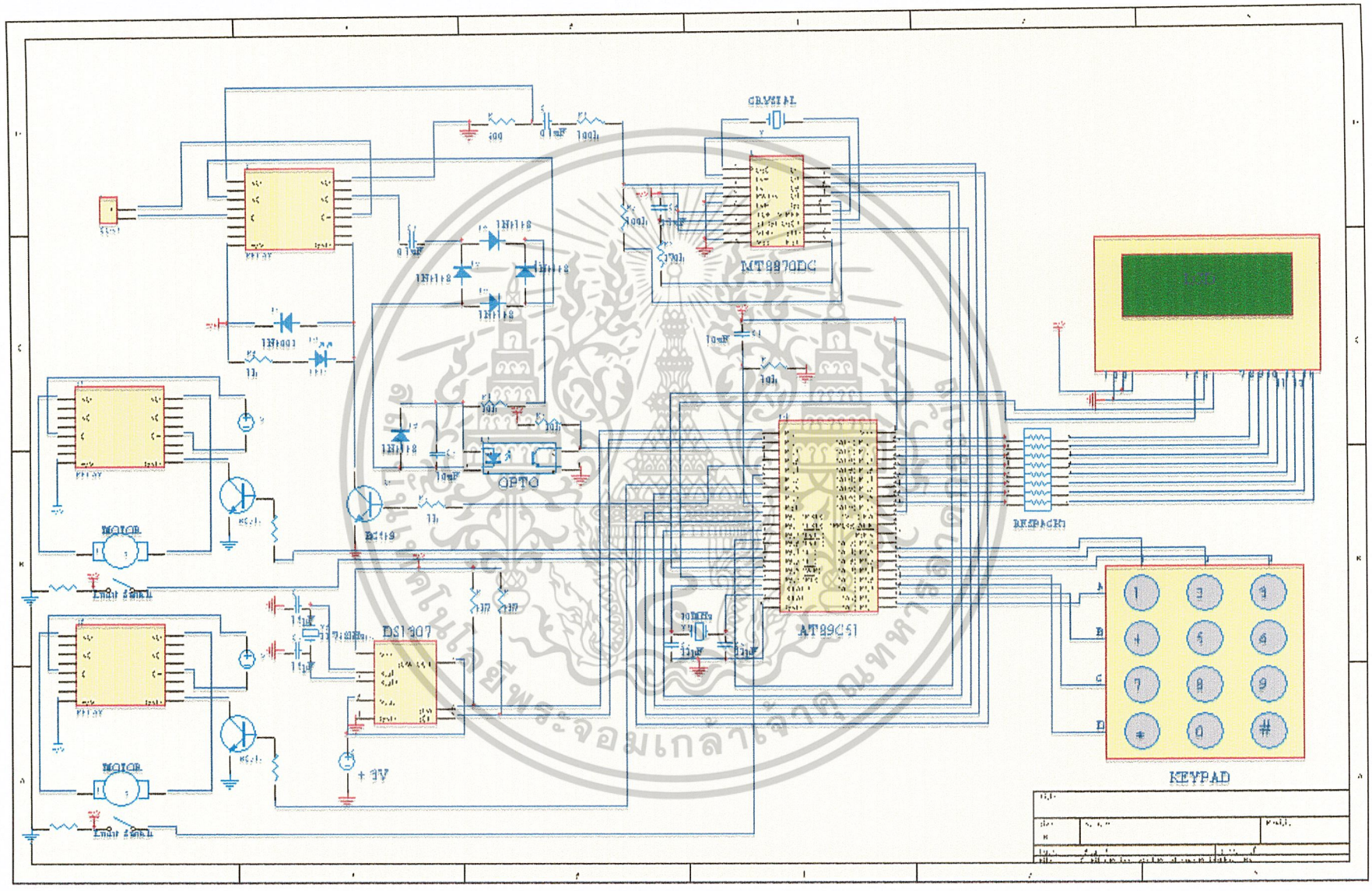
2. ส่วนของการแสดงผลใช้ LCD module ในการแสดงผล การร้องขอการใส่รหัสผ่าน , แสดงข้อความสื่อสารกับผู้ใช้งาน และแสดงข้อมูลของวันเวลาที่ได้มาจาก IC REAL TIME CLOCK(RTC)

3. ส่วนที่ใช้สร้างข้อมูลของฐานเวลาจริง ใช้ IC REAL TIME CLOCK (RTC) เบอร์ DS1307 ในการสร้างข้อมูลฐานเวลาจริง ส่งให้ ไมโครคอนโทรลเลอร์

4.ส่วนของการประมวลผลและสั่งการ ใช้ไมโครคอนโทรลเลอร์ในการประมวลผลและสั่งการ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

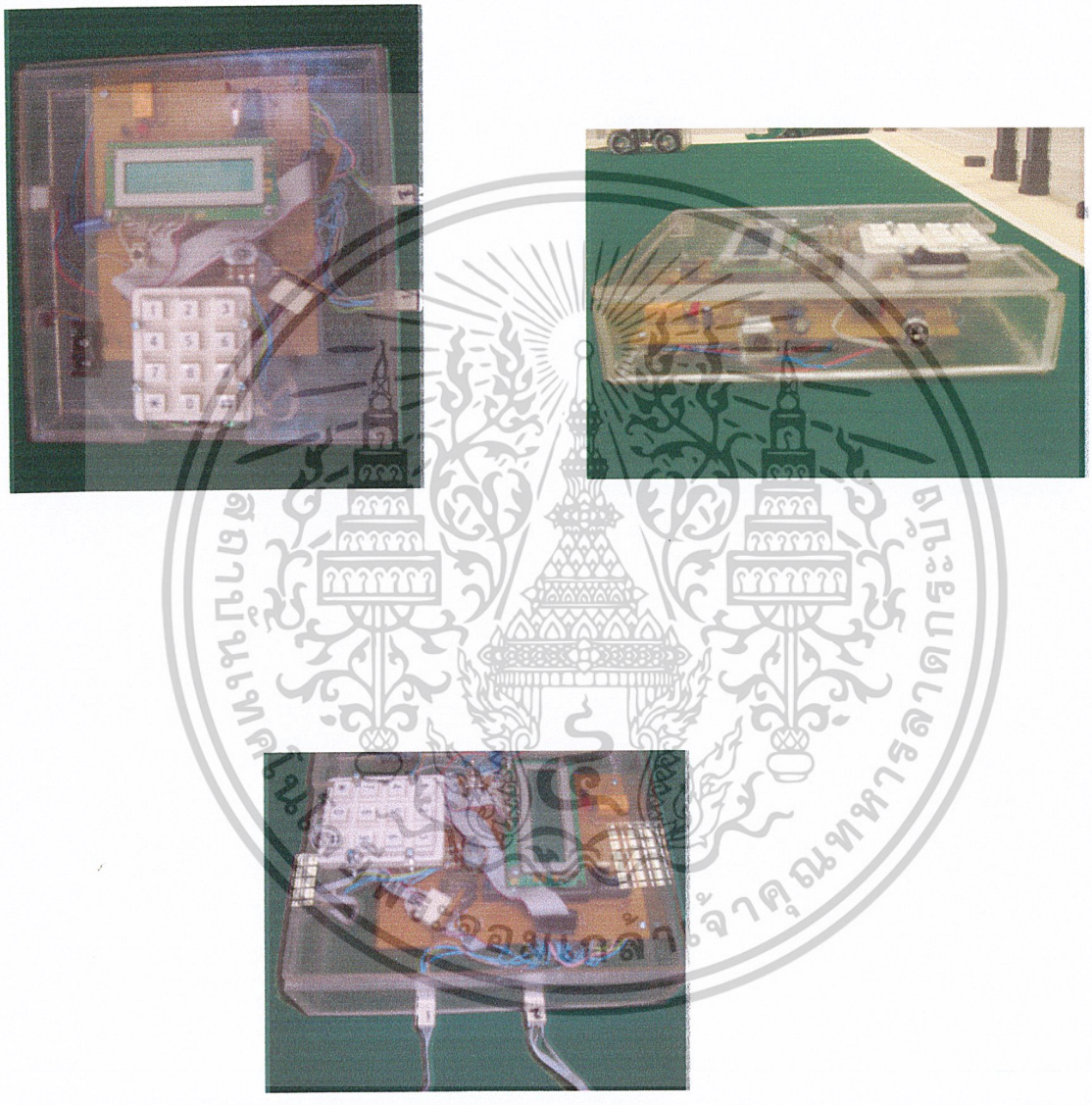
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.4 แสดงการต่อวงจรภาคสั่งงานทางโทรศัพท์ และภาคทำงานอัตโนมัติเข้ากับไมโครคอนโทรเลอร์

3.3 การออกแบบตัวเครื่องให้อาหาร

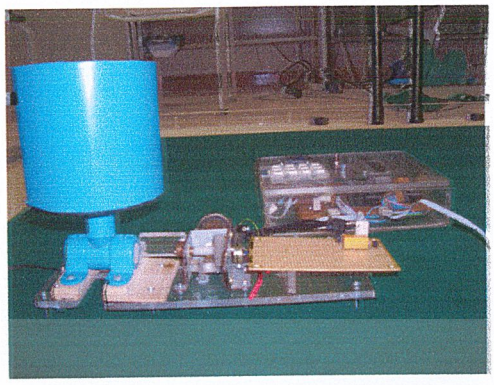
3.3.1 การออกแบบตัวเครื่องที่ใช้ควบคุมการทำงานและติดต่อกับผู้ใช้งาน

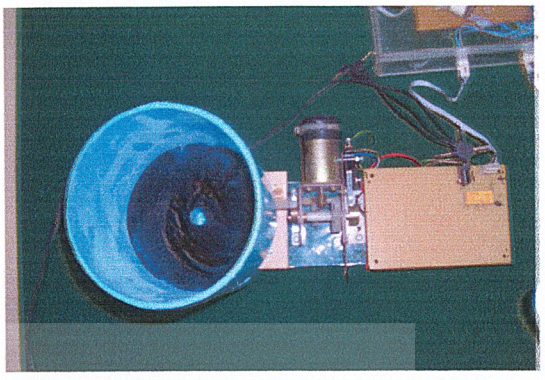


รูปที่ 3.5 แสดงรูปแบบของตัวเครื่องที่ใช้ควบคุมการทำงานและติดต่อกับผู้ใช้งาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

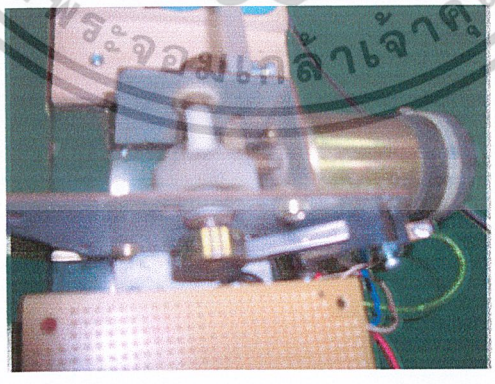
3.2 การออกแบบส่วนที่ใช้เป็นช่องทางการให้อาหารที่ 1





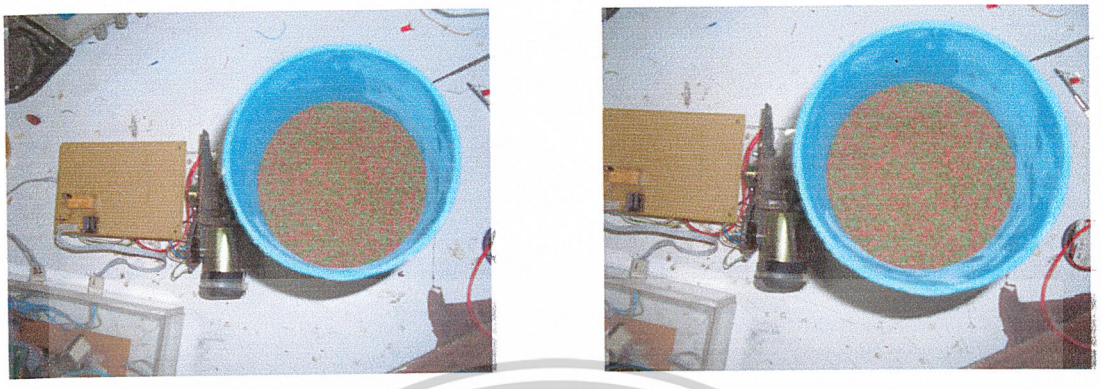


รูปที่ 3.6 แสดงการออกแบบส่วนที่ใช้เป็นช่องทางการให้อาหารที่ 1



รูปที่ 3.7 แสดงส่วนของ มอเตอร์ ชุดเฟือง และลิมิตสวิตช์ที่ใช้งานของช่องทางการให้อาหารที่ 1

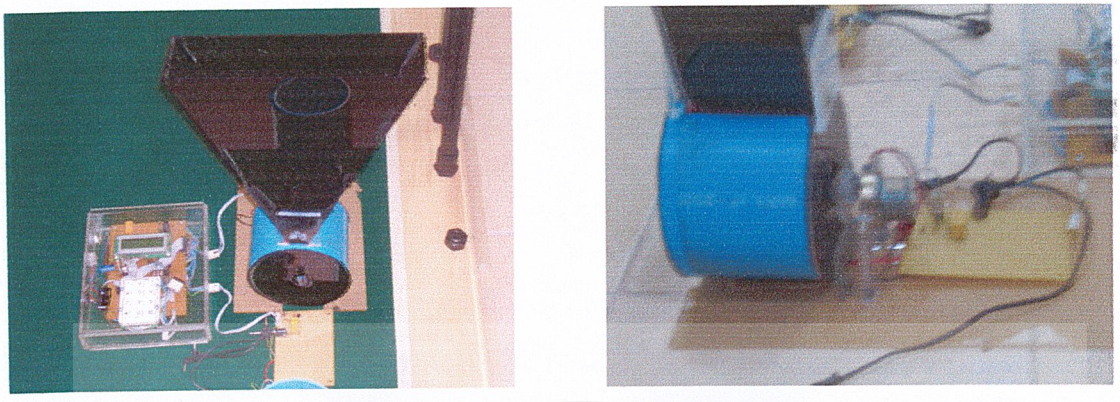
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.8 แสดงลักษณะของอาหารที่ใช้กับช่องทางการให้อาหารที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 การออกแบบส่วนที่ใช้เป็นช่องทางการให้อาหารที่ 2

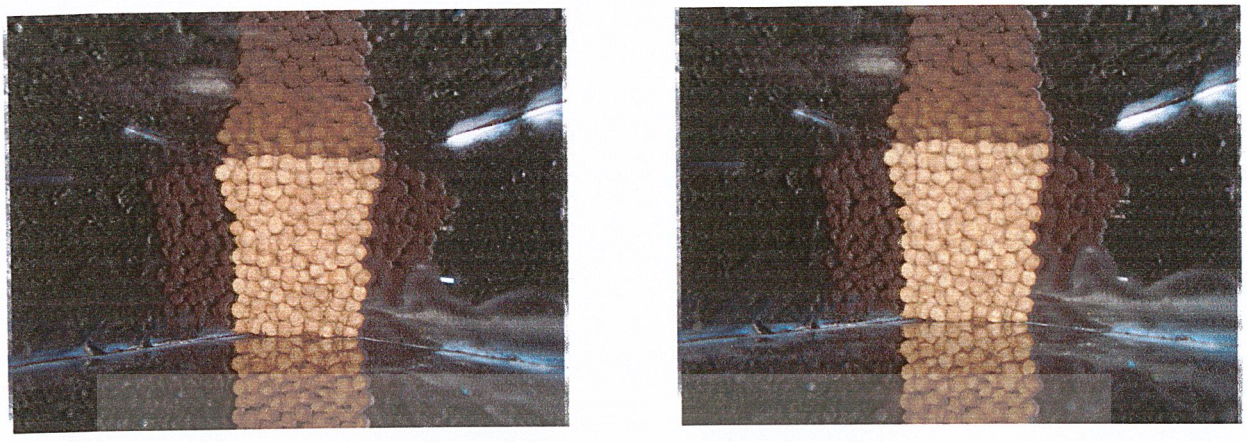


รูปที่ 3.9 แสดงการออกแบบส่วนที่ใช้เป็นช่องทางการให้อาหารที่ 2



รูปที่ 3.10 แสดงส่วนของ มอเตอร์ และลิมิตสวิตช์ที่ใช้งานของช่องทางการให้อาหารที่ 2

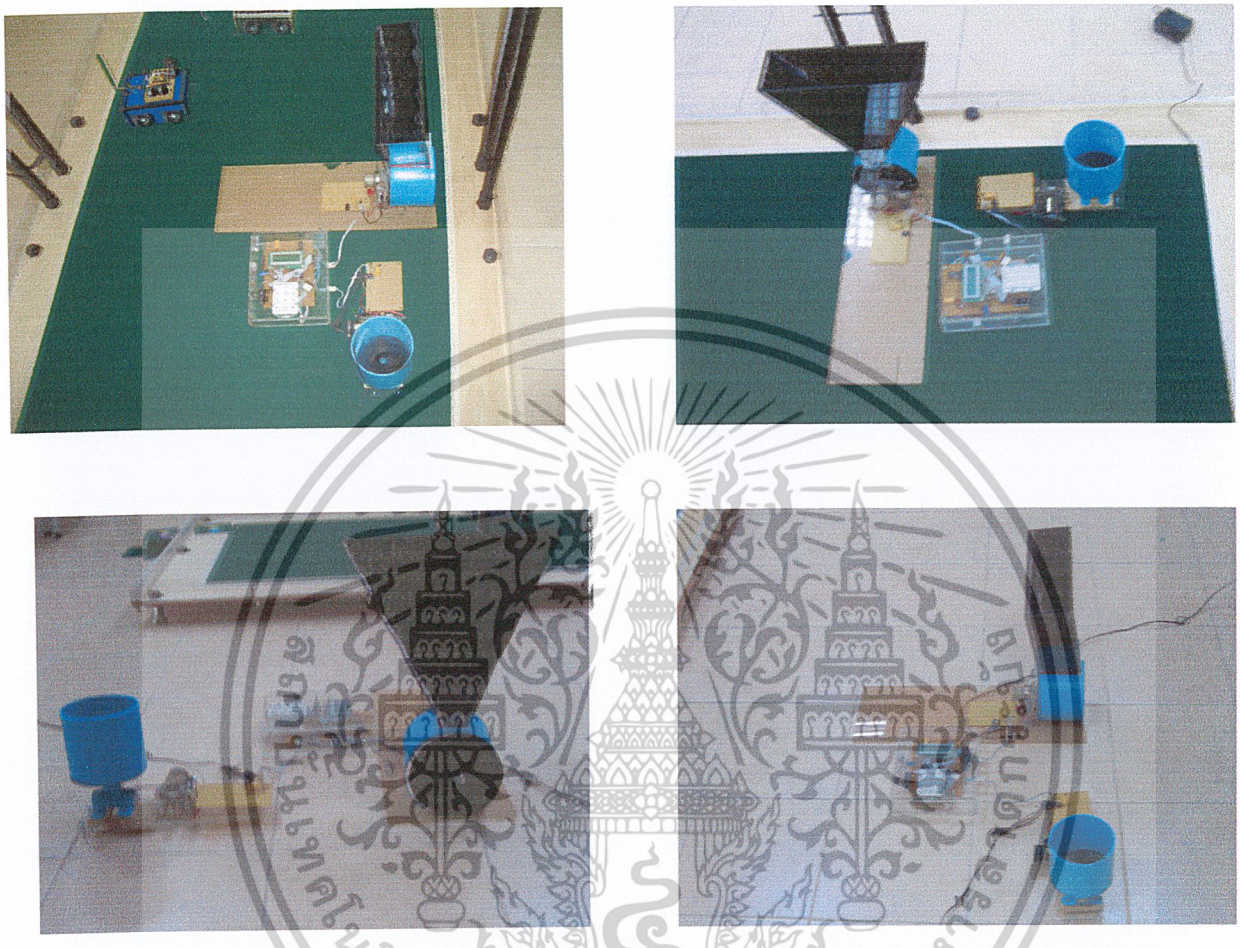
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.11 แสดงลักษณะของอาหารที่ใช้กับช่องทางการให้อาหารที่ 2

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.12 แสดงลักษะการชื่อมต่อในการใช้ของช่องทางการการให้อาหารทั้งสองกับส่วนควบคุมการทำงานและติดต่อกับผู้ใช้งาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
คุณสมบัติและลำดับการใช้งาน

4.1 คุณสมบัติและลำดับการใช้งานของภาคสั่งงานทางโทรศัพท์

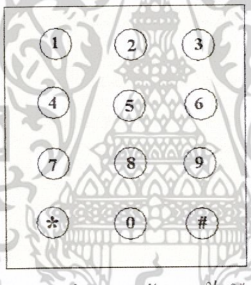
ขั้นตอนการติดต่อกับเครื่องให้อาหารปลาอัตโนมัติทางโทรศัพท์

1. เมื่อผู้ใช้โทรเข้ามาในหมายเลขโทรศัพท์ที่ต่ออยู่กับเครื่อง สัญญาณกระดิ่งจะถูกส่งมาให้กับเครื่องทำให้สามารถตรวจจับการเรียกเข้าได้

2. เมื่อเครื่องให้อาหารปลาอัตโนมัติสามารถตรวจจับสัญญาณเรียกเข้าได้แล้วจะหน่วงเวลาประมาณ 30 วินาที เพื่อให้แน่ใจว่าไม่มีคนอยู่ที่บ้าน แล้วจึงส่งการยกหู

3. หลังจากเครื่องสั่งการยกหูแล้ว การที่จะสั่งงานผ่านโทรศัพท์ได้ต้องใส่หมายเลข PASSWORD ให้ถูกต้องเสียก่อน ถ้าผิดจะให้ใส่ใหม่และถ้าผิดจนถึง 3 ครั้งเครื่องจะทำการวางหูทันที

4. ถ้ากดหมายเลข PASSWORD ถูกต้อง จะเป็นการสั่งการทางโทรศัพท์โดยสั่งการจากการกดหมายเลขจากตำแหน่งปุ่มหน้าปัดโทรศัพท์ดังรูป



รูปที่ 4.1 ตำแหน่งของปุ่มหน้าปัดโทรศัพท์

5. โดยการกด 1 หมายถึงการสั่งให้เครื่องให้อาหารทันที โดยหลังจากกด 1 แล้ว เครื่องจะให้เลือกช่องทางที่ให้อาหารนั่นคือ Channel 1 หรือ Channel 2 เมื่อเลือก Channel แล้วจะให้เลือกจำนวนรอบที่เครื่องให้อาหารจะหมุน เมื่อใส่จำนวนรอบแล้วเครื่องจะให้อาหารตามจำนวนรอบที่กำหนดทันที

6. การกด 2 เป็นการสั่งงานให้ยกเลิกโปรแกรมการตั้งเวลาและจำนวนการให้อาหาร ที่เครื่อง โดยหลังจากกดหมายเลข 2 แล้วเครื่องจะให้เลือก Channel นั่นคือ โปรแกรมการตั้งเวลาและจำนวนการให้อาหารของ Channel นั้นจะถูกยกเลิก

7. การกด 3 เป็นการตั้งหมายเลข PASSWORD ใหม่ หลังจากกด 3 เครื่องจะให้ป้อน PASSWORD ใหม่ โดย PASSWORD ใหม่นี้ จะถูกนำไปใช้ในครั้งต่อไปทันที

8. หลังจากการสั่งการด้วยหมายเลข 1หรือ2หรือ3 แล้วเครื่องจะคอยรับคำสั่งใหม่ต่อไป ถ้าต้องการวางหู ให้กดหมายเลข 4 เครื่องจะสั่งการวางหูทันที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 คุณสมบัติและลำดับการใช้งานของภาคการทำงานอัตโนมัติ

1. เมื่อเปิดเครื่องหน้าจอ LCD จะแสดงข้อมูลวัน,วันที่,เดือน,ปีและเวลา

2. การสั่งงานจะรับคำสั่งจาก keypad โดยการกดหมายเลข 1-4 การสั่งงานแต่ละครั้งจะมีการถาม ADMIN PASSWORD และ PASSWORD ถ้าเป็นการสั่งงานครั้งแรกจะถาม ADMIN PASSWORD แล้วให้ตั้ง PASSWORD และครั้งต่อไปจะถามเฉพาะ PASSWORD เท่านั้น

3. การกด 1 จะเป็นการตั้งค่าวัน,วันที่,เดือน,ปี โดยถ้าป้อนข้อมูลที่ไม่ถูกต้องคือข้อมูลที่ไม่สามารถเป็นไปได้เครื่องจะให้ป้อนค่าใหม่ เช่น วันที่ 41 ซึ่งไมมีทางเป็นจริง เป็นต้น

4. การกด 2 จะเป็นการตั้งค่า เวลา โดยถ้าป้อนข้อมูลที่ไม่ถูกต้องคือข้อมูลที่ไม่สามารถเป็นไปได้เครื่องจะให้ป้อนค่าใหม่ เช่น 25.12 ซึ่งไม่สามารถเป็นไปได้

5. การกด 3 จะเป็นการตั้งค่าหมายเลข PASSWORD ใหม่ โดยจะถาม PASSWORD เก่าก่อนแล้วจึงให้ค่า PASSWORD ใหม่

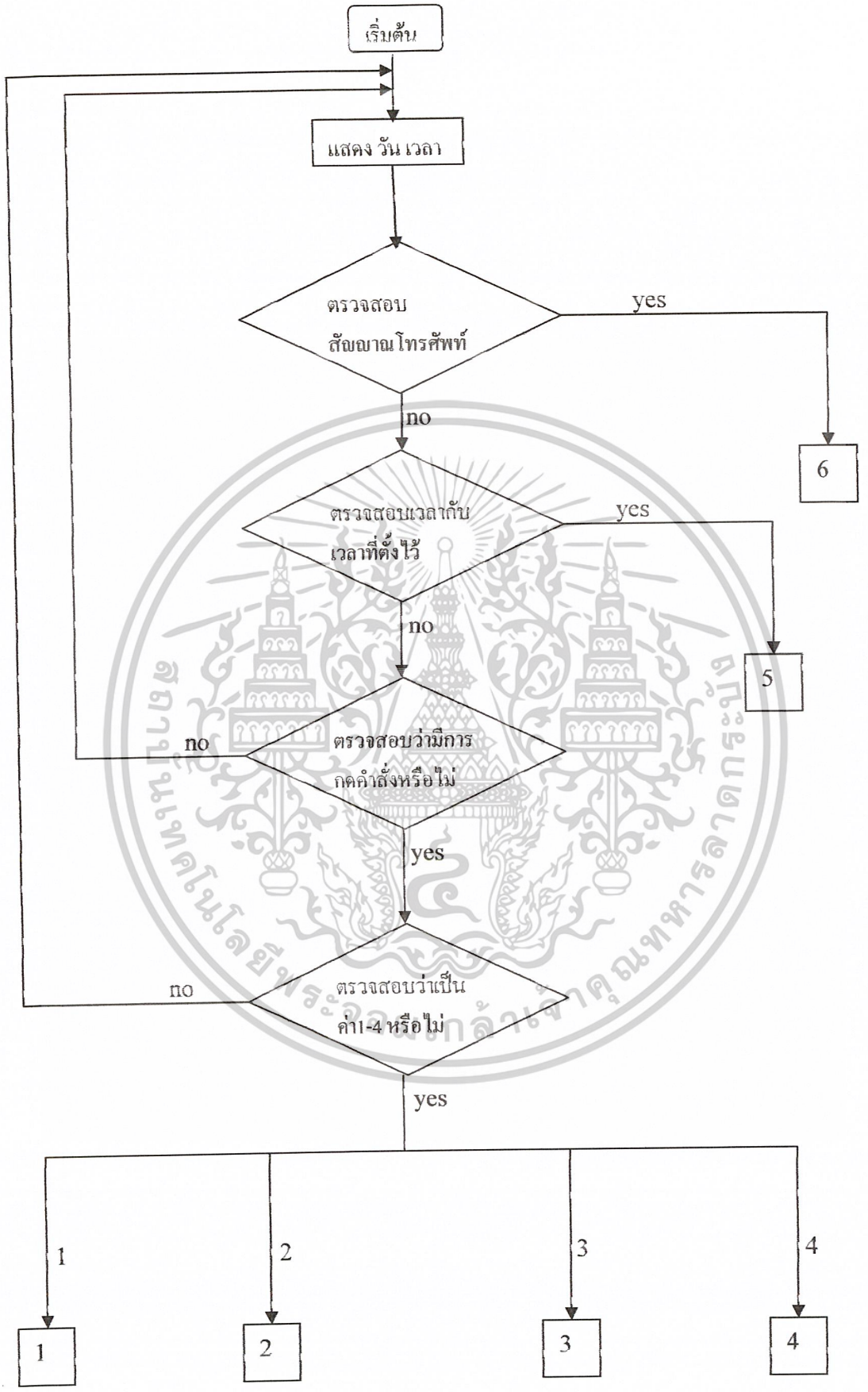
6. การกด 4 จะเป็นการตั้งวัน เวลา จำนวนครั้ง ปริมาณ และ Channel ที่จะให้อาหาร โดยเครื่องจะให้ป้อนหมายเลข Channel หมายเลขวัน จำนวนครั้งต่อวัน เวลาที่ให้ในแต่ละครั้ง จำนวนการหมุนของเครื่องในครั้งนั้นๆ

7. หลังจากทำงานในแต่ละคำสั่งเสร็จสิ้นเครื่องจะกลับมาที่หน้าจอ แสดงวัน วันที่ เดือน ปี เวลา อีกครั้งจนกว่าจะมีการสั่งงานอีกครั้ง

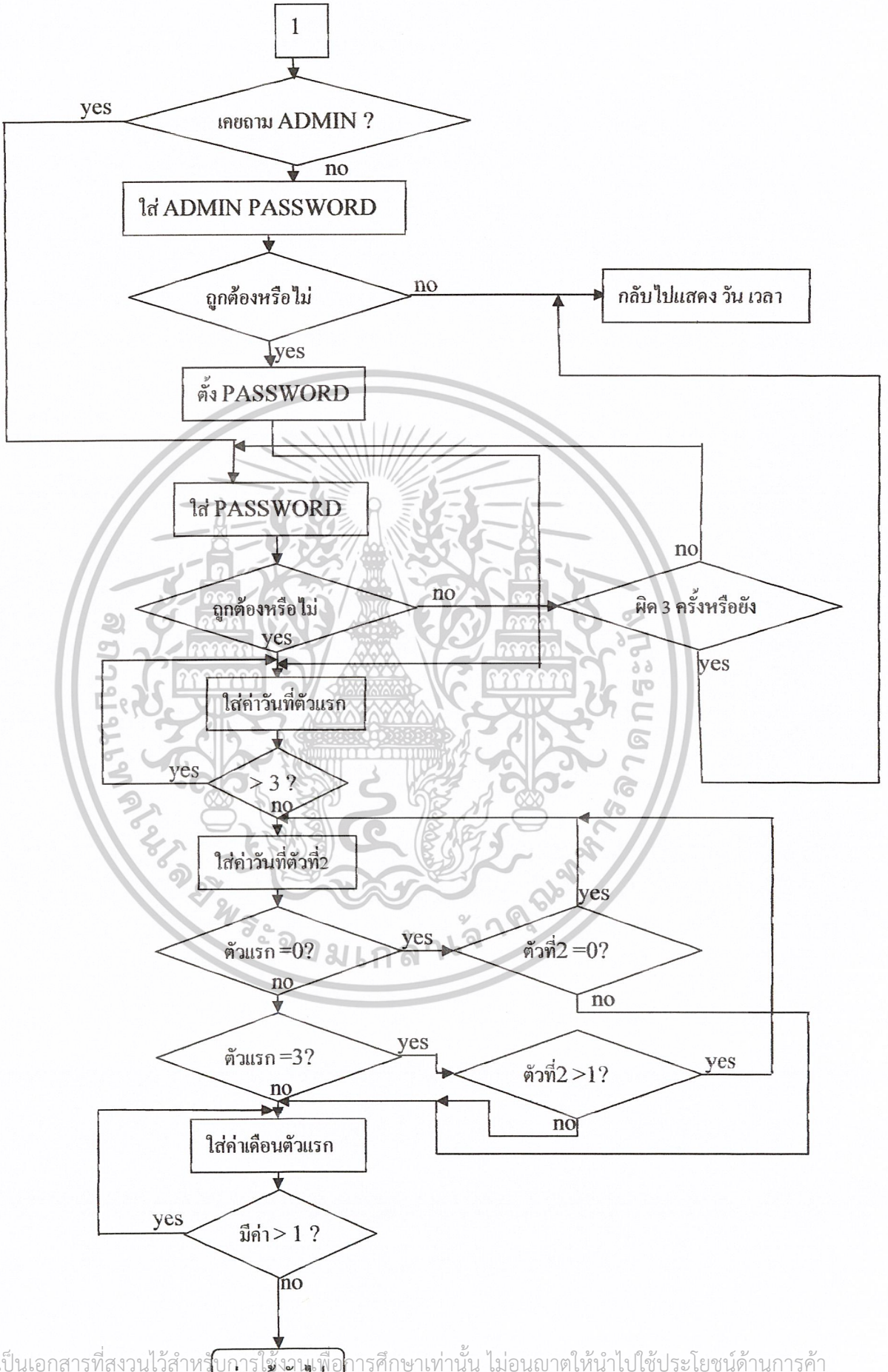
8. การตรวจสอบเวลาปัจจุบันกับเวลาที่ตั้งไว้ให้อาหารจะตรวจสอบทุกการเปลี่ยนนาที เพื่อป้องการทำงานซ้ำซ้อน

9. ในทุกขั้นตอนที่มีการป้อนค่าการกด * จะเป็นการเลื่อนเคอร์เซอร์ไปตำแหน่งซ้ายสุด และการกด # จะเป็นการออกจากการป้อนข้อมูลในคำสั่งนั้น โดยไม่บันทึกเก็บค่าที่ป้อนไปแล้วบางส่วนไว้

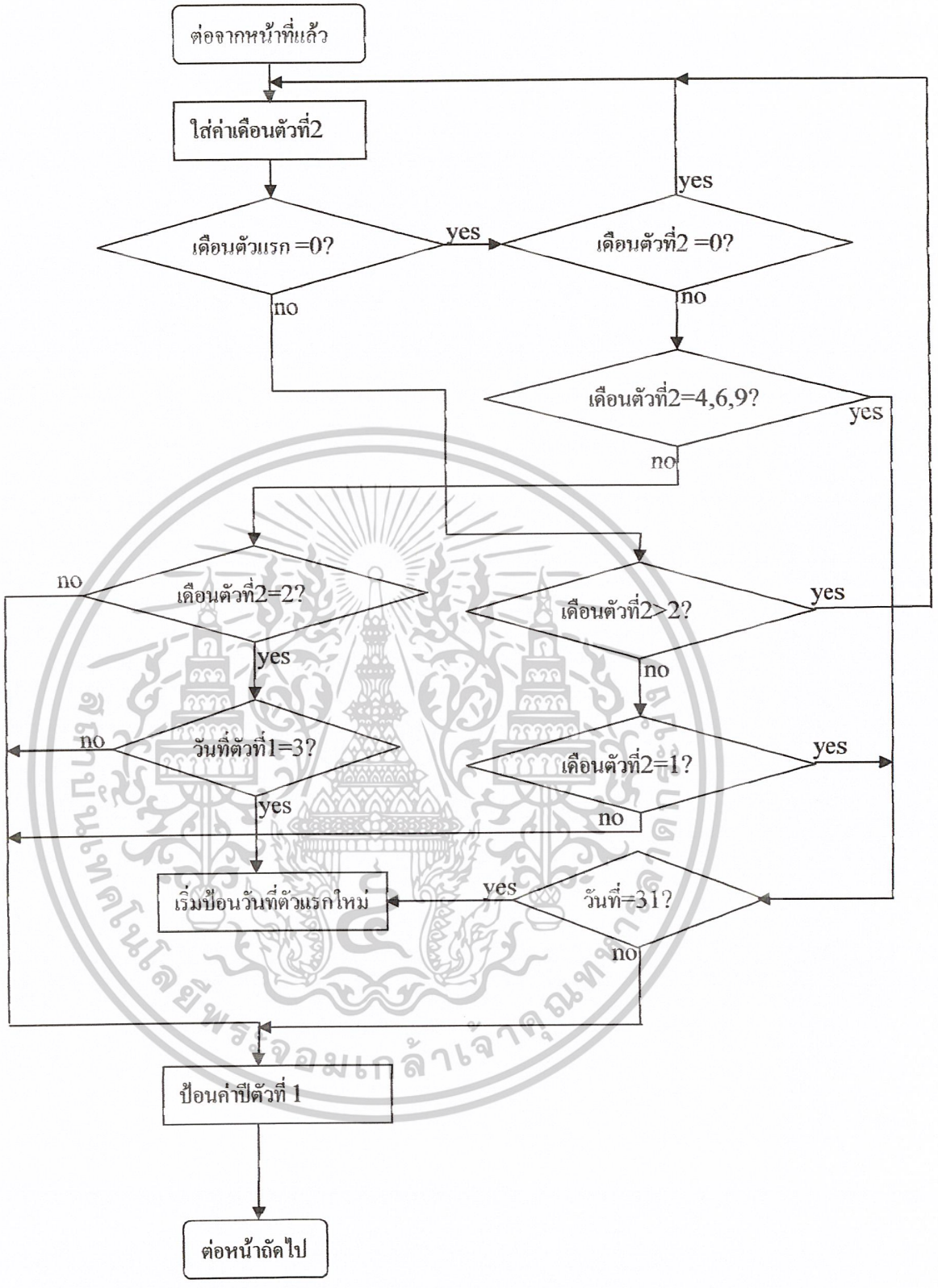
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



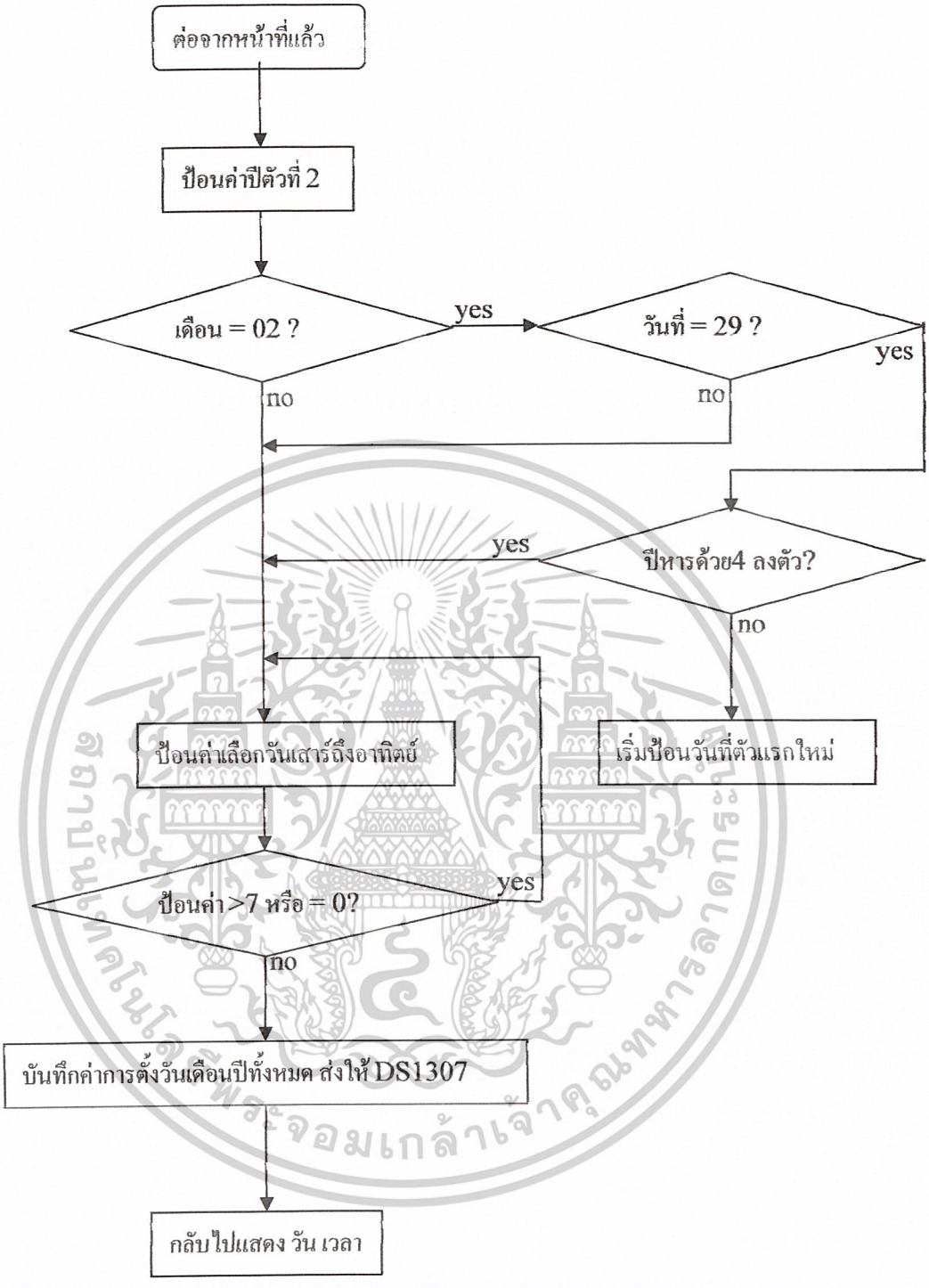
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



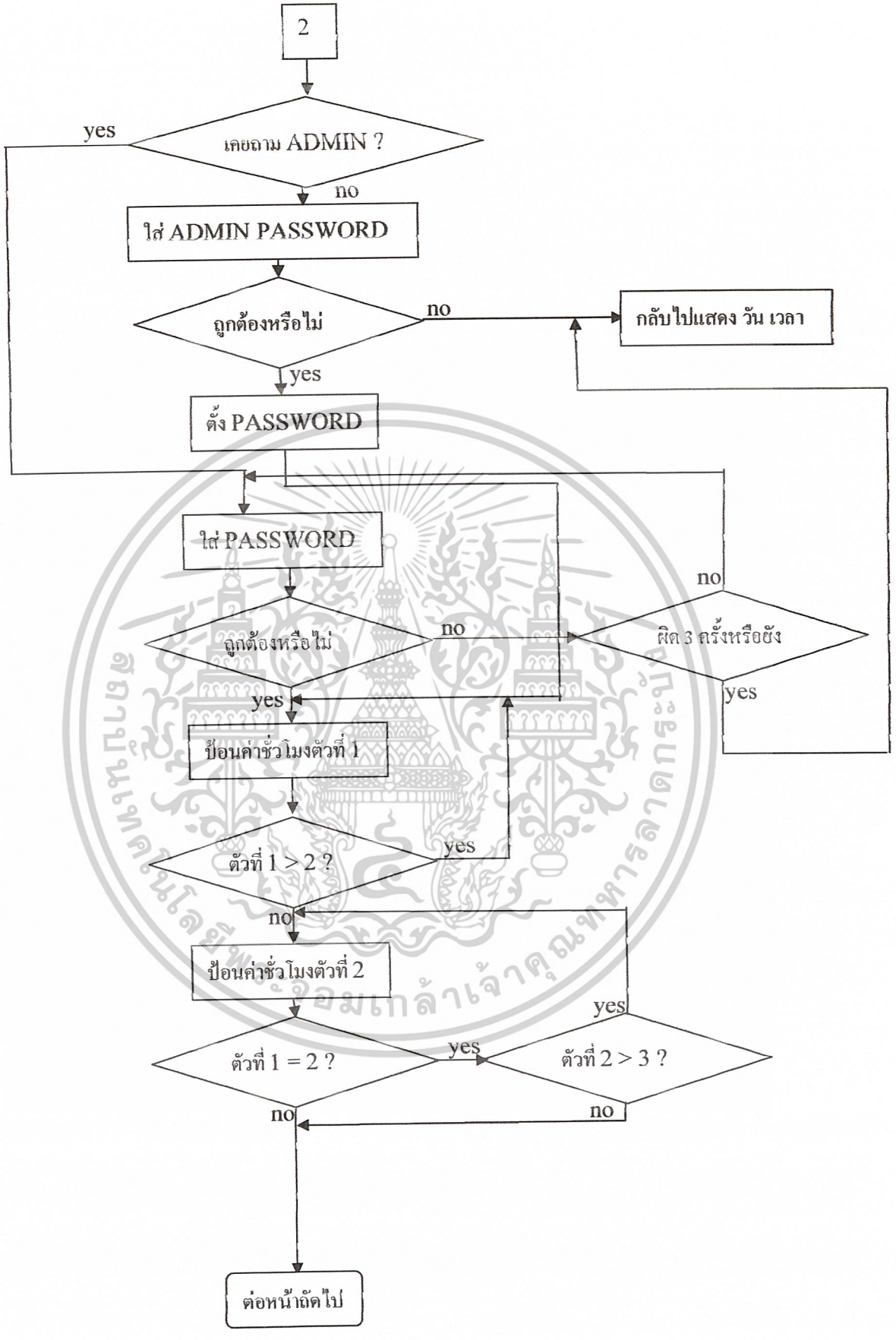
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



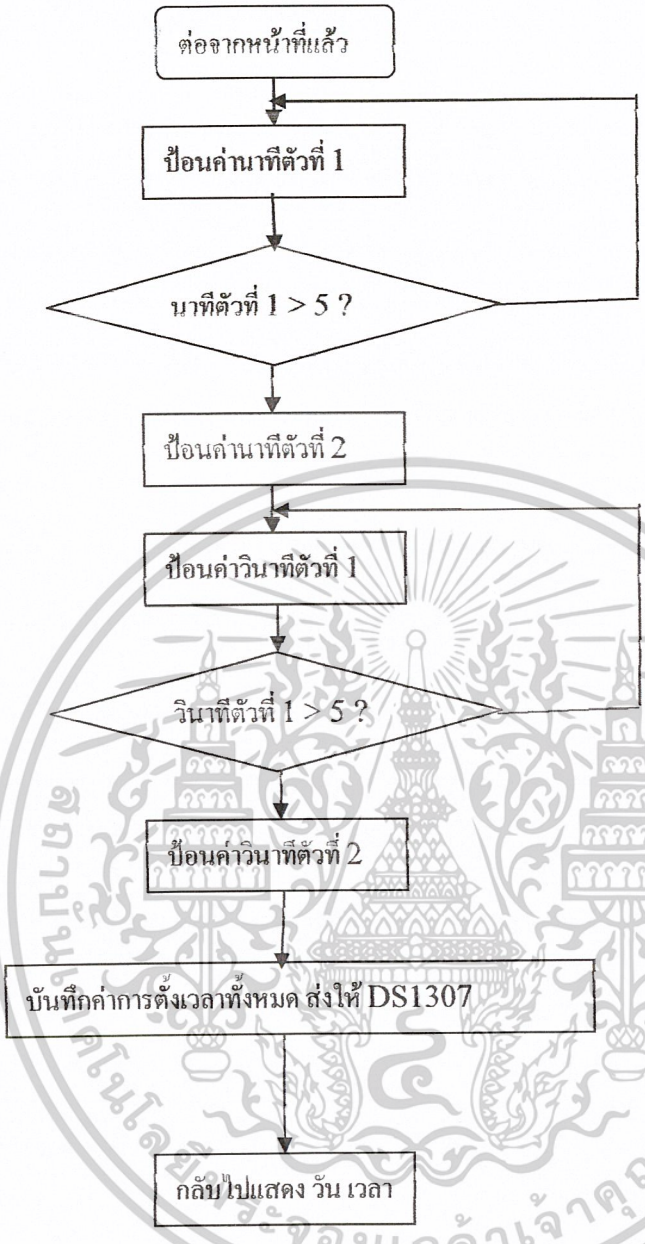
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



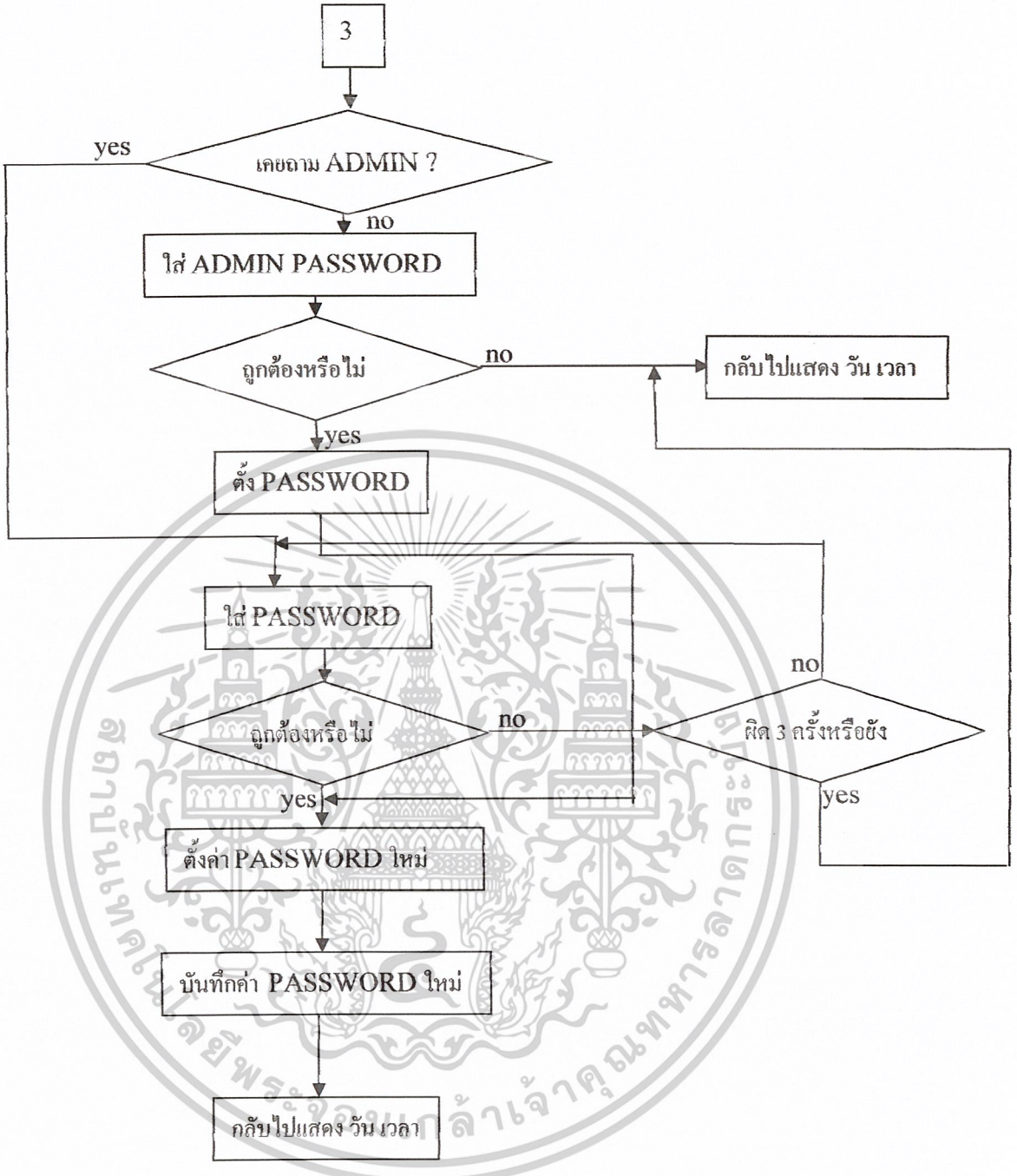
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



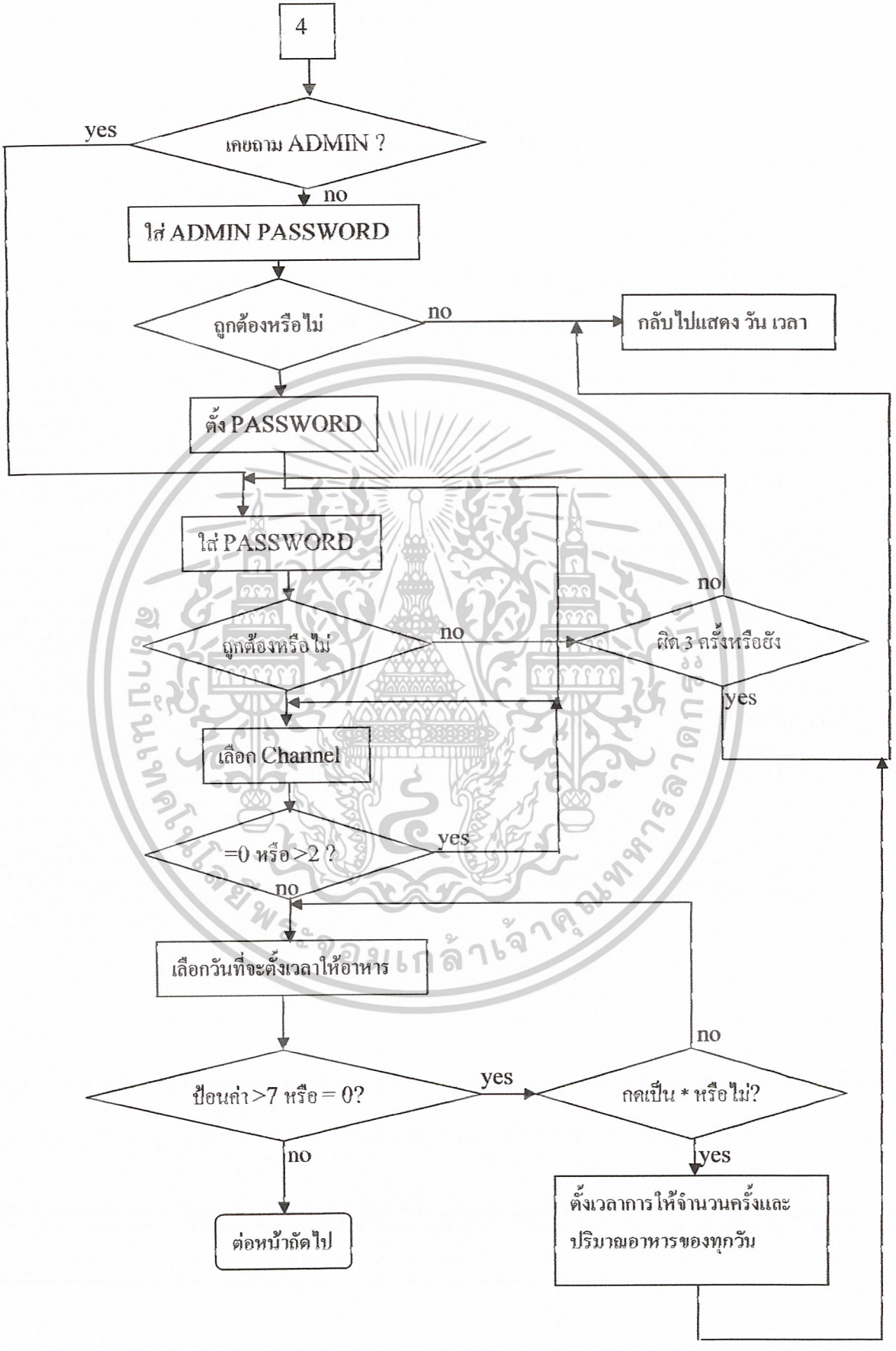
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



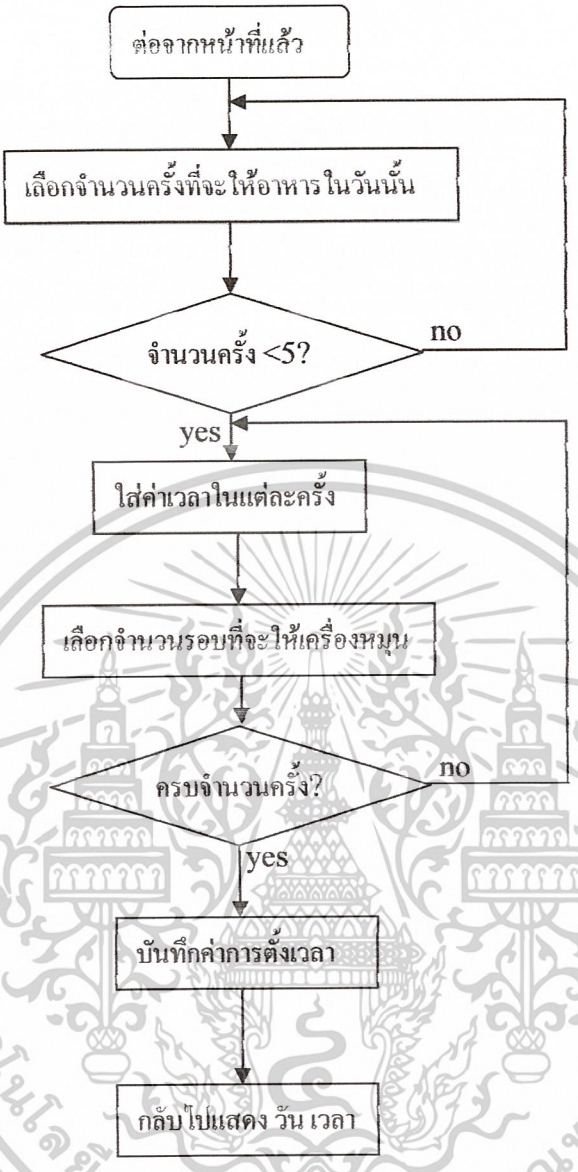
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



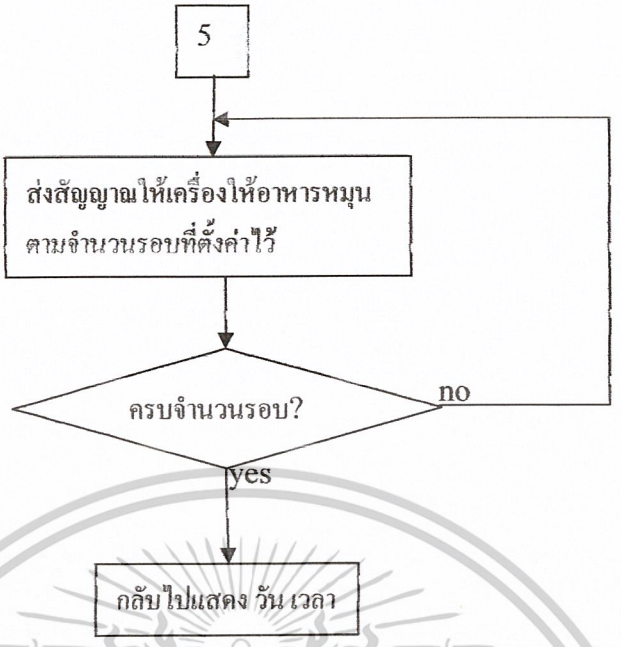
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



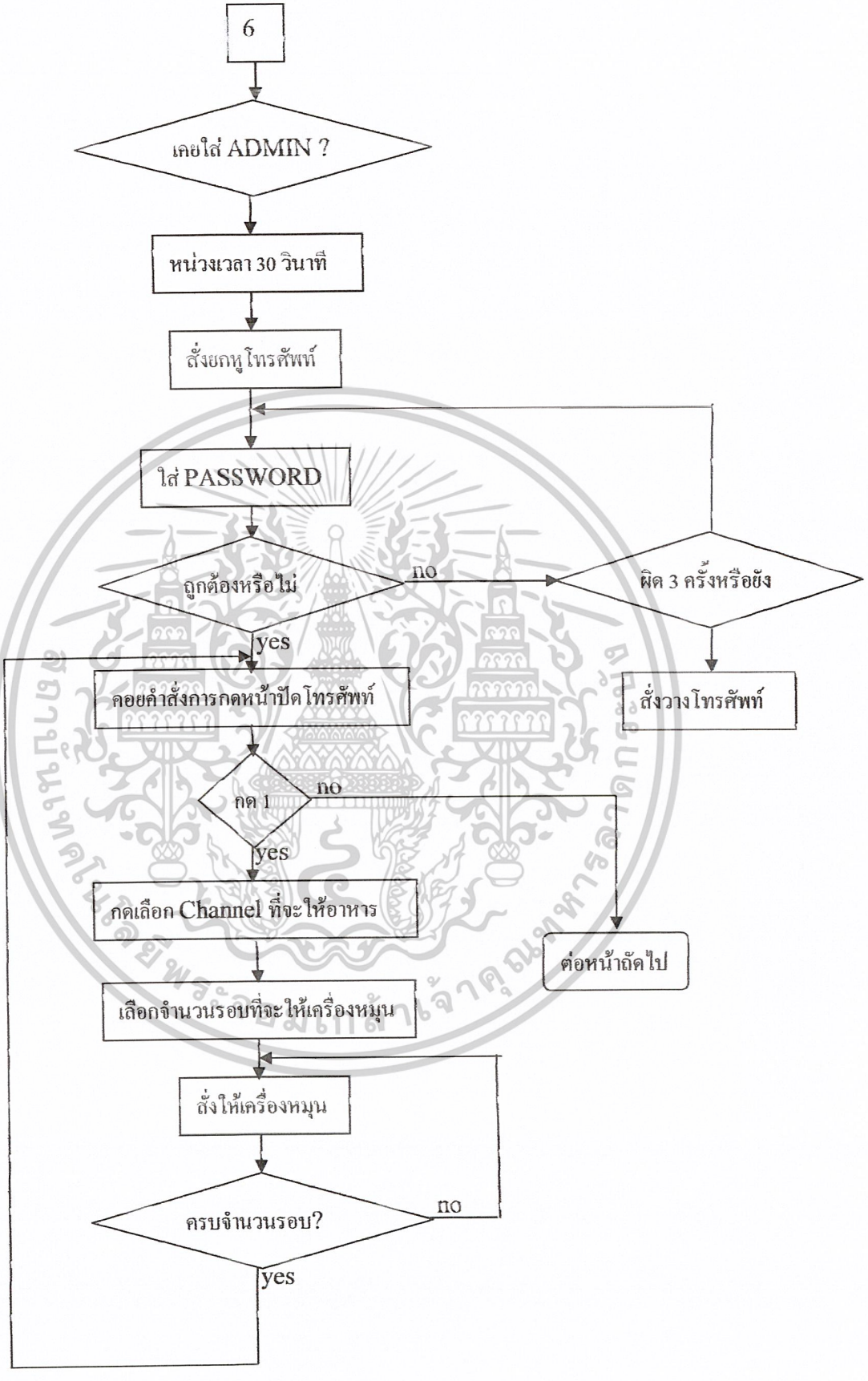
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



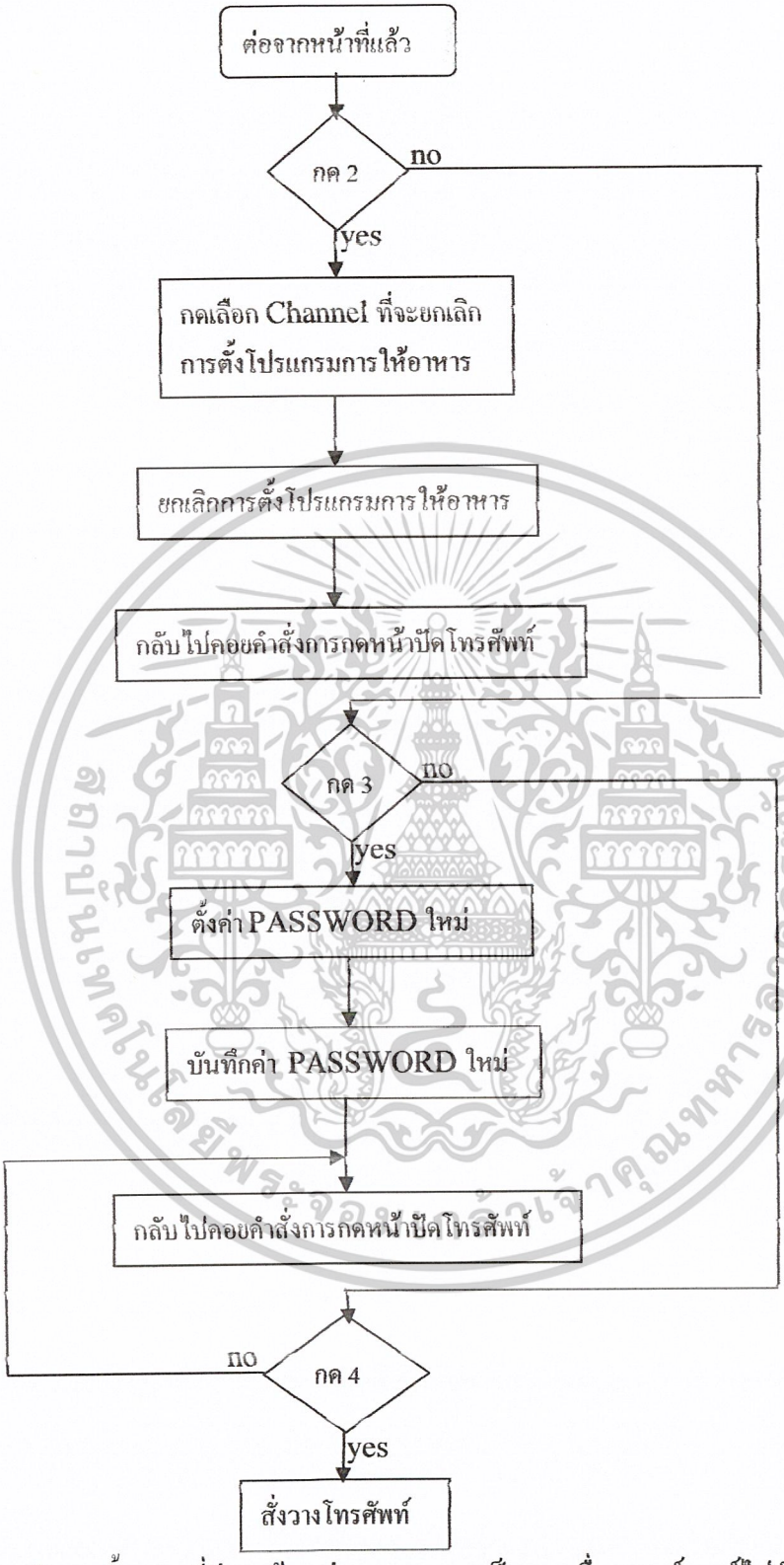
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



<u>หมายเหตุ</u> ในทุกขั้นตอนที่มีการป้อนค่าการกด * จะเป็นการเลื่อนเคอร์เซอร์ไปตำแหน่งซ้ายสุด และการกด # จะเป็นการออกจากการป้อนข้อมูลในคำสั่งนั้นโดยไม่บันทึกเก็บค่าที่ป้อนไปแล้วบางส่วนไว้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
ผลการทดลอง

5.1 กล่าวนำ

สำหรับการทดลองของเครื่องให้อาหารปลาอัตโนมัติและสั่งงานทางโทรศัพท์ จะแบ่งเป็นสามส่วน คือ การทดลองส่วนการสั่งงานทางโทรศัพท์ ส่วนการสั่งงานการทำงานอัตโนมัติที่ตัวเครื่อง และปริมาณอาหารที่ให้ต่อการหมุน 1รอบ ของแต่ละ Channel

5.2 การทดลองส่วนการสั่งงานทางโทรศัพท์

- การทดลองการตรวจจับสัญญาณโทรศัพท์และการสั่งการยกหู

ครั้งที่	การใส่ Admin Password	จำนวนครั้งก่อนสั่งยกหู	ความสามารถในการตรวจจับสัญญาณกระดิ่ง
1	ยังไม่ได้ใส่	ไม่มีการสั่งยกหู	สามารถตรวจจับได้
2	ยังไม่ได้ใส่	ไม่มีการสั่งยกหู	สามารถตรวจจับได้
3	ใส่Admin แล้ว	6	สามารถตรวจจับได้
4	ใส่Admin แล้ว	6	สามารถตรวจจับได้
5	ใส่Admin แล้ว	6	สามารถตรวจจับได้

- การทดลองการตรวจสอบการใส่รหัส Password ก่อนการสั่งงาน

ครั้งที่	รหัส Password ที่ตั้งไว้	รหัส Password ที่ใส่	ครั้งที่ผิด	ผลจากการใส่Password
1	123	102	1	เครื่องให้ใส่ Password ใหม่
2	123	007	2	เครื่องให้ใส่ Password ใหม่
3	123	457	3	เครื่องสั่งวางหูโทรศัพท์
4	589	104	1	เครื่องให้ใส่ Password ใหม่
5	589	213	2	เครื่องให้ใส่ Password ใหม่
6	589	121	3	เครื่องสั่งวางหูโทรศัพท์
7	471	471	0	เครื่องรอการกดคำสั่งต่อ
8	258	258	0	เครื่องรอการกดคำสั่งต่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การทดลองการสั่งให้เครื่องให้อาหารโดยกด 1

ครั้งที่	เลือกChannel	จำนวนรอบที่สั่ง	จำนวนรอบที่หมุน
1	1	3 รอบ	Channel 1 หมุน 3 รอบ
2	1	9 รอบ	Channel 1 หมุน 9 รอบ
3	1	0 รอบ	Channel 1 หมุน 0 รอบ
4	2	5 รอบ	Channel 2 หมุน 5 รอบ
5	2	7 รอบ	Channel 2 หมุน 7 รอบ
6	2	0 รอบ	Channel 2 หมุน 0 รอบ

หมายเหตุ หลังจากทำตามคำสั่งเสร็จเครื่องจะรอการกดคำสั่งต่อไปอีกครั้ง ยกเว้นการกด 4

- การทดลองการยกเลิกการตั้งเวลาการตั้งอาหารโดยการกดคำสั่ง 2

ครั้งที่	เลือกChannel	การตั้งเวลาในChannel นั้น	ผลที่ได้
1	1	วันเสาร์ 2 ครั้ง เวลา 02.35 ,18.22 วันจันทร์ 4 ครั้ง เวลา 06.45,09.44,12.00,15.22	Chennel 1 ไม่ทำงาน ในทุกเวลาที่ตั้งไว้
2	1	วันอังคาร1ครั้ง เวลา 03.25 วันศุกร์3ครั้ง เวลา 12.00,14.33ม20.30	Chennel 1 ไม่ทำงาน ในทุกเวลาที่ตั้งไว้
3	1	วันอาทิตย์ 2 ครั้ง เวลา 14.00,18.55 วันพุธ 1 ครั้งเวลา 23.00 วันอังคาร 3 ครั้งเวลา 06.00,11.30,17.58	Chennel 1 ไม่ทำงาน ในทุกเวลาที่ตั้งไว้
4	2	วันจันทร์ 1 ครั้ง เวลา 07.00 วันอังคาร 3 ครั้งเวลา 12.00,13.11,15.56 วันศุกร์ 2 ครั้ง เวลา 11.25,17.52	Chennel 2 ไม่ทำงาน ในทุกเวลาที่ตั้งไว้
5	2	วันพุธ 2 ครั้งเวลา 05.07,05.09 วันพฤหัส 1 ครั้งเวลา 18.55	Chennel 2 ไม่ทำงาน ในทุกเวลาที่ตั้งไว้
6	2	วันเสาร์ 4 ครั้งเวลา 00.00,00.55,17.22,18.01 วันอาทิตย์ 1 ครั้งเวลา 11.33	Chennel 2 ไม่ทำงาน ในทุกเวลาที่ตั้งไว้

หมายเหตุ หลังจากทำตามคำสั่งเสร็จเครื่องจะรอการกดคำสั่งต่อไปอีกครั้ง ยกเว้นการกด 4

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การทดลองการเปลี่ยนหมายเลข Password ที่ตั้งไว้เดิมโดยการกดคำสั่ง 3

ครั้งที่	หมายเลข Password ที่ตั้งใหม่	หมายเลข Password เดิม	ผลจากการใช้หมายเลข Password เดิม	ผลจากการใช้หมายเลข Password ที่ตั้งใหม่
1	123	111	เครื่องแสดงข้อความ INVALID PASSWORD	เครื่องแสดงข้อความ YOU ARE WELCOME
2	178	122	เครื่องแสดงข้อความ INVALID PASSWORD	เครื่องแสดงข้อความ YOU ARE WELCOME
3	023	023	เครื่องแสดงข้อความ YOU ARE WELCOME	เครื่องแสดงข้อความ YOU ARE WELCOME

หมายเหตุ หลังจากทำตามคำสั่งเสร็จเครื่องจะรอการกดคำสั่งต่อไปอีกครั้ง ยกเว้นการกด 4

- การทดลองการสั่งการวางหูจากโทรศัพท์โดยการกด 4

ครั้งที่	ผลจากคำสั่ง
1	เครื่องสั่งวางหูทันที
2	เครื่องสั่งวางหูทันที
3	เครื่องสั่งวางหูทันที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.3 การทดลองส่วนการสั่งงานการทำงานอัตโนมัติที่ตัวเครื่อง

- การทดลองการเที่ยงตรงของการแสดงวันเวลาเดือนปี

ครั้งที่	เวลาเริ่มต้น	ระยะเวลาการจับเวลา	เวลาที่เครื่องแสดง	ระยะเวลาการจับเวลา	เวลาที่เครื่องแสดง
1	Wed 23.55 30/03/05	10 นาที	Thu 00.05 31/03/05	30 นาที	Thu 00.25 31/03/05
2	Mon 22.00 28/02/05	3.50 ชั่วโมง	Tue 01.50 01/03/05	4 ชั่วโมง	Tue 02.00 01/03/05
3	Sun 06.00 06/11/05	15 นาที	Sun 06.15 06/11/05	20 นาที	Sun 06.20 06/11/05

- การทดลองการใส่ค่าAdmin Password และ Password

กรณีที่ไม่เคยใส่ Admin Password ถูกต้องมาก่อน

รหัส Admin Password ของเครื่องคือ 123456

ครั้งที่	คำสั่งที่กด	การใส่ ADMIN	ผลลัพธ์ที่ได้	รหัส Password ที่ตั้ง	ผลจากการใช้ Password
1	1	123457	กลับไปเสดงวันเวลา	_	_
		123456	เครื่องให้ตั้ง Password แล้วเข้าสู่การตั้งวันวันที่เดือนปี	123	เครื่องแสดงข้อความ YOU ARE WELCOME
2	2	111111	กลับไปเสดงวันเวลา	_	_
		123456	เครื่องให้ตั้ง Password แล้วเข้าสู่การตั้งเวลา	111	เครื่องแสดงข้อความ YOU ARE WELCOME
3	3	654321	กลับไปเสดงวันเวลา	_	_
		123456	เครื่องให้ตั้ง Password	789	เครื่องแสดงข้อความ YOU ARE WELCOME
4	4	456456	กลับไปเสดงวันเวลา	_	_
		123456	เครื่องให้ตั้ง Password แล้วเข้าสู่การตั้งโปรแกรมการให้อาหาร	0*#	เครื่องแสดงข้อความ YOU ARE WELCOME

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กรณีที่เคยใส่ Admin Password ได้ถูกต้องแล้วนับจากเปิดเครื่อง

คำสั่งที่กด	หมายเลข Pasword	การใส่หมายเลข Password	ผลลัพธ์ที่ได้
1	123	111	ให้ใส่หมายเลข Password อีก 3 ครั้ง จนกว่าจะถูกผิดครบ 3 ครั้ง จะกลับไปแสดง วัน เวลา ครั้งต่อไปจะถาม Admin
		123	เข้าสู่การตั้งวันวันที่เดือนปี
2	456	546	ให้ใส่หมายเลข Password อีก 3 ครั้ง จนกว่าจะถูกผิดครบ 3 ครั้ง จะกลับไปแสดง วัน เวลา ครั้งต่อไปจะถาม Admin
		456	เข้าสู่การตั้งเวลา
3	789	111	ให้ใส่หมายเลข Password อีก 3 ครั้ง จนกว่าจะถูกผิดครบ 3 ครั้ง จะกลับไปแสดง วัน เวลา ครั้งต่อไปจะถาม Admin
		789	เครื่องให้ตั้งPassword ใหม่
4	0*#	**7	ให้ใส่หมายเลข Password อีก 3 ครั้ง จนกว่าจะถูกผิดครบ 3 ครั้ง จะกลับไปแสดง วัน เวลา ครั้งต่อไปจะถาม Admin
		0*#	เข้าสู่การตั้งโปรแกรมการให้อาหาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การทดลองการตั้งวัน วันที่ เดือน ปีจากการกดคำสั่ง 1 ที่ Keypad ที่ตัวเครื่อง หลังจากการกด Admin Password หรือPassword ได้ถูกต้องแล้ว

ครั้งที่	วัน วัน ที่ เดือน ปี ที่ตั้ง	วัน วัน ที่ เดือน ปี ที่แสดง
1	Mon 12/04/83	Mon 12/04/83
2	Sat 28/02/99	Sat 28/02/99
3	Tue 01/05/77	Tue 01/05/77

- การทดลองการตั้งเวลาจากการกดคำสั่ง 2 ที่ Keypad ที่ตัวเครื่อง หลังจากการกด Admin Password หรือPassword ได้ถูกต้องแล้ว

ครั้งที่	เวลา ที่ตั้ง	เวลา ที่แสดง
1	11.22	11.22
2	23.33	23.33
3	00.00	00.00

- การทดลองการตั้ง Password จากการกดคำสั่ง 3 ที่ Keypad ที่ตัวเครื่อง หลังจากการกด Admin Password หรือPassword ได้ถูกต้องแล้ว

ครั้งที่	Password เดิม	Password ใหม่ที่ตั้ง	การใส่หมายเลข Password	ผลลัพธ์ที่ได้
1	111	123	111	เครื่องแสดงข้อความ INVALID PASSWORD
			123	เครื่องแสดงข้อความ YOU ARE WELCOME
2	456	0**	456	เครื่องแสดงข้อความ INVALID PASSWORD
			0**	เครื่องแสดงข้อความ YOU ARE WELCOME
3	789	222	999	เครื่องแสดงข้อความ INVALID PASSWORD
			222	เครื่องแสดงข้อความ YOU ARE WELCOME

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การทดลองการตั้ง เวลา,จำนวนครั้ง และปริมาณอาหารที่จะแต่ละวันในแต่ละ Channel จากการกดคำสั่ง 4 ที่ Keypad ที่ตัวเครื่อง หลังจากการกด Admin Password หรือPassword ได้ถูกต้องแล้ว

Channel	การตั้งโปรแกรมการให้อาหารในแต่ละวัน	การทำงานตามโปรแกรมที่ตั้งไว้ในแต่ละวัน
1	วันอาทิตย์ 1 ครั้ง 21.00 หมุน 2 รอบ	วันอาทิตย์ เครื่องให้อาหารเวลา21.00 หมุน 2 รอบ
	วันจันทร์ 3 ครั้ง 06.00 หมุน 5 รอบ 12.00 หมุน 9 รอบ 14.00 หมุน 1 รอบ	วันจันทร์ เครื่องให้อาหารเวลา 06.00 หมุน 5 รอบ 12.00 หมุน 9 รอบ 14.00 หมุน 1 รอบ
	วันอังคาร 2 ครั้ง 02.00 หมุน 7 รอบ 11.23 หมุน 3 รอบ	วันอังคาร เครื่องให้อาหารเวลา 02.00 หมุน 7 รอบ 11.23 หมุน 3 รอบ
	วันพุธ 4 ครั้ง 11.23 หมุน 5 รอบ 12.36 หมุน 4 รอบ 15.55 หมุน 1 รอบ 18.31 หมุน 3 รอบ	วันพุธ เครื่องให้อาหารเวลา 11.23 หมุน 5 รอบ 12.36 หมุน 4 รอบ 15.55 หมุน 1 รอบ 18.31 หมุน 3 รอบ
	วันพฤหัส 1 ครั้ง 23.33 หมุน 1 รอบ	วันพฤหัส เครื่องให้อาหารเวลา 23.33 หมุน 1 รอบ
	วันศุกร์ 3 ครั้ง 10.00 หมุน 1 รอบ 16..02 หมุน 6 รอบ 17.25 หมุน 3 รอบ	วันศุกร์ เครื่องให้อาหารเวลา 10.00 หมุน 1 รอบ 16..02 หมุน 6 รอบ 17.25 หมุน 3 รอบ
	วันเสาร์ 2 ครั้ง 00.00 หมุน 1 รอบ 00.02 หมุน 6 รอบ	วันเสาร์ เครื่องให้อาหารเวลา 00.00 หมุน 1 รอบ 00.02 หมุน 6 รอบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Channel	การตั้งโปรแกรมการให้อาหารในแต่ละวัน	การทำงานตามโปรแกรมที่ตั้งไว้ในแต่ละวัน
2	วันอาทิตย์ 2 ครั้ง 01.00 หมุน 1 รอบ 04.22 หมุน 5 รอบ	วันอาทิตย์ เครื่องให้อาหารเวลา01.00 หมุน 1 รอบ 04.22 หมุน 5 รอบ
	วันจันทร์ 4 ครั้ง 13.11 หมุน 8 รอบ 15.59 หมุน 1 รอบ 18.22 หมุน 7 รอบ 19.00 หมุน 2 รอบ	วันจันทร์ เครื่องให้อาหารเวลา 06.00 หมุน 8 รอบ 12.00 หมุน 1 รอบ 14.00 หมุน 7 รอบ 19.00 หมุน 2 รอบ
	วันอังคาร 1 ครั้ง 12.58 หมุน 2 รอบ	วันอังคาร เครื่องให้อาหารเวลา 12.58 หมุน 2 รอบ
	วันพุธ 3 ครั้ง 05.28 หมุน 9 รอบ 22.33 หมุน 2 รอบ 23.55 หมุน 7 รอบ	วันพุธ เครื่องให้อาหารเวลา 05.28 หมุน 9 รอบ 22.33 หมุน 2 รอบ 23.55 หมุน 7 รอบ
	วันพฤหัส 4 ครั้ง 13.26 หมุน 7 รอบ 17.33 หมุน 1 รอบ 18.28 หมุน 6 รอบ 23.05 หมุน 9 รอบ	วันพฤหัส เครื่องให้อาหารเวลา 13.26 หมุน 7 รอบ 17.33 หมุน 1 รอบ 18.28 หมุน 6 รอบ 23.05 หมุน 9 รอบ
	วันศุกร์ 2 ครั้ง 21.36 หมุน 7 รอบ 00.29 หมุน 3 รอบ	วันศุกร์ เครื่องให้อาหารเวลา 21.36 หมุน 7 รอบ 00.29 หมุน 3 รอบ
	วันเสาร์ 3 ครั้ง 07.59 หมุน 6 รอบ 19.22 หมุน 2 รอบ	วันเสาร์ เครื่องให้อาหารเวลา 07.59 หมุน 6 รอบ 19.22 หมุน 2 รอบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.4 การทดลองวัดปริมาณอาหารที่ให้ต่อการหมุน 1 รอบ

- การทดลองวัดปริมาณอาหารที่ให้ต่อการหมุน 1 รอบ ของ Channel 1

ครั้งที่ 1	ปริมาณอาหารที่ให้ (เม็ด)
1	61
2	64
3	62
4	63
5	65
6	61
7	63
8	67
9	63
10	62

- การทดลองวัดปริมาณอาหารที่ให้ต่อการหมุน 1 รอบ ของ Channel 2

ครั้งที่ 1	ปริมาณอาหารที่ให้ (เม็ด)
1	108
2	110
3	107
4	100
5	105
6	102
7	107
8	100
9	113
10	107

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6
สรุปและวิเคราะห์ผลการทดลอง

6.1 บทสรุป

เครื่องให้อาหารปลาอัตโนมัติสามารถทำงานจากการสั่งงานทางโทรศัพท์และทำงานอัตโนมัติจากการตั้งโปรแกรมที่ตัวเครื่องโดยใช้ไมโครคอนโทรลเลอร์เป็นศูนย์กลางการประมวลผลและสั่งการได้ตามขอบเขตที่ตั้งไว้ทุกประการ และมีความเที่ยงตรงและแม่นยำในการทำงานสูงสังเกตได้จากผลการทดลองที่ได้

อย่างไรก็ตาม ยังพบความไม่เท่ากันของปริมาณอาหารที่ให้ต่อการหมุน 1 รอบ ของเครื่องของทั้ง 2 Channel แต่ก็มีความแตกต่างกันไม่มากซึ่งอยู่ในเกณฑ์ที่ยอมรับได้

6.2 วิเคราะห์ผลการทดลอง

สาเหตุของปริมาณอาหารที่ให้ต่อการหมุน 1 รอบของทั้ง 2 Channel มีความแตกต่างกันน่าจะเกิดจากการที่ขนาดของเม็ดอาหารแต่ละเม็ดมีขนาดแตกต่างกันเล็กน้อย จึงทำให้จำนวนเม็ดอาหารที่ให้แตกต่างกันบ้างเล็กน้อย

6.3 แนวทางการพัฒนา

เนื่องจากเครื่องให้อาหารปลาอัตโนมัติและสั่งงานทางโทรศัพท์ ใช้ฐานเวลาจริงในการทำงาน และการปิดเปิดการให้อาหาร เกิดจากการส่งสัญญาณจากไมโครคอนโทรลเลอร์ไปสั่งการวงจรทริกรีเลย์ดังนั้นเครื่องให้อาหารปลาอัตโนมัติและสั่งงานทางโทรศัพท์จึงสามารถนำไปประยุกต์ใช้กับการควบคุมการเปิดปิดอุปกรณ์ไฟฟ้าได้ทุกชนิดจากการตั้งโปรแกรมที่ตัวเครื่องและสั่งงานผ่านโทรศัพท์ โดยแก้ไขในส่วนโปรแกรมของไมโครคอนโทรลเลอร์และวงจรการเชื่อมต่อเครื่องใช้ไฟฟ้าบางส่วน โดยอาจเพิ่มจำนวน Channel เพื่อให้สามารถควบคุมเครื่องใช้ไฟฟ้าได้มากจำนวนขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

1.วรพจน์ กรแก้ววัฒนกุล , ชัยวัฒน์ ลิ้มพรจิตรวิไล, " MCS-51 Microcontroloer Theory & Practical Approach:Atmel AT89C5x", Innovative Experiment Co.,Ltd.

2.นคร ภักดีชาติ , ธีรบูลย์ หล่อวิเชียรรุ่ง , ชัยวัฒน์ ลิ้มพรจิตรวิไล, "ปฏิบัติการไมโครคอนโทรลเลอร์ MCS-51 ด้วยโปรแกรมภาษา C", Innovative Experiment Co.,Ltd.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*-------------------------------------------------------------------------------*/
// Program  for AUTOMATIC FEEDING FISH MACHINE AND CONTROLABLE BY TELEPHONE
// CHANCHAI USSAWAPUSITGUL
// NATTAPONG JULLABUD
// FACULTY OF ENGINEERING
// KING MONGKUT 'S INSTITUTE OF TECHNOLOGY LADKRABANG
//
/*-------------------------------------------------------------------------------*/
#include<reg52.h>
#include<i2c.h>                    // Module function i2c bus
#include<lcd.h>                    // Module function LCD
#include<scankey.h>              // Module function scankey 4x3
#include<scankeyx.h>
#include<password.h>
#define DS1307_ID 0xD0             // Define constant DS1307_ID
unsigned char sec,min,hour,day,date,month,year;
code unsigned char *start1 = "AUTOMATIC FEEDING MACHINE";
code unsigned char *start2 = "  KING MONGKUT  ";
code unsigned char *start3 = "   LADKRABANG   ";
code unsigned char *setdate = "Setting Date";
code unsigned char *dmy = "dd/mm/yy";
code unsigned char *settime = "Setting Time";
code unsigned char *hms = "hh:mm:ss";
code unsigned char *sun = "Sun";
code unsigned char *mon = "Mon";
code unsigned char *tue = "Tue";
code unsigned char *wed = "Wed";
code unsigned char *thu = "Thu";
code unsigned char *fri = "Fri";
code unsigned char *sat = "Sat";
code unsigned char *setday = "Sun(1) - Sat(7):";
code unsigned char *feed = " SETUP FOR FEED ";
code unsigned char *channel = " SELECL CHANNEL ";
code unsigned char *dayfeed = "all(*) cancel(#)";
code unsigned char *suntimes = "SUNDAY    TIMES ";
code unsigned char *montimes = "MONDAY    TIMES ";
code unsigned char *tuetimes = "TUESDAY   TIMES ";
code unsigned char *wedtimes = "WEDNESDAY TIMES ";
code unsigned char *thutimes = "THURSDAY  TIMES ";
code unsigned char *fritimes = "FRIDAY    TIMES ";
code unsigned char *sattimes = "SATURDAY  TIMES ";
code unsigned char *u = "). :    ";
code unsigned char *loop = " SPIN      ROUND";
code unsigned char *teletext = "  TELEPHONE !!  ";
code unsigned char *wait_dail= "WAIT PRESS TELE!";
code unsigned char *stop = " STOP CHANNEL   ";
code unsigned char *stop_tele = "STOP TELEPHONE!!";
unsigned char select=0;
unsigned char ch,timef;
unsigned char dayf;
unsigned char push_key = 0;        // Keep for value push key
unsigned char i,j;
unsigned char x=0;
unsigned char k=0;
unsigned char x_key = 0;
unsigned char date_buf[7];
unsigned char time_buf[7];
unsigned char ontime[4];
unsigned char chktelepass;
idata unsigned char sun_h_1[4];
idata unsigned char sun_m_1[4];
idata unsigned char sun_h_2[4];
idata unsigned char sun_m_2[4];
idata unsigned char sun_loop1[4];
idata unsigned char sun_loop2[4];
idata unsigned char mon_h_1[4];
idata unsigned char mon_m_1[4];
idata unsigned char mon_h_2[4];
idata unsigned char mon_m_2[4];
idata unsigned char mon_loop1[4];
idata unsigned char mon_loop2[4];
idata unsigned char tue_h_1[4];
idata unsigned char tue_m_1[4];
idata unsigned char tue_h_2[4];
idata unsigned char tue_m_2[4];
idata unsigned char tue_loop1[4];
idata unsigned char tue_loop2[4];
idata unsigned char wed_h_1[4];
idata unsigned char wed_m_1[4];
idata unsigned char wed_h_2[4];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
idata unsigned char wed_m_2[4];
idata unsigned char wed_loop1[4];
idata unsigned char wed_loop2[4];
idata unsigned char thu_h_1[4];
idata unsigned char thu_m_1[4];
idata unsigned char thu_h_2[4];
idata unsigned char thu_m_2[4];
idata unsigned char thu_loop1[4];
idata unsigned char thu_loop2[4];
idata unsigned char fri_h_1[4];
idata unsigned char fri_m_1[4];
idata unsigned char fri_h_2[4];
idata unsigned char fri_m_2[4];
idata unsigned char fri_loop1[4];
idata unsigned char fri_loop2[4];
idata unsigned char sat_h_1[4];
idata unsigned char sat_m_1[4];
idata unsigned char sat_h_2[4];
idata unsigned char sat_m_2[4];
idata unsigned char sat_loop1[4];
idata unsigned char sat_loop2[4];
unsigned char page = 20;
sbit press = P1^7;
sbit ack = P1^4;
sbit tele = P1^2;
sbit drive1 = P3^4;
sbit chkdrive1 = P3^5;
sbit drive2 = P1^6;
sbit chkdrive2 = P1^5;


void set1(void)
{
i=0;
lcd_clear();
for(s=0;s<16;s++)
lcd_text(*(loop+s));
lcd_command(0x87);
lcd_text(x|0x30);
}

void set2(void)
{
drive1=1;
if(chkdrive1==1)
i++;
while(chkdrive1==1);
lcd_command(0xC7);
lcd_text(i|0x30);
}

void set3(void)
{
drive2=1;
if(chkdrive2==1)
i++;
while(chkdrive2==1);
lcd_command(0xC7);
lcd_text(i|0x30);
}
/*****************************************************************************/
/*********************** Function convert to ascii and send to LCD ***********/
/*****************************************************************************/
void send_to_lcd(unsigned char value)
{
    unsigned char buf = 0;
    buf = value & 0xF0;
    buf = (buf>>4)|(0x30);          // Shift to 4 time(swap) OR 0x30 for convert to ascii c

    lcd_text(buf);                  // Send to show LCD
    buf = value & 0x0F;               // Filter for low byte
    buf = buf | 0x30;               // OR 0x30 for convert to ascii code
    lcd_text(buf);                      // Send to show LCD
}
/////////////////////////////////////////////////////////////////
void check(void)
{
switch(day)
{
    case 1 :   for(j=0;j<4;j++)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                if((sun_h_1[j]==hour)&&(sun_m_1[j]==min))
                {
                 x=sun_loop1[j];
                  set1();

                while(i<sun_loop1[j])
                   {
                      set2();
                   }
                    lcd_clear();
                   drive1=0;

                }

              if((sun_h_2[j]==hour)&&(sun_m_2[j]==min))
                   {
                    x=sun_loop2[j];
                    set1();

                   while(i<sun_loop2[j])
                      {
                         set3();
                       }
                       lcd_clear();
                       drive2=0;

                   }
                 }break;
case 2 :         for(j=0;j<4;j++)
                 {
                   if((mon_h_1[j]==hour)&&(mon_m_1[j]==min))
                   {
                    x=mon_loop1[j];
                    set1();
                   while(i<mon_loop1[j])
                      {
                         set2();
                       }
                        lcd_clear();
                       drive1=0;

                   }
                 if((mon_h_2[j]==hour)&&(mon_m_2[j]==min))
                   {
                    x=mon_loop2[j];
                    set1();
                   while(i<mon_loop2[j])
                      {
                         set3();
                       }
                       lcd_clear();
                       drive2=0;

                   }
                 }break;

case 3 :         for(j=0;j<4;j++)
                 {
                   if((tue_h_1[j]==hour)&&(tue_m_1[j]==min))
                   {

                    x=tue_loop1[j];
                     set1();

                   while(i<tue_loop1[j])
                      {
                         set2();
                       }
                        lcd_clear();
                       drive1=0;

                    }

                if((tue_h_2[j]==hour)&&(tue_m_2[j]==min))
                   {

                    x=tue_loop2[j];
                    set1();
                   while(i<tue_loop2[j])
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                {
                   set3();
                }
                lcd_clear();
                drive2=0;

              }
            }break;
case 4 :    for(j=0;j<4;j++)
            {
              if((wed_h_1[j]==hour)&&(wed_m_1[j]==min))
              {

               x=wed_loop1[j];
                set1();

              while(i<wed_loop1[j])
                 {
                    set2();
                 }
                  lcd_clear();
                 drive1=0;

              }

           if((wed_h_2[j]==hour)&&(wed_m_2[j]==min))
               {
                x=wed_loop2[j];
                 set1();

               while(i<wed_loop2[j])
                  {
                     set3();
                   }
                   lcd_clear();
                   drive2=0;

               }
             }break;
case 5 :    for(j=0;j<4;j++)
            {
              if((thu_h_1[j]==hour)&&(thu_m_1[j]==min))
              {
               x=thu_loop1[j];
                set1();

              while(i<thu_loop1[j])
                 {
                    set2();
                 }
                  lcd_clear();
                 drive1=0;

              }

           if((thu_h_2[j]==hour)&&(thu_m_2[j]==min))
               {

                x=thu_loop2[j];
                 set1();
               while(i<thu_loop2[j])
                  {
                     set3();
                   }
                   lcd_clear();
                   drive2=0;

               }
             }break;
case 6 :    for(j=0;j<4;j++)
            {
              if((fri_h_1[j]==hour)&&(fri_m_1[j]==min))
              {

               x=fri_loop1[j];
                set1();

              while(i<fri_loop1[j])
                 {
                    set2();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    }
                     lcd_clear();
                    drive1=0;

                  }

              if((fri_h_2[j]==hour)&&(fri_m_2[j]==min))
                  {

                   x=fri_loop2[j];
                    set1();
                  while(i<fri_loop2[j])
                     {
                        set3();
                      }
                      lcd_clear();
                      drive2=0;

                  }
                }break;
    case 7 :    for(j=0;j<4;j++)
                {
                  if((sat_h_1[j]==hour)&&(sat_m_1[j]==min))
                  {
                   x=sat_loop1[j];
                    set1();
                  while(i<sat_loop1[j])
                     {
                        set2();
                      }
                      lcd_clear();
                      drive1=0;

                   }

                if((sat_h_2[j]==hour)&&(sat_m_2[j]==min))
                   {

                    x=sat_loop2[j];
                    set1();

                   while(i<sat_loop2[j])
                     {
                        set3();
                      }
                      lcd_clear();
                      drive2=0;

                   }
                 }break;
}
}

/*****************************************************************************/
/*********************** Function read data from DS1307 **********************/
/*****************************************************************************/
unsigned char DS1307_rd(unsigned char addr)
{
    unsigned char ret;                  // For keep return value
    i2c_start();                        // i2c start condition
    i2c_wrdata(DS1307_ID);             // Write DS1307 ID for connect
    i2c_wrdata(addr);                       // Write RAM address on DS1307 for connect

    i2c_start();                            // i2c start condition
    i2c_wrdata(DS1307_ID+1);           // Write DS1307 ID for Read Mode connect

    ret = i2c_rddata();                // Read data and keep to ret
   i2c_stop();                              // i2c stop condition
   return(ret);                         // return value
}
/*****************************************************************************/
/****************** Function write hour/min/sec on chip DS1307 ***************/
/*****************************************************************************/
void DS1307_wrtime()
{
    hour = (time_buf[1]<<4)|(time_buf[2]);          // Convet input hour value from key t
    min=(time_buf[3]<<4)|(time_buf[4]);             // Convet input min value from key to
    sec=(time_buf[5]<<4)|(time_buf[6]);             // Convet input sec value from key to
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    hour = 0x23;                        // IF hour over range reload maximum value
    if(min>0x59)                            // Check min over range
    min = 0x59;                             // IF min over range reload maximum value
    if(sec>0x59)                            // Check sec over range
    sec = 0x59;                             // IF sec over range reload maximum value
    i2c_start();                    // i2c start condition
    i2c_wrdata(DS1307_ID);              // Write DS1307 ID for connect
    i2c_wrdata(0x00);                       // Write control byte to access RAM address 00H
    i2c_wrdata(sec);                        // Write sec on RAM address 00H
    i2c_wrdata(min);                        // Write min on RAM address 01H
    i2c_wrdata(hour);                       // Write hour on RAM address 02H
    i2c_stop();                     // i2c stop condition
}
/*****************************************************************************/
/************************ Function write date/month/year on chip DS1307 **************/
/*****************************************************************************/
void DS1307_wrdate()
{
    date = (date_buf[1]<<4)|(date_buf[2]);          // Convet input date value from key to

    month = (date_buf[3]<<4)|(date_buf[4]);     // Convet input month value from key to He

    year = (date_buf[5]<<4)|(date_buf[6]);          // Convet input year value from key to

    i2c_start();                            // i2c start condition
    i2c_wrdata(DS1307_ID);              // Write DS1307 ID for connect
    i2c_wrdata(0x03);                       // Write control byte to access RAM address 04H
    i2c_wrdata(day);
    i2c_wrdata(date);                       // Write date on RAM address 04H
    i2c_wrdata(month);                  // Write month on RAM address 05H
    i2c_wrdata(year);                       // Write year on RAM address 06H
    i2c_stop();                             // i2c stop condition
}
/*****************************************************************************/
/************************ Function display screen on LCD*************************/
/*****************************************************************************/
void showstart(void)
{
    /*--------------Show text screen on LCD----------------*/

    lcd_command(0x07);
    lcd_command(0x8F);              // Set LCD address
    for(i=0;i<25;i++)               // Loop for show screen "Date" by pointer _Date
    {
    lcd_text(*(start1+i));
    delay(30);
    }
    delay(500);
    lcd_clear();
    lcd_command(0x0C);
    lcd_command(0x06);
    lcd_command(0x80);
    for(i=0;i<16;i++)               // Loop for show screen "Date" by pointer _Date
    lcd_text(*(start2+i));
    lcd_command(0xC0);
    for(i=0;i<16;i++)               // Loop for show screen "Date" by pointer _Date
    lcd_text(*(start3+i));
    delay(500);
    lcd_clear();
    }

/************************ Display Normal(Page1) ****************************/
//------------------------ Date  dd/mm/yy-----------------------------------//
//------------------------ Time  hh/mm/ss-----------------------------------//
/*****************************************************************************/
void display_datetime(void)
{
        sec = DS1307_rd(0);         // Read sec before show on LCD
        min = DS1307_rd(1);         // Read min before show on LCD
        hour = DS1307_rd(2);            // Read hour before show on LCD
        day = DS1307_rd(3);         // Read day before show on LCD
        date = DS1307_rd(4);            // Read date before show on LCD
        month = DS1307_rd(5);       // Read month before show on LCD
        year = DS1307_rd(6);            // Read year before show on LCD


    /*---------------------------Show Date-------------------------*/

    switch(day)
    {
        case 1 :  lcd_command(0x82);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    lcd_text(*(sun+i));
                    break;
          case 2 :  lcd_command(0x82);
                    for(i=0;i<3;i++)
                     lcd_text(*(mon+i));
                     break;
          case 3 :  lcd_command(0x82);
                    for(i=0;i<3;i++)
                     lcd_text(*(tue+i));
                     break;
          case 4 :  lcd_command(0x82);
                    for(i=0;i<3;i++)
                     lcd_text(*(wed+i));
                     break;
          case 5 :  lcd_command(0x82);
                    for(i=0;i<3;i++)
                     lcd_text(*(thu+i));
                     break;
          case 6 :  lcd_command(0x82);
                    for(i=0;i<3;i++)
                     lcd_text(*(fri+i));
                     break;
          case 7 :  lcd_command(0x82);
                    for(i=0;i<3;i++)
                     lcd_text(*(sat+i));
                     break;
        }
    lcd_command(0x87);
    send_to_lcd(date);          // Write date show on LCD
    lcd_text('/');                 // Write "/" show on LCD
    send_to_lcd(month);         // Write month show on LCD
    lcd_text('/');                 // Write "/" show on LCD
    send_to_lcd(year);          // Write year show on LCD
    /*-------------------------------------------------------------*/
    /*-------------------------Show Time---------------------------*/
    lcd_command(0xC7);          // Set LCD address 46H
    send_to_lcd(hour);          // Write hour show on LCD
    lcd_text(':');
    lcd_command(0xCA);
    send_to_lcd(min);              // Write min show on LCD
    /*-------------------------------------------------------------*/
}
/*********************** Display Setting Date(Page2) ****************************/
//---------------------- Setting Date ------------------------------------------//
//---------------------- dd/mm/yy ----------------------------------------------//
/********************************************************************************/
void display_setdate(void)
{
    /*--------------Show text screen on LCD("Setting Date")----------------*/
    lcd_clear();                   // Clear LCD display
    lcd_command(0x82);          // Set LCD address 02H
    for(i=0;i<12;i++)              // Loop for show screen "Setting Date" by pointer setd

    lcd_text(*(setdate+i));
    /*-------------------------------------------------------------------*/
    /*--------------Show text screen on LCD("dd/mm/yy")------------------*/
    lcd_command(0xC3);          // Set LCD address 43H
    for(i=0;i<8;i++)               // Loop for show screen "dd/mm/yy" by pointer dmy
    lcd_text(*(dmy+i));
    /*-------------------------------------------------------------------*/
    /*------------ Return to set curcer bring for insert value ----------*/
    lcd_command(0xC3);          // Set LCD address 43H
    lcd_command(0x0F);          // Set display ON and curcer bring
    /*-------------------------------------------------------------------*/
}
/**********************Display Setting Time(Page3)****************************/
//---------------------- Setting Time ------------------------------------------//
//---------------------- hh/mm/ss ----------------------------------------------//
/********************************************************************************/
void display_settime(void)
{
    /*--------------Show text screen on LCD("Setting Time")----------------*/
    lcd_clear();                   // Clear LCD display
    lcd_command(0x82);          // Set LCD address 02H
    for(i=0;i<12;i++)              // Loop for show screen "Setting Time" by pointer sett

    lcd_text(*(settime+i));
    /*-------------------------------------------------------------------*/
    /*--------------Show text screen on LCD("hh/mm/ss")------------------*/
    lcd_command(0xC3);          // Set LCD address 43H
    for(i=0;i<8;i++)               // Loop for show screen "hh/mm/ss" by pointer hms
    lcd_text(*(hms+i));
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    /*-------------------------------------------------------------------*/
    /*------------ Return to set curcer bring for insert value ----------*/
    lcd_command(0x0F);          // Display ON and OFF curcer bring
    lcd_command(0xC3);          // Set LCD address 43H
    /*-------------------------------------------------------------------*/
}
//*****************************************************
void support(void)
{
 if((date_buf[1]==3)&&(date_buf[2]==1)&&(x_key==4))
 {
     x_key=0;
     lcd_command(0xC7);
     lcd_text('!');
     delay(200);
     display_setdate();
 }
}
//*****************************************************

void protect(unsigned char x,unsigned char ad)
{
    x_key=x;
    lcd_command(ad);
    lcd_text('!');
    lcd_command(ad);

}

//*********************************************************
void func_set1(void)
{
lcd_command(0x0F);
lcd_command(0x89);
i=0xFF;
while(((i==0xFF)||(i>4))&&(i!=12))
{
i=scankey();
    if((i>4)&&(i!=0xFF))
    {
        lcd_command(0x89);
        lcd_text('!');
        lcd_command(0x89);
    }
    if(i<5)
    timef=i;
}

}

////////////////////////////////////////////////////////////////////////
void func_set2(void)
{   lcd_clear();
    lcd_command(0x0C);
    lcd_command(0x80);
    lcd_text((j+1)+0x30);
    for(i=0;i<8;i++)            // Loop for show screen
    lcd_text(*(u+i));
    lcd_command(0x0F);
    lcd_command(0x83);
    x_key=0;
    while(x_key<=3)
    {
        push_key=scankey();
    if((push_key != 10)&&(push_key != 12)&&(push_key != 0xFF))
    {
        ontime[x_key] = push_key;
        x_key++;
        if(x_key==2)
            lcd_command(0x86);
        if(ontime[0]>2)
            protect(0,0x83);
         if((ontime[0]==2)&&(ontime[1]>=4))
            protect(1,0x84);
        if(ontime[2]>=6)
            protect(2,0x86);
    }
    if(push_key==12)
    x_key=4;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    j=timef;
    }
    if(push_key == 10)
    {
        x_key=0;
        lcd_command(0x83);
    }
    }

}

//*****************************************************************************
void ad_pass(void)
{
    if(select==0)
    {
    i=pass();
        if(i==1)
        setpass();
    }
    else
    {i=0;
     x=0;
        while((i==0)&&(x<3))
        {
            x++;
            i=pass2();
            if(x==3)
            select=0;
        }
    }
}

//*****************************************************************************
void wr_loop(void)
{
    for(i=0;i<16;i++)              // Loop for show screen
    lcd_text(*(loop+i));
    lcd_command(0x87);
}


//*********************************************************
void set_sun(void)
{
    for(i=0;i<16;i++)              // Loop for show screen
    lcd_text(*(suntimes+i));
    func_set1();
    if(push_key!=12)
    {
    for(j=0;j<timef;j++)
    {
    func_set2();

    if(push_key!=12)
    {
    lcd_command(0x80);
    wr_loop();
    if(ch==1)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            sun_h_1[x]=0xFF;
            sun_m_1[x]=0xFF;
        }
        sun_h_1[j]=(ontime[0]<<4)|(ontime[1]);
        sun_m_1[j]=(ontime[2]<<4)|(ontime[3]);
        sun_loop1[j]=0xFF;
        lcd_command(0x87);
        while(sun_loop1[j]==0xFF)
        sun_loop1[j]=scankey();
    }
    if(ch==2)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            sun_h_2[x]=0xFF;
            sun_m_2[x]=0xFF;
        }
        sun_h_2[j]=(ontime[0]<<4)|(ontime[1]);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        sun_m_2[j]=(ontime[2]<<4)|(ontime[3]);
        sun_loop2[j]=0xFF;
        lcd_command(0x87);
        while(sun_loop2[j]==0xFF)
        sun_loop2[j]=scankey();
    }

    }
    }
    }
}

////////////////////////////////////////////////////////////////////

void set_mon(void)
{
    for(i=0;i<16;i++)              // Loop for show screen
    lcd_text(*(montimes+i));
    func_set1();
    if(push_key!=12)
    {
    for(j=0;j<timef;j++)
    {
    func_set2();

    if(push_key!=12)
    {
    lcd_command(0x80);
    wr_loop();
    if(ch==1)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            mon_h_1[x]=0xFF;
            mon_m_1[x]=0xFF;
        }
        mon_h_1[j]=(ontime[0]<<4)|(ontime[1]);
        mon_m_1[j]=(ontime[2]<<4)|(ontime[3]);
        mon_loop1[j]=0xFF;
        lcd_command(0x87);
        while(mon_loop1[j]==0xFF)
        mon_loop1[j]=scankey();
    }
    if(ch==2)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            mon_h_2[x]=0xFF;
            mon_m_2[x]=0xFF;
        }
        mon_h_2[j]=(ontime[0]<<4)|(ontime[1]);
        mon_m_2[j]=(ontime[2]<<4)|(ontime[3]);
        mon_loop2[j]=0xFF;
        lcd_command(0x87);
        while(mon_loop2[j]==0xFF)
        mon_loop2[j]=scankey();
    }
    }
    }
    }

////////////////////////////////////////////////////////////////////////////////////////////////
void set_tue()
{
    for(i=0;i<16;i++)              // Loop for show screen
    lcd_text(*(tuetimes+i));
    func_set1();
    if(push_key!=12)
    {
    for(j=0;j<timef;j++)
    {
    func_set2();

    if(push_key!=12)
    {
    lcd_command(0x80);
    wr_loop();
    if(ch==1)
    {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            for(x=0;((x<4)&&(j==0));x++)
            {
                tue_h_1[x]=0xFF;
                tue_m_1[x]=0xFF;
            }
            tue_h_1[j]=(ontime[0]<<4)|(ontime[1]);
            tue_m_1[j]=(ontime[2]<<4)|(ontime[3]);
            tue_loop1[j]=0xFF;
            lcd_command(0x87);
            while(tue_loop1[j]==0xFF)
            tue_loop1[j]=scankey();
        }
        if(ch==2)
        {
            for(x=0;((x<4)&&(j==0));x++)
            {
                tue_h_2[x]=0xFF;
                tue_m_2[x]=0xFF;
            }
            tue_h_2[j]=(ontime[0]<<4)|(ontime[1]);
            tue_m_2[j]=(ontime[2]<<4)|(ontime[3]);
            tue_loop2[j]=0xFF;
            lcd_command(0x87);
            while(mon_loop2[j]==0xFF)
            tue_loop2[j]=scankey();
        }
        }
        }
        }
}
////////////////////////////////////////////////////////////////////
void set_wed(void)
{
    for(i=0;i<16;i++)                    // Loop for show screen
    lcd_text(*(wedtimes+i));
    func_set1();
    if(push_key!=12)
    {
    for(j=0;j<timef;j++)
    {
    func_set2();

    if(push_key!=12)
    {
    lcd_command(0x80);
    wr_loop();
    if(ch==1)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            wed_h_1[x]=0xFF;
            wed_m_1[x]=0xFF;
        }
        wed_h_1[j]=(ontime[0]<<4)|(ontime[1]);
        wed_m_1[j]=(ontime[2]<<4)|(ontime[3]);
        wed_loop1[j]=0xFF;
        lcd_command(0x87);
        while(wed_loop1[j]==0xFF)
        wed_loop1[j]=scankey();
    }
    if(ch==2)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            wed_h_2[x]=0xFF;
            wed_m_2[x]=0xFF;
        }
        wed_h_2[j]=(ontime[0]<<4)|(ontime[1]);
        wed_m_2[j]=(ontime[2]<<4)|(ontime[3]);
        wed_loop2[j]=0xFF;
        lcd_command(0x87);
        while(wed_loop2[j]==0xFF)
        wed_loop2[j]=scankey();
    }
    }
    }
    }
}
////////////////////////////////////////////////////////////////////
void set_thu(void)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
    for(i=0;i<16;i++)           // Loop for show screen
    lcd_text(*(thutimes+i));
    func_set1();
    if(push_key!=12)
    {
    for(j=1;j<timef;j++)
    {
    func_set2();

    if(push_key!=12)
    {
    lcd_command(0x80);
    wr_loop();
    if(ch==1)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            thu_h_1[x]=0xFF;
            thu_m_1[x]=0xFF;
        }
        thu_h_1[j]=(ontime[0]<<4)|(ontime[1]);
        thu_m_1[j]=(ontime[2]<<4)|(ontime[3]);
        thu_loop1[j]=0xFF;
        lcd_command(0x87);
        while(thu_loop1[j]==0xFF)
        thu_loop1[j]=scankey();
    }
    if(ch==2)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            thu_h_2[x]=0xFF;
            thu_m_2[x]=0xFF;
        }
        thu_h_2[j]=(ontime[0]<<4)|(ontime[1]);
        thu_m_2[j]=(ontime[2]<<4)|(ontime[3]);
        thu_loop2[j]=0xFF;
        lcd_command(0x87);
        while(thu_loop2[j]==0xFF)
        thu_loop2[j]=scankey();
    }
    }
    }
    }
}
////////////////////////////////////////////////////////
void set_fri()
{
    for(i=0;i<16;i++)           // Loop for show screen
    lcd_text(*(fritimes+i));
    func_set1();
    if(push_key!=12)
    {
    for(j=0;j<timef;j++)
    {
    func_set2();

    if(push_key!=12)
    {
    lcd_command(0x80);
    wr_loop();
    if(ch==1)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            fri_h_1[x]=0xFF;
            fri_m_1[x]=0xFF;
        }
        fri_h_1[j]=(ontime[0]<<4)|(ontime[1]);
        fri_m_1[j]=(ontime[2]<<4)|(ontime[3]);
        fri_loop1[j]=0xFF;
        lcd_command(0x87);
        while(fri_loop1[j]==0xFF)
        fri_loop1[j]=scankey();
    }
    if(ch==2)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้ในการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                fri_h_2[x]=0xFF;
                fri_m_2[x]=0xFF;
            }
            fri_h_2[j]=(ontime[0]<<4)|(ontime[1]);
            fri_m_2[j]=(ontime[2]<<4)|(ontime[3]);
            fri_loop2[j]=0xFF;
            lcd_command(0x87);
            while(fri_loop2[j]==0xFF)
            fri_loop2[j]=scankey();
        }
        }
        }
        }
}
////////////////////////////////////////////////////
void set_sat(void)
{
    for(i=0;i<16;i++)              // Loop for show screen
    lcd_text(*(sattimes+i));
    func_set1();
    if(push_key!=12)
    {
    for(j=0;j<timef;j++)
    {
    func_set2();

    if(push_key!=12)
    {
    lcd_command(0x80);
    wr_loop();
    if(ch==1)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            sat_h_1[x]=0xFF;
            sat_m_1[x]=0xFF;
        }
        sat_h_1[j]=(ontime[0]<<4)|(ontime[1]);
        sat_m_1[j]=(ontime[2]<<4)|(ontime[3]);
        sat_loop1[j]=0xFF;
        lcd_command(0x87);
        while(sat_loop1[j]==0xFF)
        sat_loop1[j]=scankey();
    }
    if(ch==2)
    {
        for(x=0;((x<4)&&(j==0));x++)
        {
            sat_h_2[x]=0xFF;
            sat_m_2[x]=0xFF;
        }
        sat_h_2[j]=(ontime[0]<<4)|(ontime[1]);
        sat_m_2[j]=(ontime[2]<<4)|(ontime[3]);
        sat_loop2[j]=0xFF;
        lcd_command(0x87);
        while(sat_loop2[j]==0xFF)
        sat_loop2[j]=scankey();
    }
    }
    }
    }
}
////////////////////////////////////////////////////////////////////////////////
void presstele(void)
{
    if(press==1)
    {
    x = P3 & 0x0F;
    lcd_text(x|0x30);
    while(press==1);
    }
}
////////////////////////////////////////////////////////////////////////////////
void passtele(void)
{   num=0;
    lcd_clear();
    lcd_command(0x80);
    for(s=0;s<16;s++)
    lcd_text(*(in2+s));
    lcd_command(0xC1);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
lcd_command(0x0F);

while(num<3)
    {
        if(press==1)
        {
        x = P3 & 0x0F;
        if(x==10)
         x=0;
         if(x==11)
         x=10;
        pass_buf[num]=x;
        delay(10);
        lcd_text('*');
        while(press==1);
        num++;
        }
    lcd_command(0x0C);

     }
 if((pass_buf[3]==pass_buf[0])&&(pass_buf[4]==pass_buf[1])&&(pass_buf[5]==pass_buf[2]))
    {
        lcd_clear();
    lcd_command(0x80);
    for(s=0;s<16;s++)
    lcd_text(*(ok+s));
    delay(100);
    chktelepass=1;

     }
     else
     {
    lcd_clear();
    lcd_command(0x80);
    for(s=0;s<16;s++)
    lcd_text(*(no+s));
    delay(100);
    chktelepass=0;
    num=0;
    }

}
////////////////////////////////////////////////////////////////////////////
void sup_tele(void)
{
lcd_command(0x80);
 lcd_text(x|0x30);
 delay(100);
 // sound
 lcd_clear();
 lcd_command(0x80);
 for(i=0;i<16;i++)                    // Loop for show screen
 lcd_text(*(channel+i));
 lcd_command(0x0F);
 lcd_command(0xC7);
 while(press==0);
 presstele();
 lcd_command(0x0C);
 delay(100);
 while((x>2)||(x==0))
 {
 if((x>2)||(x==0))
    {
        x=0xFF;
        lcd_command(0xC7);
        lcd_text('!');
        lcd_command(0xC7);
        while(press==0);
        presstele();
    }
  }
 ch=x;
}
////////////////////////////////////////////////////////////////////////////

void main(void)                       // Main loop
{
    unsigned char a=0;
    unsigned char min_x;
    ack =0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
P0 = 0x00;
chkdrive1=0;
drive1=0;
chkdrive2=0;
drive2=0;

lcd_init();                          // Initial set LCD display
while(1)                             // Infinite loop
{
/*-------------Scan keypad for select page to show ---------*/
    if(page==0)
    {
    lcd_command(0x80);
    page=scankey();
    }
    if(page==0xFF)
    page=0;

/*-------------------------------------------------------------------------*/
/*---------------Check condition display page1,page2 or page3-----------*/
   switch(page)                      // Check display page
   {
    case 20 :   showstart();
                page = 0;
                break;
    case 0 :    lcd_command(0x0C);
                min_x=min;
                display_datetime();                          // Call function display p
eal Date and Real Time)
                if(tele==0)
                {
                lcd_clear();
                for(i=0;i<16;i++)            // Loop for show screen
                lcd_text(*(teletext+i));
                delay(500);
                    if(select==1)
                    {
                     ack=1;
                     delay(500);
                     passtele();
                     i=0;
                     while((chktelepass==0)&&(i<2))
                        {
                          i++;
                          passtele();
                          if(i==2)
                          ack=0;
                        }
                     if(chktelepass==1)
                     {
                       while(ack==1)
                       {
                       lcd_clear();
                       for(s=0;s<16;s++)
                       lcd_text(*(wait_dail+s));
                       while(press==0);
                       presstele();
                       lcd_clear();
                       switch(x)
                       {
                         case 1 : sup_tele();

                                  lcd_clear();
                                  lcd_command(0x80);
                                  for(i=0;i<16;i++)          // Loop for show screen
                                  lcd_text(*(loop+i));
                                  lcd_command(0x87);
                                  lcd_command(0x0F);
                                  while(press==0);
                                  presstele();
                                  lcd_command(0x0C);
                                  if(x==10)
                                  x=0;


                                  i=0;
                                     if(ch==1)
                                      {
                                      while(i<x)
                                      {
                                        drive1=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    if(chkdrive1==1)
                    {
                    i++;
                    while(chkdrive1==1);

                    }
                    lcd_command(0xC7);
                    lcd_text(i|0x30);
                  }
                 drive1=0;
                 }

               i=0;
                  if(ch==2)
                   {
                   while(i<x)
                   {
                       drive2=1;
                       if(chkdrive2==1)
                       {
                       i++;
                       while(chkdrive2==1);

                       }
                       lcd_command(0xC7);
                       lcd_text(i|0x30);
                    }
                   drive2=0;
                   }
                 break;

case 2 : sup_tele();

                 lcd_clear();
                 lcd_command(0x80);
                 for(i=0;i<16;i++)                    // Loop for show screen
                 lcd_text(*(stop+i));
                 lcd_command(0x8E);
                 lcd_text(ch|0x30);
                 delay(300);
                 if(ch==1)
                 {
                 for(x=0;(x<=3);x++)
                    {
                        sun_h_1[x]=0xFF;
                        sun_m_1[x]=0xFF;
                        mon_h_1[x]=0xFF;
                        mon_m_1[x]=0xFF;
                        tue_h_1[x]=0xFF;
                        tue_m_1[x]=0xFF;
                        wed_h_1[x]=0xFF;
                        wed_m_1[x]=0xFF;
                        thu_h_1[x]=0xFF;
                        thu_m_1[x]=0xFF;
                        fri_h_1[x]=0xFF;
                        fri_m_1[x]=0xFF;
                        sat_h_1[x]=0xFF;
                        sat_m_1[x]=0xFF;
                    }
                 }

                 if(ch==2)
                 {
                 for(x=0;(x<=3);x++)
                    {
                        sun_h_2[x]=0xFF;
                        sun_m_2[x]=0xFF;
                        mon_h_2[x]=0xFF;
                        mon_m_2[x]=0xFF;
                        tue_h_2[x]=0xFF;
                        tue_m_2[x]=0xFF;
                        wed_h_2[x]=0xFF;
                        wed_m_2[x]=0xFF;
                        thu_h_2[x]=0xFF;
                        thu_m_2[x]=0xFF;
                        fri_h_2[x]=0xFF;
                        fri_m_2[x]=0xFF;
                        sat_h_2[x]=0xFF;
                        sat_m_2[x]=0xFF;
                    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                    }
                                    break;

                            case 3 :  lcd_command(0x80);
                                      lcd_text(x|0x30);
                                      delay(100);
                                      // sound
                                      lcd_clear();
                                      lcd_command(0x80);
                                      for(i=0;i<16;i++)              // Loop for show screen
                                      lcd_text(*(set+i));
                                      num=3;
                                      lcd_command(0xC1);
                                      lcd_command(0x0F);

                                      while(num<6)
                                        {
                                        while(press==0);
                                        presstele();
                                        pass_buf[num]=x;
                                        num++;
                                        }
                                       lcd_command(0x0C);
                                        delay(50);

                                      break;

                            case 4 :  lcd_command(0x80);
                                      lcd_text(x|0x30);
                                      delay(100);
                                      lcd_clear();
                                      lcd_command(0x80);
                                      for(i=0;i<16;i++)              // Loop for show screen
                                      lcd_text(*(stop_tele+i));
                                      delay(200);
                                      ack=0;

                                      break;


                                  }
                                }
                              }
                                }
                              lcd_clear();
                              }
                              if(min_x!=min)
                              check();
                              break;
                    case 1 :  ad_pass();
                              if(i==1)
                              {
                                  select=1;
                                 display_setdate();
                                 while(x_key < 6)
                                 {
                                    push_key = scankey();
                                   if((push_key != 10)&&(push_key!=12)&&(push_key != 0xFF))

                                    {
                                      x_key++;                       // increase counter
                                      date_buf[x_key] = push_key;
                                      if(x_key==2)                   // IF push key 2 tim
t curcer 1 time
                                          lcd_command(0xC6);
                                      if(x_key==4)                           // IF push ke
me shift curcer 1 time
                                          lcd_command(0xC9);
                                      if(date_buf[1]==3)
                                      {
                                          if(date_buf[2]>1)
                                            protect(1,0xC4);
                                       }
                                      if(date_buf[1]>3)
                                       protect(0,0xC3);
                                      if((date_buf[1]==0)&&(date_buf[2]==0)&&(x_key==2))
                                        protect(1,0xC4);
                                      switch(date_buf[3])
                                      {
                                          case 0 : if((date_buf[4]==0)&&(x_key==4))
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
  case 2 : if((date_buf[1]==3)&&(x_key==
           { protect(0,0xC7);
             delay(200);
             display_setdate();
           }
           if((date_buf[1]==2)&&(date_bu
           {
             a=((date_buf[5]<<4)+(date_
               if(a!=0)
               { protect(0,0xCA);
                 delay(200);
                 display_setdate();
               }
           }
           break;
  case 4 : support();
           break;
  case 6 : support();
           break;
  case 9 : support();
           break;
 }
 break;
        case 1 : if(date_buf[4]>2)
                  protect(3,0xC7);
                 if((date_buf[4]==1)&&(x_key==4))
                 support();
                 break;
        default :  x_key=2;
                   lcd_command(0xC6);
                   lcd_text('!');
                   lcd_command(0xC6);
                   break;
      }
    }
   if(push_key==12)
     x_key=7;
   if(push_key == 10)
     {
       x_key=0;
       lcd_command(0xC3);
     }
  }
 if(push_key!=12)
 {
  lcd_clear();
  lcd_command(0x80);
  for(i=0;i<16;i++)
  lcd_text(*(setday+i));
  day=0xFF;
  lcd_command(0xC7);
  while(day==0xFF)
  {
  day=scankey();
  if(((day>7)||(day==0))&&(day!=0xFF))
    {
      day=0xFF;
      lcd_command(0xC7);
      lcd_text('!');
      lcd_command(0xC7);
    }
  }
  //delay(100);
  DS1307_wrdate();                          // Write Date
to DS1307

  }
 }
 x_key = 0;
 page = 0;
 lcd_clear();
 push_key = 0xFF;
 break;
case 2 :   ad_pass();
           if(i==1)
           {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    while(x_key < 6)                                    // Check for end inser
g Date value 6 time(hh/mm/ss)
                    {
                        push_key = scankey();                           // Scankey and Keep to pus

                        if((push_key != 10)&&(push_key!=12)&&(push_key != 0xFF))
                        {
                            x_key++;                                    // increase counte
e
                            time_buf[x_key] = push_key;


                            if(x_key==2)                                // IF push key 2 time
urcer 1 time
                                lcd_command(0xC6);
                            if(x_key==4)                                // IF push key 4 time
urcer 1 time
                                lcd_command(0xC9);
                            if(time_buf[1]>2)
                                protect(0,0xC3);
                             if((time_buf[1]==2)&&(time_buf[2]>=4))
                                protect(1,0xC4);
                            if(time_buf[3]>=6)
                                protect(2,0xC6);
                            if(time_buf[5]>=6)
                                protect(4,0xC9);
                        }
                    if(push_key == 10)
                        {
                            x_key=0;
                            lcd_command(0xC3);
                        }
                    if(push_key==12)
                        x_key=6;
                    }
                    if(push_key!=12)
                    {
                    delay(100);
                    DS1307_wrtime();                    // Write Time data to DS1307
                    }
                    }
                    x_key = 0;                          // Reload counter
                    page = 0;
                    push_key = 0xFF;
                    lcd_clear();
                    break;

                //*********************************************************************

    case 3 :        if(select==0)
                    {
                    i=pass();
                    if(i==1)
                        {
                        select=1;
                        setpass();
                        }
                        page=0;
                        lcd_clear();
                    }
                    else
                    {i=changepass();
                     x=0;
                     while((i==1)&&(x<2))
                     {  x++;
                        i=changepass();
                        if(x==2)
                            select=0;
                      }
                        page=0;
                        lcd_clear();
                     }
                     break;

                /*-------------------------------------------------------------------*/
        case 4 :    ad_pass();
                    if(i==1)
                    {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
for(i=0;i<16;i++)                    // Loop for show screen
lcd_text(*(feed+i));
delay(500);
lcd_clear();
lcd_command(0x80);
for(i=0;i<16;i++)                    // Loop for show screen
lcd_text(*(channel+i));
lcd_command(0x0F);
lcd_command(0xC7);
i=0xFF;
while(i==0xFF)
     {
     i=scankey();
     if(((i>2)||(i==0))&&(i!=0xFF)&&(i!=12))
       {
         i=0xFF;
         lcd_command(0xC7);
         lcd_text('!');
         lcd_command(0xC7);
       }
      if((i<3)&&(i!=0))
      ch=i;
     }
if(i!=12)
{
delay(100);
lcd_command(0x0C);
lcd_command(0x80);
for(i=0;i<16;i++)                    // Loop for show screen
lcd_text(*(setday+i));
lcd_command(0xC0);
for(i=0;i<16;i++)                    // Loop for show screen
lcd_text(*(dayfeed+i));
lcd_command(0x0F);
i=0xFF;
while(((i==0xFF)||(i>7)||(i==0))&&(i!=12)&&(i!=10))
{
     lcd_command(0xC6);
     i=scankey();
        if(((i>7)&&(i!=10)&&(i!=12)&&(i!=0xFF))||(i==0))
        {
           lcd_command(0xC6);
           lcd_text('!');
           lcd_command(0xC6);
        }
        if((i<=7)&&(i!=0))
        dayf=i;

}
  lcd_clear();
  lcd_command(0x0C);
  lcd_command(0x80);
 switch(i)
 {   case 1 :     set_sun();
                  break;
     case 2 :     set_mon();
                  break;
     case 3 :     set_tue();
                  break;
     case 4 :     set_wed();
                  break;
     case 5 :     set_thu();
                  break;
     case 6 :     set_fri();
                  break;
     case 7 :    set_sat();
                  break;
     case 10 :    set_sun();
                 set_mon();
                 set_tue();
                 set_wed();
                 set_thu();
                 set_fri();
                 set_sat();
                 break;
  }
          }
          page = 0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        lcd_clear();
                        x_key=0;

                        break;
        //*****************************************************************

        }
    }
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*--------------------------------------------------------------------------
REG52.H

Header file for generic 80C52 and 80C32 microcontroller.
Copyright (c) 1988-2002 Keil Elektronik GmbH and Keil Software, Inc.
All rights reserved.
--------------------------------------------------------------------------*/

#ifndef __REG52_H__
#define __REG52_H__

/*  BYTE Registers  */
sfr P0    = 0x80;
sfr P1    = 0x90;
sfr P2    = 0xA0;
sfr P3    = 0xB0;
sfr PSW   = 0xD0;
sfr ACC   = 0xE0;
sfr B     = 0xF0;
sfr SP    = 0x81;
sfr DPL   = 0x82;
sfr DPH   = 0x83;
sfr PCON  = 0x87;
sfr TCON  = 0x88;
sfr TMOD  = 0x89;
sfr TL0   = 0x8A;
sfr TL1   = 0x8B;
sfr TH0   = 0x8C;
sfr TH1   = 0x8D;
sfr IE    = 0xA8;
sfr IP    = 0xB8;
sfr SCON  = 0x98;
sfr SBUF  = 0x99;

/*  8052 Extensions  */
sfr T2CON  = 0xC8;
sfr RCAP2L = 0xCA;
sfr RCAP2H = 0xCB;
sfr TL2    = 0xCC;
sfr TH2    = 0xCD;


/*  BIT Registers  */
/*  PSW  */
sbit CY    = PSW^7;
sbit AC    = PSW^6;
sbit F0    = PSW^5;
sbit RS1   = PSW^4;
sbit RS0   = PSW^3;
sbit OV    = PSW^2;
sbit P     = PSW^0; //8052 only

/*  TCON  */
sbit TF1   = TCON^7;
sbit TR1   = TCON^6;
sbit TF0   = TCON^5;
sbit TR0   = TCON^4;
sbit IE1   = TCON^3;
sbit IT1   = TCON^2;
sbit IE0   = TCON^1;
sbit IT0   = TCON^0;

/*  IE  */
sbit EA    = IE^7;
sbit ET2   = IE^5; //8052 only
sbit ES    = IE^4;
sbit ET1   = IE^3;
sbit EX1   = IE^2;
sbit ET0   = IE^1;
sbit EX0   = IE^0;

/*  IP  */
sbit PT2   = IP^5;
sbit PS    = IP^4;
sbit PT1   = IP^3;
sbit PX1   = IP^2;
sbit PT0   = IP^1;
sbit PX0   = IP^0;

/*  P3  */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
sbit RD    = P3^7;
sbit WR    = P3^6;
sbit T1    = P3^5;
sbit T0    = P3^4;
sbit INT1  = P3^3;
sbit INT0  = P3^2;
sbit TXD   = P3^1;
sbit RXD   = P3^0;

/*  SCON  */
sbit SM0   = SCON^7;
sbit SM1   = SCON^6;
sbit SM2   = SCON^5;
sbit REN   = SCON^4;
sbit TB8   = SCON^3;
sbit RB8   = SCON^2;
sbit TI    = SCON^1;
sbit RI    = SCON^0;

/*  P1  */
sbit T2EX  = P1^1; // 8052 only
sbit T2    = P1^0; // 8052 only

/*  T2CON  */
sbit TF2    = T2CON^7;
sbit EXF2   = T2CON^6;
sbit RCLK   = T2CON^5;
sbit TCLK   = T2CON^4;
sbit EXEN2  = T2CON^3;
sbit TR2    = T2CON^2;
sbit C_T2   = T2CON^1;
sbit CP_RL2 = T2CON^0;

#endif
```

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#include<intrins.h>
sbit SDA = P1^0;
sbit SCL = P1^1;
void i2c_delay(void)
{
    unsigned char i;
    for(i=0;i<20;i++)
    _nop_();
}
void i2c_clk(void)
{
    i2c_delay();
    SCL = 1;
    i2c_delay();
    SCL = 0;
}
void i2c_start(void)
{
    if(SCL)
    SCL = 0;
    SDA = 1;
    SCL = 1;
    i2c_delay();
    SDA = 0;
    i2c_delay();
    SCL = 0;
}
void i2c_stop(void)
{
    if(SCL)
    SCL = 0;
    SDA = 0;
    i2c_delay();
    SCL = 1;
    i2c_delay();
    SDA = 1;
}
bit i2c_wrdata(unsigned char dat)
{
    bit data_bit;
    unsigned char i;
    for(i=0;i<8;i++)
    {
        data_bit = dat & 0x80;
        SDA = data_bit;
        i2c_clk();
        dat = dat<<1;
    }
    SDA = 1;
    i2c_delay();
    SCL = 1;
    i2c_delay();
    data_bit = SDA;
    SCL = 0;
    i2c_delay();
    return(data_bit);
}
unsigned char i2c_rddata(void)
{
    bit rd_bit;
    unsigned char i,dat;
    dat = 0x00;
    for(i=0;i<8;i++)
    {
        i2c_delay();
        SCL = 1;
        i2c_delay();
        rd_bit = SDA;
        dat = dat<<1;
        dat = dat | rd_bit;
        SCL = 0;
    }
    SDA = 1;
    i2c_delay();
    i2c_clk();
    SCL = 1;
    return(dat);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*---------------------------------------------------------------------------
INTRINS.H

Intrinsic functions for C51.
Copyright (c) 1988-2002 Keil Elektronik GmbH and Keil Software, Inc.
All rights reserved.
---------------------------------------------------------------------------*/

#ifndef __INTRINS_H__
#define __INTRINS_H__

extern void          _nop_      (void);
extern bit           _testbit_  (bit);
extern unsigned char _cror_     (unsigned char, unsigned char);
extern unsigned int  _iror_     (unsigned int,  unsigned char);
extern unsigned long _lror_     (unsigned long, unsigned char);
extern unsigned char _crol_     (unsigned char, unsigned char);
extern unsigned int  _irol_     (unsigned int,  unsigned char);
extern unsigned long _lrol_     (unsigned long, unsigned char);
extern unsigned char _chkfloat_(float);

#endif
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
sbit e = P3^6;
sbit rs = P3^7;
void delay(int tick)
{
    unsigned int i,j;
    for(i=0;i<tick;i++)
    for(j=0;j<250;j++);
}
void lcd_command(unsigned char com) //write command
{
    rs = 0;
    e = 1;
    P0 = com;
    delay(10);
    e = 0;
    delay(10);
}
void lcd_text(unsigned char text)    //write command
{
    rs = 1;
    e = 1;
    P0 = text;
    delay(10);
    e = 0;
    delay(10);
}
void lcd_clear()
{
    lcd_command(0x01);
}
void lcd_init()
{
    delay(250);
    delay(250);
    lcd_command(0x38);      //on display ,8 bit display ,5*7 dot
    lcd_command(0x0C);      //none curser
    lcd_command(0x01);      //clear screen
}
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
sbit c1 = P2^6;                 //colum1
sbit c2 = P2^5;                 //colum2
sbit c3 = P2^4;                 //colum3
sbit r1 = P2^0;                 //row1
sbit r2 = P2^1;                 //row2
sbit r3 = P2^2;                 //row3
sbit r4 = P2^3;                 //row4

unsigned char scankey(void)     // function scankey 4x3
{
  unsigned char ret = 0xFF;
    c1 = 0;                             // scan colum1
    if(r1==0)
    {
    lcd_text(0x31);
        while(r1==0);           //check push key 1
            ret = 0x01;             // send index = 1
    }
    if(r2==0)                           //check push key 4
    {
    lcd_text(0x34);
        while(r2==0);
        ret = 0x04;
    }                           // send index = 4
    if(r3==0)
    {
    lcd_text(0x37);
        while(r3==0);
        ret = 0x07;             // send index = 7
    }

        if(r4==0)           // check push key 0
    {
        while(r4==0);
        ret = 0x0A;
    }                           // send index = 0

    c1 = 1;                     // stop check colum1

    c2 = 0;                             // scan colum2
    if(r1==0)
    {
    lcd_text(0x32);
        while(r1==0);
        ret = 0x02;
    }                           // send index = 2
    if(r2==0)                           //check push key 7
    {
    lcd_text(0x35);
        while(r2==0);
        ret = 0x05;
    }                           // send index = 5
    if(r3==0)                           //check push key 8
    {
    lcd_text(0x38);
        while(r3==0);
        ret = 0x08;
    }                           // send index = 8
    if(r4==0)                           // check push key 0
    {
    lcd_text(0x30);
        while(r4==0);
        ret = 0x00;
    }                           // send index = 0
    c2 = 1;                             // stop check colum2

    c3 = 0;                             // scan colum3
    if(r1==0)
    {
    lcd_text(0x33);
        while(r1==0);
        ret = 0x03;
    }                           // send index = 3
    if(r2==0)                           //check push key 6
    {
    lcd_text(0x36);
        while(r2==0);
        ret = 0x06;
    }                           // send index = 6
    if(r3==0)                           //check push key 9
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

: 1

```
    {
    lcd_text(0x39);
        while(r3==0);
        ret = 0x09;
    }                                   // send index = 9
    if(r4==0)                           //check push key 9
    {
        while(r4==0);
        ret = 0x0C;
    }                                   // send index = 9
    c3 = 1;                             // stop check colum3
    return(ret);
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
unsigned char scanx(void)      // function scankey 4x3
{
  unsigned char retx = 0xFF;
    c1 = 0;                          // scan colum1
    if(r1==0)
    {
    lcd_text('*');
        while(r1==0);              //check push key 1
            retx = 0x01;                // send index = 1
    }
    if(r2==0)                           //check push key 4
    {
    lcd_text('*');
        while(r2==0);
        retx = 0x04;
    }                                  // send index = 4
    if(r3==0)
    {
    lcd_text('*');
        while(r3==0);
        retx = 0x07;                    // send index = 7
    }

    if(r4==0)                          // check push key 0
    {
    lcd_text('*');
        while(r4==0);
        retx = 0x0A;
    }                                  // send index = 0

    c1 = 1;                           // stop check colum1

    c2 = 0;                                 // scan colum2
    if(r1==0)
    {
    lcd_text('*');
        while(r1==0);
        retx = 0x02;
    }                                  // send index = 2
    if(r2==0)                               //check push key 7
    {
    lcd_text('*');
        while(r2==0);
        retx = 0x05;
    }                                  // send index = 5
    if(r3==0)                               //check push key 8
    {
    lcd_text('*');
        while(r3==0);
        retx = 0x08;
    }                                  // send index = 8
    if(r4==0)                               // check push key 0
    {
    lcd_text('*');
        while(r4==0);
        retx = 0x00;
    }                                  // send index = 0
    c2 = 1;                                 // stop check colum2

    c3 = 0;                                 // scan colum3
    if(r1==0)
    {
    lcd_text('*');
        while(r1==0);
        retx = 0x03;
    }                                  // send index = 3
    if(r2==0)                               //check push key 6
    {
    lcd_text('*');
        while(r2==0);
        retx = 0x06;
    }                                  // send index = 6
    if(r3==0)                               //check push key 9
    {
    lcd_text('*');
        while(r3==0);
        retx = 0x09;
```

เอกสารนี้จัดทำขึ้นไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }                                       // send index = 9
    if(r4==0)                               //check push key 9
    {
    lcd_text('*');
        while(r4==0);
        retx = 0x0C;
    }                                       // send index = 9
    c3 = 1;                                 // stop check colum3
    return(retx);
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
unsigned char pass_buf[6];
code unsigned char *in= "ADMIN PASSWORD:";
code unsigned char *ok= "YOU ARE WELCOME!";
code unsigned char *no= "INVALID PASSWORD";
code unsigned char *in2="INSERT PASSWORD:";
code unsigned char *set=" SETUP PASSWORD:";
code unsigned char *old=" OLD PASSWORD ?";
unsigned char num=0;
unsigned char s;


unsigned char pass(void)
{       num=0;
        lcd_clear();
        lcd_command(0x80);
        for(s=0;s<15;s++)
        lcd_text(*(in+s));
        lcd_command(0xC1);
        lcd_command(0x0F);
        while(num<6)
        {
            s = scanx();
            if(s!=0xFF)
            {
                pass_buf[num]=s;
                num++;
            }
         }
      if((1==pass_buf[0])&&(2==pass_buf[1])&&(3==pass_buf[2])&&(4==pass_buf[3])&&(5==pass_buf[4]
=pass_buf[5]))
        {
            lcd_clear();
            lcd_command(0x80);
            lcd_command(0x0C);
            for(s=0;s<16;s++)
            lcd_text(*(ok+s));
            delay(500);
            s=1;
        }
        else
        {
            lcd_clear();
            lcd_command(0x80);
            lcd_command(0x0C);
            for(s=0;s<16;s++)
            lcd_text(*(no+s));
            delay(500);
            s=0;
            num=0;
        }
        return(s);
    }

//*********************************************************

 unsigned char pass2(void)
{
        num=0;
        lcd_clear();
        lcd_command(0x80);
        for(s=0;s<16;s++)
        lcd_text(*(in2+s));
        lcd_command(0xC1);
        lcd_command(0x0F);
        while(num<3)
        {
            s = scanx();
            if(s!=0xFF)
            {
                pass_buf[num]=s;
                num++;
            }
         }
      if((pass_buf[3]==pass_buf[0])&&(pass_buf[4]==pass_buf[1])&&(pass_buf[5]==pass_buf[2]))
        {
            lcd_clear();
            lcd_command(0x80);
            lcd_command(0x0C);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        for(s=0;s<16;s++)
        lcd_text(*(ok+s));
        delay(500);
        s=1;

    }
        else
    {
        lcd_clear();
        lcd_command(0x80);
        lcd_command(0x0C);
        for(s=0;s<16;s++)
        lcd_text(*(no+s));
        delay(500);
        s=0;
        num=0;
    }
    return(s);
}

//*****************************************************************************

void setpass(void)
{
   num=3;
   lcd_clear();
   lcd_command(0x80);
   for(s=0;s<16;s++)
    lcd_text(*(set+s));
   lcd_command(0xC1);
   lcd_command(0x0F);

   while(num<6)
        {
        s = scanx();
        if(s!=0xFF)
            {
                pass_buf[num]=s;
                num++;
            }
         }
}

//*****************************************************************************

unsigned char changepass(void)
{
        num=0;
        lcd_clear();
        lcd_command(0x80);
        for(s=0;s<15;s++)
         lcd_text(*(old+s));
        lcd_command(0xC1);
        lcd_command(0x0F);
        while(num<3)
        {
        s = scanx();
        if(s!=0xFF)
            {
                pass_buf[num]=s;
                num++;
            }
         }

    if((pass_buf[3]==pass_buf[0])&&(pass_buf[4]==pass_buf[1])&&(pass_buf[5]==pass_buf[2]))
        {
            lcd_clear();
        lcd_command(0x80);
        lcd_command(0x0C);
        for(s=0;s<16;s++)
        lcd_text(*(ok+s));
        delay(500);
        setpass();
        s=0;
      }
         else
      {
        lcd_clear();
        lcd_command(0x80);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        lcd_command(0x0C);
        for(s=0;s<16;s++)
        lcd_text(*(no+s));
        delay(500);
        num=0;
        s=1;
     }
    return(s);
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DALLAS SEMICONDUCTOR MAXIM

DS1307
64 x 8, Serial, I²C Real-Time Clock

GENERAL DESCRIPTION

The DS1307 serial real-time clock (RTC) is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially through an I²C™, bidirectional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power-sense circuit that detects power failures and automatically switches to the battery supply.

FEATURES

- Real-Time Clock (RTC) Counts Seconds, Minutes, Hours, Date of the Month, Month, Day of the week, and Year with Leap-Year Compensation Valid Up to 2100
- 56-Byte, Battery-Backed, Nonvolatile (NV) RAM for Data Storage
- I²C Serial Interface
- Programmable Square-Wave Output Signal
- Automatic Power-Fail Detect and Switch Circuitry
- Consumes Less than 500nA in Battery-Backup Mode with Oscillator Running
- Optional Industrial Temperature Range: -40°C to +85°C
- Available in 8-Pin DIP or SO
- Underwriters Laboratory (UL) Recognized

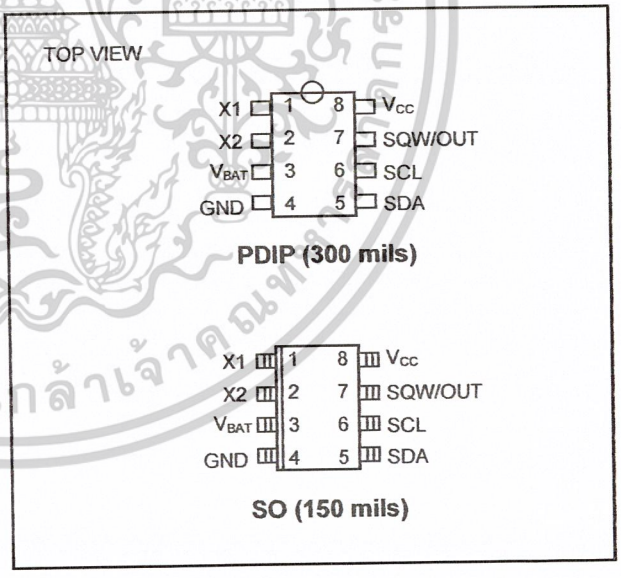
ORDERING INFORMATION

PART	TEMP RANGE	PIN-PACKAGE	TOP MARK
DS1307	0°C to +70°C	8 PDIP	DS1307
DS1307Z	0°C to +70°C	8 SO	DS1307
DS1307N	-40°C to +85°C	8 PDIP	DS1307*
DS1307ZN	-40°C to +85°C	8 SO	DS1307N

** An 'N' is added to the lower right-hand corner of the top brand.*

I²C is a trademark of Philips Corp. Purchase of I²C components of Maxim Integrated Products, Inc., or one of its sublicensed Associated Companies, conveys a license under the Philips I²C Patent Rights to use these components in an I²C system, provided that the system conforms to the I²C Standard Specification as defined by Philips Corp.

PIN CONFIGUATIONS



Typical Operating Circuit appears at end of data sheet.

Note: *Some revisions of this device may incorporate deviations from published specifications known as errata. Multiple revisions of any device may be simultaneously available through various sales channels. For information about device errata, click here: www.maxim-ic.com/errata.*

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ABSOLUTE MAXIMUM RATINGS

Voltage Range on Any Pin Relative to Ground ....-0.5V to +7.0V
Operating Temperature Range (noncondensing) 0°C to +70°C (Commercial), -40°C to +85°C (Industrial)
Storage Temperature Range ... .. -55°C to +125°C
Soldering Temperature (DIP, leads) .....+260°C for 10 seconds
Soldering Temperature (surface mount) .See JPC/JEDEC Standard J-STD-020A

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to the absolute maximum rating conditions for extended periods may affect device reliability.

RECOMMENDED DC OPERATING CONDITIONS

(T_A = 0°C to +70°C, T_A = -40°C to +85°C.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Supply Voltage	V_{CC}		4.5	5.0	5.5	V
Logic 1 Input	V_{IH}		2.2		V_{CC} + 0.3	V
Logic 0 Input	V_{IL}		-0.3		+0.8	V
V_{BAT} Battery Voltage	V_{BAT}		2.0	3	3.5	V

DC ELECTRICAL CHARACTERISTICS

(V_{CC} = 4.5V to 5.5V; T_A = 0°C to +70°C, T_A = -40°C to +85°C.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Input Leakage (SCL)	I_{LI}				1	µA
I/O Leakage (SDA, SQW/OUT)	I_{LO}				1	µA
Logic 0 Output (I_{OL} = 5mA)	V_{OL}				0.4	V
Active Supply Current (f_{SCL} = 100kHz)	I_{CCA}				1.5	mA
Standby Current	I_{CCS}	(Note 3)			200	µA
V_{BAT} Leakage Current	I_{BATLKG}			5	50	nA
Power-Fail Voltage (V_{BAT} = 3.0V)	V_{PF}		1.216 x V_{BAT}	1.25 x V_{BAT}	1.284 x V_{BAT}	V

DC ELECTRICAL CHARACTERISTICS

(V_{CC} = 0V, V_{BAT} = 3.0V; T_A = 0°C to +70°C, T_A = -40°C to +85°C.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
V_{BAT} Current (OSC ON); SQW/OUT OFF	I_{BAT1}			300	500	nA
V_{BAT} Current (OSC ON); SQW/OUT ON (32kHz)	I_{BAT2}			480	800	nA
V_{BAT} Data-Retention Current (Oscillator Off)	I_{BATDR}			10	100	nA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

AC ELECTRICAL CHARACTERISTICS

(V_{CC} = 4.5V to 5.5V; T_A = 0°C to +70°C, T_A = -40°C to +85°C.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
SCL Clock Frequency	f_{SCL}		0		100	kHz
Bus Free Time Between a STOP and START Condition	t_{BUF}		4.7			µs
Hold Time (Repeated) START Condition	$t_{HD:STA}$	(Note 4)	4.0			µs
LOW Period of SCL Clock	t_{LOW}		4.7			µs
HIGH Period of SCL Clock	t_{HIGH}		4.0			µs
Setup Time for a Repeated START Condition	$t_{SU:STA}$		4.7			µs
Data Hold Time	$t_{HD:DAT}$		0			µs
Data Setup Time	$t_{SU:DAT}$	(Notes 5, 6)	250			ns
Rise Time of Both SDA and SCL Signals	t_R				1000	ns
Fall Time of Both SDA and SCL Signals	t_F				300	ns
Setup Time for STOP Condition	$t_{SU:STO}$		4.7			µs

CAPACITANCE

(T_A = +25°C)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Pin Capacitance (SDA, SCL)	$C_{I/O}$				10	pF
Capacitance Load for Each Bus Line	C_B	(Note 7)			400	pF

Note 1: All voltages are referenced to ground.

Note 2: Limits at -40°C are guaranteed by design and are not production tested.

Note 3: I_{CCS} specified with V_{CC} = 5.0V and SDA, SCL = 5.0V.

Note 4: After this period, the first clock pulse is generated.

Note 5: A device must internally provide a hold time of at least 300ns for the SDA signal (referred to the $V_{IH(MIN)}$ of the SCL signal) to bridge the undefined region of the falling edge of SCL.

Note 6: The maximum $t_{HD:DAT}$ only has to be met if the device does not stretch the LOW period (t_{LOW}) of the SCL signal.

Note 7: C_B—total capacitance of one bus line in pF.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TIMING DIAGRAM

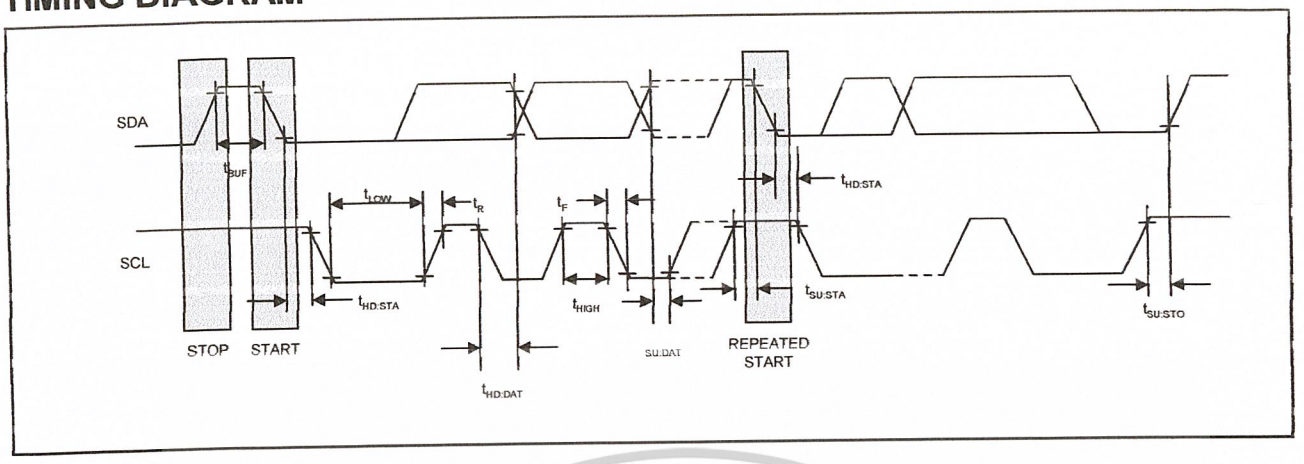
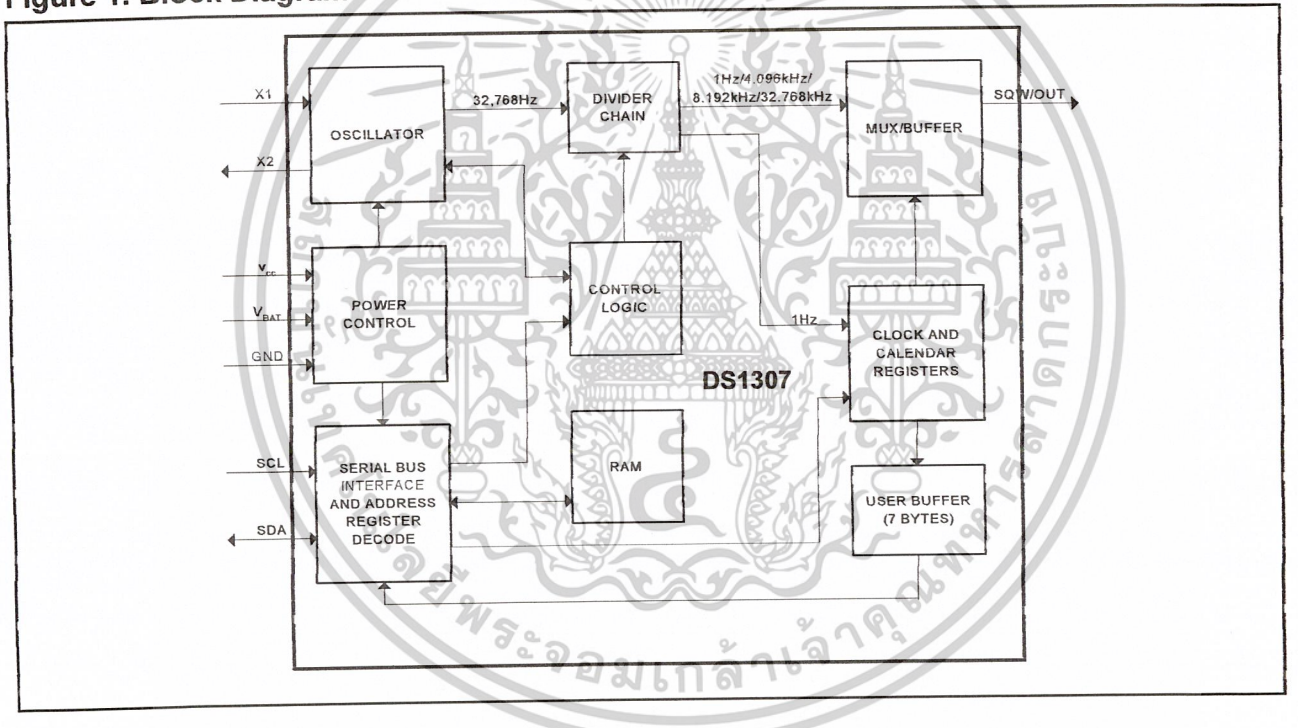


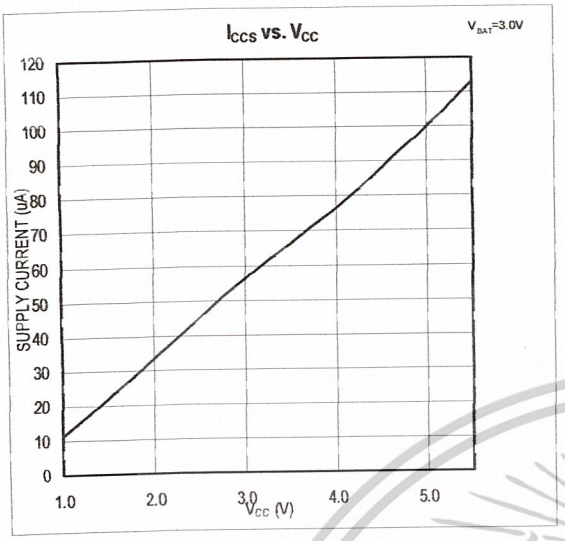
Figure 1. Block Diagram

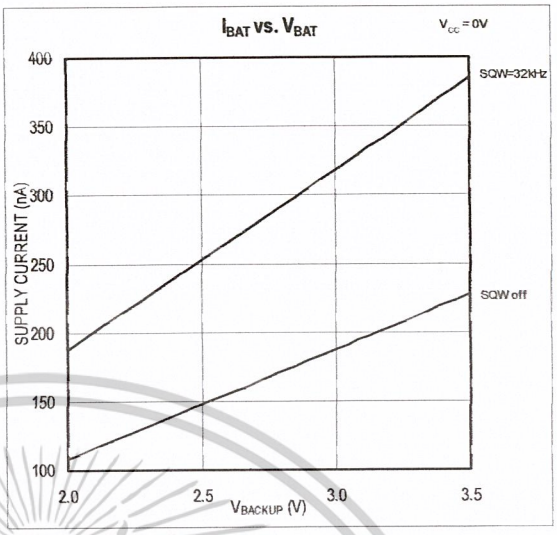


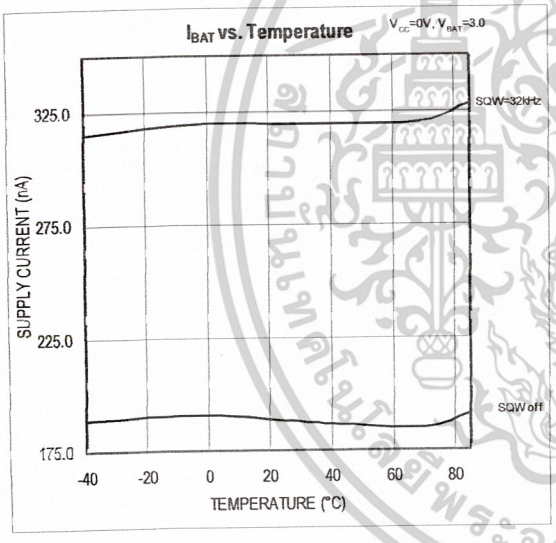
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

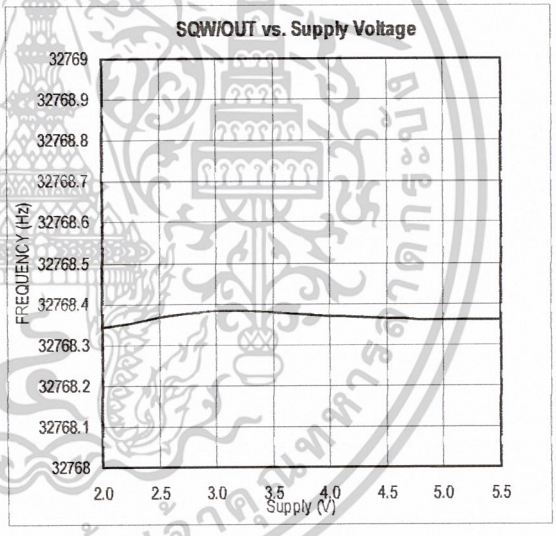
TYPICAL OPERATING CHARACTERISTICS

(V_{CC} = 5.0V, T_A = +25°C, unless otherwise noted.)









เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PIN DESCRIPTION

PIN	NAME	FUNCTION
1	X1	Connections for Standard 32.768kHz Quartz Crystal. The internal oscillator circuitry is designed for operation with a crystal having a specified load capacitance (C_L) of 12.5pF. X1 is the input to the oscillator and can optionally be connected to an external 32.768kHz oscillator. The output of the internal oscillator, X2, is floated if an external oscillator is connected to X1. **Note:** For more information on crystal selection and crystal layout considerations, refer to *Application Note 58: Crystal Considerations with Dallas Real-Time Clocks*.
2	X2	
3	V_{BAT}	Backup Supply Input for Any Standard 3V Lithium Cell or Other Energy Source. Battery voltage must be held between the minimum and maximum limits for proper operation. Diodes in series between the battery and the V_{BAT} pin may prevent proper operation. If a backup supply is not required, V_{BAT} may be grounded. The nominal power-fail trip point (V_{PF}) voltage at which access to the RTC and user RAM is denied is set by the internal circuitry as 1.25 x V_{BAT} nominal. A lithium battery with 48mAhr or greater will back up the DS1307 for more than 10 years in the absence of power at +25°C. UL recognized to ensure against reverse charging current when used with a lithium battery.
4	GND	Ground.
5	SDA	Serial Data Input/Output. SDA is the data input/output for the I^2C serial interface. The SDA pin is open drain and requires an external pullup resistor.
6	SCL	Serial Clock Input. SCL is the clock input for the I^2C interface and is used to synchronize data movement on the serial interface.
7	SWQ/OUT	Square Wave/Output Driver. When enabled, the SQWE bit set to 1, the SQW/OUT pin outputs one of four square-wave frequencies (1Hz, 4kHz, 8kHz, 32kHz). The SQW/OUT pin is open drain and requires an external pullup resistor. SQW/OUT operates with either V_{CC} or V_{BAT} applied.
8	V_{CC}	Primary Power Supply. When voltage is applied within normal limits, the device is fully accessible and data can be written and read. When a backup supply is connected to the device and V_{CC} is below V_{TP}, read and writes are inhibited. However, the timekeeping function continues unaffected by the lower input voltage.

DETAILED DESCRIPTION

The DS1307 is a low-power clock/calendar with 56 bytes of battery-backed SRAM. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The DS1307 operates as a slave device on the I^2C bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed. When V_{CC} falls below 1.25 x V_{BAT}, the device terminates an access in progress and resets the device address counter. Inputs to the device will not be recognized at this time to prevent erroneous data from being written to the device from an out-of-tolerance system. When V_{CC} falls below V_{BAT}, the device switches into a low-current battery-backup mode. Upon power-up, the device switches from battery to V_{CC} when V_{CC} is greater than V_{BAT} +0.2V and recognizes inputs when V_{CC} is greater than 1.25 x V_{BAT}. The block diagram in Figure 1 shows the main elements of the serial RTC.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

OSCILLATOR CIRCUIT

The DS1307 uses an external 32.768kHz crystal. The oscillator circuit does not require any external resistors or capacitors to operate. Table 1 specifies several crystal parameters for the external crystal. Figure 3 shows a functional schematic of the oscillator circuit. If using a crystal with the specified characteristics, the startup time is usually less than one second.

CLOCK ACCURACY

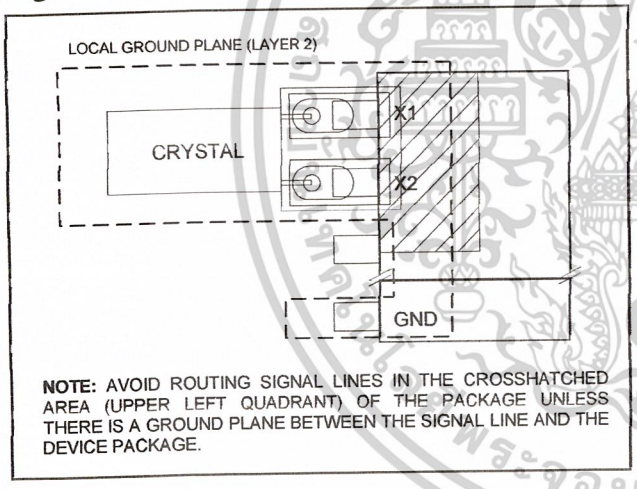
The accuracy of the clock is dependent upon the accuracy of the crystal and the accuracy of the match between the capacitive load of the oscillator circuit and the capacitive load for which the crystal was trimmed. Additional error will be added by crystal frequency drift caused by temperature shifts. External circuit noise coupled into the oscillator circuit may result in the clock running fast. Refer to *Application Note 58: Crystal Considerations with Dallas Real-Time Clocks* for detailed information.

Table 1. Crystal Specifications*

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS
Nominal Frequency	f_O		32.768		kHz
Series Resistance	ESR			45	kΩ
Load Capacitance	C_L		12.5		pF

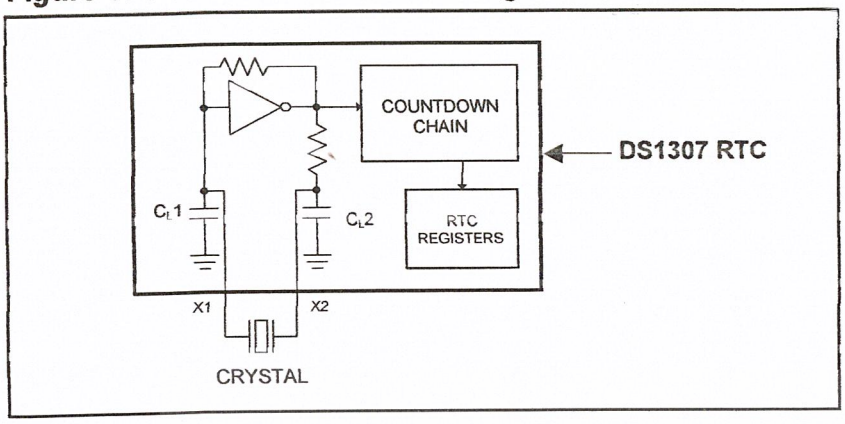
**The crystal, traces, and crystal input pins should be isolated from RF generating signals. Refer to Application Note 58: Crystal Considerations for Dallas Real-Time Clocks for additional specifications.*

Figure 2. Recommended Layout for Crystal



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 3. Oscillator Circuit Showing Internal Bias Network



RTC AND RAM ADDRESS MAP

Table 2 shows the address map for the DS1307 RTC and RAM registers. The RTC registers are located in address locations 00h to 07h. The RAM registers are located in address locations 08h to 3Fh. During a multibyte access, when the address pointer reaches 3Fh, the end of RAM space, it wraps around to location 00h, the beginning of the clock space.

CLOCK AND CALENDAR

The time and calendar information is obtained by reading the appropriate register bytes. Table 2 shows the RTC registers. The time and calendar are set or initialized by writing the appropriate register bytes. The contents of the time and calendar registers are in the BCD format. The day-of-week register increments at midnight. Values that correspond to the day of week are user-defined but must be sequential (i.e., if 1 equals Sunday, then 2 equals Monday, and so on.) Illogical time and date entries result in undefined operation. Bit 7 of Register 0 is the clock halt (CH) bit. When this bit is set to 1, the oscillator is disabled. When cleared to 0, the oscillator is enabled.

Please note that the initial power-on state of all registers is not defined. Therefore, it is important to enable the oscillator (CH bit = 0) during initial configuration.

The DS1307 can be run in either 12-hour or 24-hour mode. Bit 6 of the hours register is defined as the 12-hour or 24-hour mode-select bit. When high, the 12-hour mode is selected. In the 12-hour mode, bit 5 is the AM/PM bit with logic high being PM. In the 24-hour mode, bit 5 is the second 10-hour bit (20 to 23 hours). The hours value must be re-entered whenever the 12/24-hour mode bit is changed.

When reading or writing the time and date registers, secondary (user) buffers are used to prevent errors when the internal registers update. When reading the time and date registers, the user buffers are synchronized to the internal registers on any I²C START. The time information is read from these secondary registers while the clock continues to run. This eliminates the need to re-read the registers in case the internal registers update during a read. The divider chain is reset whenever the seconds register is written. Write transfers occur on the I²C acknowledge from the DS1307. Once the divider chain is reset, to avoid rollover issues, the remaining time and date registers must be written within one second.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Table 2. Timekeeper Registers

ADDRESS	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	FUNCTION	RANGE
00H	CH	10 Seconds			Seconds				Seconds	00 59
01H	0	10 Minutes			Minutes				Minutes	00 59
02H	0	12	10 Hour	10 Hour	Hours				Hours	1 12 +AM/PM 00 23
		24	PM/AM							
03H	0	0	0	0	0	DAY			Day	01 07
04H	0	0	10 Date		Date				Date	01 31
05H	0	0	0	10 Month	Month				Month	01 12
06H	10 Year				Year				Year	00 99
07H	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08H-3FH									RAM 56 x 8	00H–FFH

0 = Always reads back as 0.

CONTROL REGISTER

The DS1307 control register is used to control the operation of the SQW/OUT pin.

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
OUT	0	0	SQWE	0	0	RS1	RS0

Bit 7: Output Control (OUT). This bit controls the output level of the SQW/OUT pin when the square-wave output is disabled. If SQWE = 0, the logic level on the SQW/OUT pin is 1 if OUT = 1 and is 0 if OUT = 0.

Bit 4: Square-Wave Enable (SQWE). This bit, when set to logic 1, enables the oscillator output. The frequency of the square-wave output depends upon the value of the RS0 and RS1 bits. With the square-wave output set to 1Hz, the clock registers update on the falling edge of the square wave.

Bits 1, 0: Rate Select (RS1, RS0). These bits control the frequency of the square-wave output when the square-wave output has been enabled. The following table lists the square-wave frequencies that can be selected with the RS bits.

RS1	RS0	SQUARE-WAVE OUTPUT FREQUENCY
0	0	1Hz
0	1	4.096kHz
1	0	8.192kHz
1	1	32.768kHz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

I²C DATA BUS

The DS1307 supports the I²C protocol. A device that sends data onto the bus is defined as a transmitter and a device receiving data as a receiver. The device that controls the message is called a master. The devices that are controlled by the master are referred to as slaves. The bus must be controlled by a master device that generates the serial clock (SCL), controls the bus access, and generates the START and STOP conditions. The DS1307 operates as a slave on the I²C bus.

Figures 4, 5, and 6 detail how data is transferred on the I²C bus.

- Data transfer may be initiated only when the bus is not busy.
- During data transfer, the data line must remain stable whenever the clock line is HIGH. Changes in the data line while the clock line is high will be interpreted as control signals.

Accordingly, the following bus conditions have been defined:

Bus not busy: Both data and clock lines remain HIGH.

Start data transfer: A change in the state of the data line, from HIGH to LOW, while the clock is HIGH, defines a START condition.

Stop data transfer: A change in the state of the data line, from LOW to HIGH, while the clock line is HIGH, defines the STOP condition.

Data valid: The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the HIGH period of the clock signal. The data on the line must be changed during the LOW period of the clock signal. There is one clock pulse per bit of data.

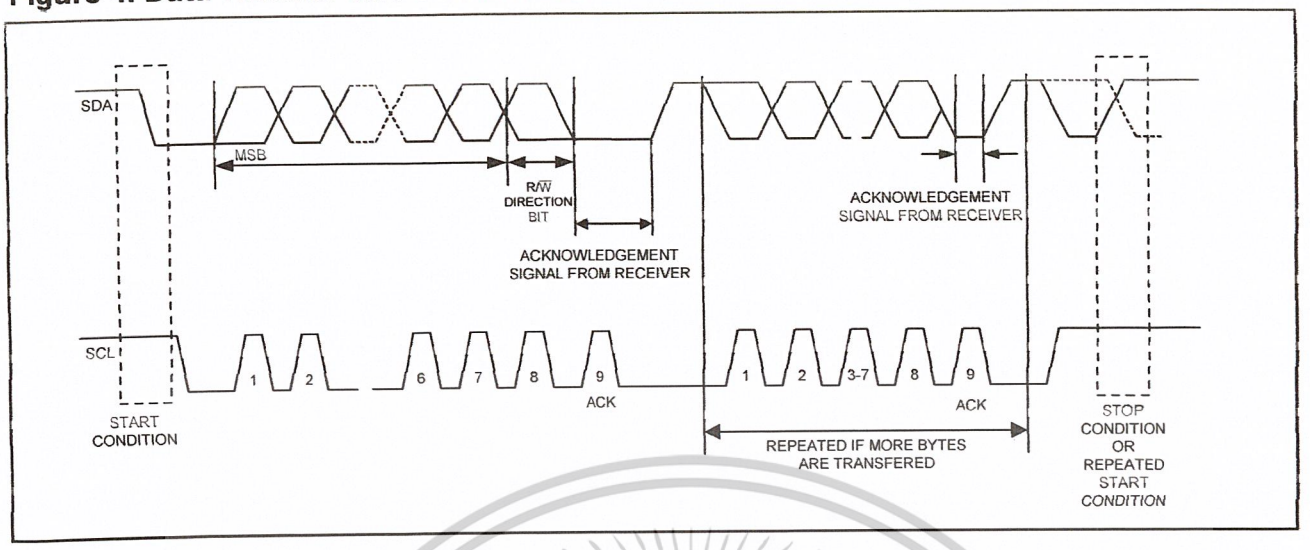
Each data transfer is initiated with a START condition and terminated with a STOP condition. The number of data bytes transferred between START and STOP conditions is not limited, and is determined by the master device. The information is transferred byte-wise and each receiver acknowledges with a ninth bit. Within the 2-wire bus specifications a standard mode (100kHz clock rate) and a fast mode (400kHz clock rate) are defined. The DS1307 operates in the standard mode (100kHz) only.

Acknowledge: Each receiving device, when addressed, is obliged to generate an acknowledge after the reception of each byte. The master device must generate an extra clock pulse which is associated with this acknowledge bit.

A device that acknowledges must pull down the SDA line during the acknowledge clock pulse in such a way that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse. Of course, setup and hold times must be taken into account. A master must signal an end of data to the slave by not generating an acknowledge bit on the last byte that has been clocked out of the slave. In this case, the slave must leave the data line HIGH to enable the master to generate the STOP condition.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 4. Data Transfer on I²C Serial Bus



Depending upon the state of the R/$\overline{W}$ bit, two types of data transfer are possible:

1. **Data transfer from a master transmitter to a slave receiver.** The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte. Data is transferred with the most significant bit (MSB) first.

2. **Data transfer from a slave transmitter to a master receiver.** The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. This is followed by the slave transmitting a number of data bytes. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a "not acknowledge" is returned.

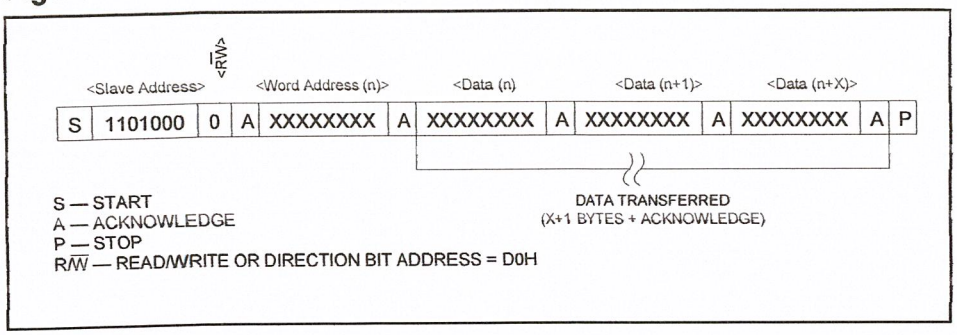
   The master device generates all the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since a repeated START condition is also the beginning of the next serial transfer, the bus will not be released. Data is transferred with the most significant bit (MSB) first.

The DS1307 may operate in the following two modes:

1. **Slave Receiver Mode (Write Mode):** Serial data and clock are received through SDA and SCL. After each byte is received an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Hardware performs address recognition after reception of the slave address and direction bit (see Figure 5). The slave address byte is the first byte received after the master generates the START condition. The slave address byte contains the 7-bit DS1307 address, which is 1101000, followed by the direction bit (R/$\overline{W}$), which for a write is 0. After receiving and decoding the slave address byte, the DS1307 outputs an acknowledge on SDA. After the DS1307 acknowledges the slave address + write bit, the master transmits a word address to the DS1307. This sets the register pointer on the DS1307, with the DS1307 acknowledging the transfer. The master can then transmit zero or more bytes of data with the DS1307 acknowledging each byte received. The register pointer automatically increments after each data byte are written. The master will generate a STOP condition to terminate the data write.

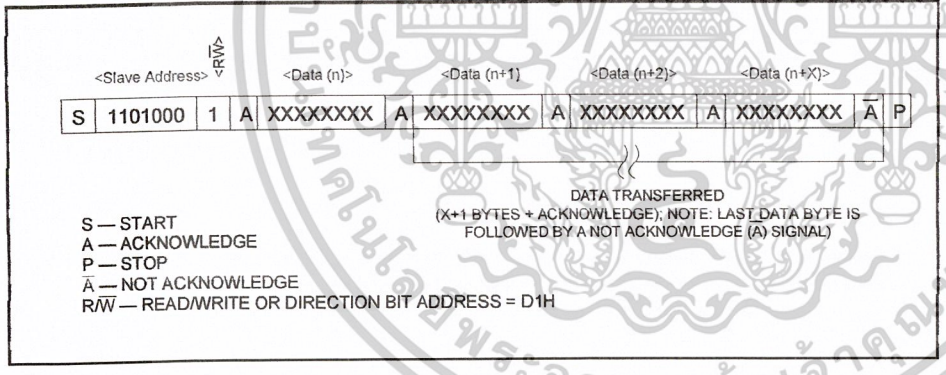
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 5. Data Write—Slave Receiver Mode



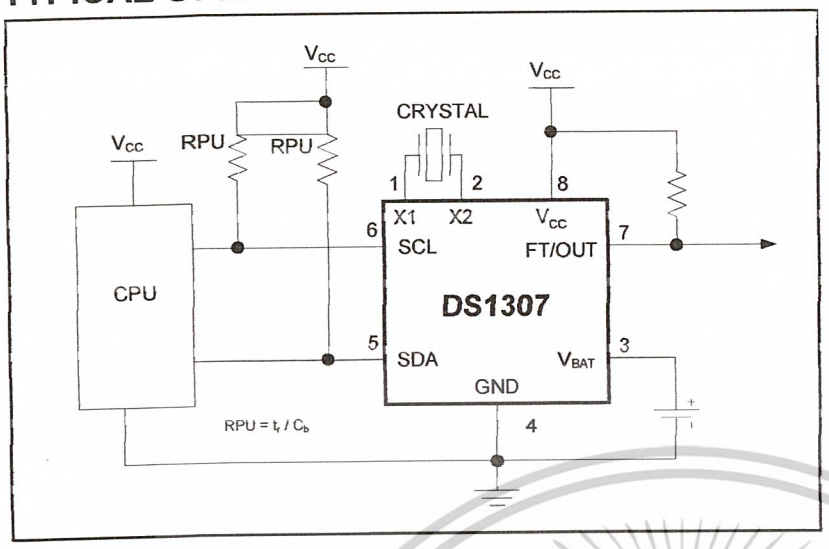
2. **Slave Transmitter Mode (Read Mode):** The first byte is received and handled as in the slave receiver mode. However, in this mode, the direction bit will indicate that the transfer direction is reversed. The DS1307 transmits serial data on SDA while the serial clock is input on SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer (see Figure 6). The slave address byte is the first byte received after the START condition is generated by the master. The slave address byte contains the 7-bit DS1307 address, which is 1101000, followed by the direction bit ($R/\overline{W}$), which is 1 for a read. After receiving and decoding the slave address the DS1307 outputs an acknowledge on SDA. The DS1307 then begins to transmit data starting with the register address pointed to by the register pointer. If the register pointer is not written to before the initiation of a read mode the first address that is read is the last one stored in the register pointer. The register pointer automatically increments after each byte are read. The DS1307 must receive a Not Acknowledge to end a read.

Figure 6. Data Read—Slave Transmitter Mode



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TYPICAL OPERATING CIRCUIT

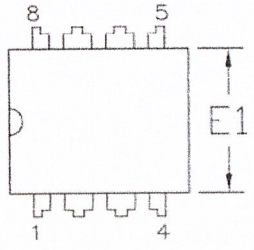


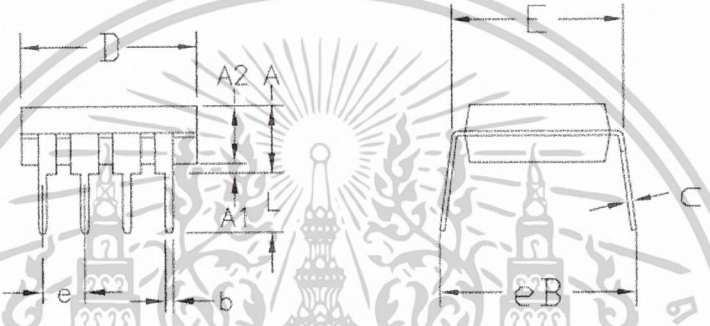
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PACKAGE INFORMATION

(The package drawing(s) in this data sheet may not reflect the most current specifications. For the latest package outline information, go to www.maxim-ic.com/DallasPackInfo.)

REVISIONS			
LTR	DESCRIPTION	DATE	APPROVED
A	NEW DRAWING	12/01	





	8 PIN	
	MIN	MAX
A	–	0.170
A1	0.015	–
A2	0.115	0.195
b	0.015	0.022
c	0.008	0.012
D	0.360	0.380
E	0.300	0.325
E1	0.240	0.260
e	0.090	0.110
L	0.125	0.135
eB	–	0.430

ALL DIMENSIONS ARE IN INCHES

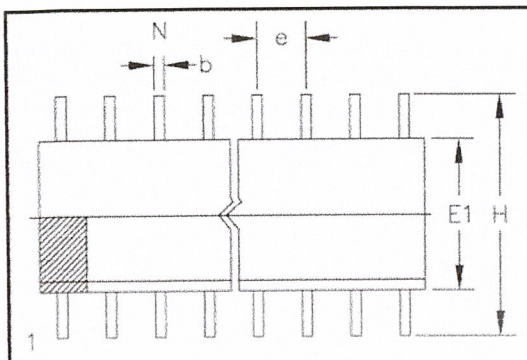
SIGNATURE	DATE	DALLAS SEMICONDUCTOR MAXIM			
DOC. CONTROL:					
ENGR. MGR:		TITLE: MARKETING OUTLINE, 8 LEAD PLASTIC DUAL-IN-LINE PACKAGE (0.300")			
MFG. ENGR:					
CHECKED BY: TWM	12/01	SIZE A	FSCM NO	PART NO. 56-G5005-000	REV A
DRAWN BY: JFD	12/01				
DO NOT SCALE DWG.		SCALE N/A		SHEET 1 OF 1	

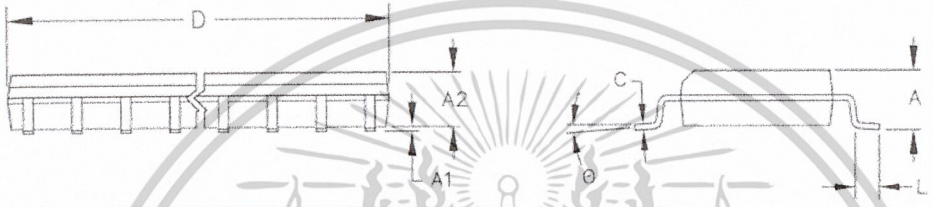
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PACKAGE INFORMATION (continued)

(The package drawing(s) in this data sheet may not reflect the most current specifications. For the latest package outline information, go to www.maxim-ic.com/DallasPackInfo.)

REVISIONS			
LTR	DESCRIPTION	DATE	APPROVED
A	NEW DRAWING	2/95	J.W.
B	UPDATE DIMENSIONS		





PKG		8 PIN		14 PIN		16 PIN	
LTR		MIN	MAX	MIN	MAX	MIN	MAX
A	IN.	0.053	0.069	0.053	0.069	0.053	0.069
	MM	1.35	1.75	1.35	1.75	1.35	1.75
A1	IN.	0.004	0.010	0.004	0.010	0.004	0.010
	MM	0.10	0.25	0.10	0.25	0.10	0.25
A2	IN.	0.048	0.062	0.048	0.062	0.048	0.062
	MM	1.22	1.57	1.22	1.57	1.22	1.57
b	IN.	0.012	0.020	0.012	0.020	0.012	0.020
	MM	0.030	0.51	0.30	0.51	0.30	0.51
C	IN.	0.007	0.011	0.007	0.011	0.007	0.011
	MM	0.18	0.28	0.18	0.28	0.18	0.28
D	IN.	0.188	0.196	0.337	0.344	0.386	0.393
	MM	4.78	4.98	8.56	8.74	9.80	9.98
e	IN.	.050 BSC		.050 BSC		.050 BSC	
	MM	1.27 BSC		1.27 BSC		1.27 BSC	
E1	IN.	0.150	0.158	0.150	0.158	0.150	0.158
	MM	3.81	4.01	3.81	4.01	3.81	4.01
H	IN.	0.230	0.244	0.230	0.244	0.230	0.244
	MM	5.84	6.20	5.84	6.20	5.84	6.20
L	IN.	0.016	0.050	0.016	0.050	0.016	0.050
	MM	0.41	1.27	0.41	1.27	0.41	1.27
θ		0°	8°	0°	8°	0°	8°

THE CHAMFER ON THE BODY IS OPTIONAL. IF IT IS NOT PRESENT, A TERMINAL 1 IDENTIFIER MUST BE POSITIONED SO THAT 1/2 OR MORE OF IT'S AREA IS CONTAINED IN THE HATCHED ZONE.

SIGNATURE		DATE
DOC. CONTROL:		
ENGR. MGR:		
MFG. ENGR:		
CHECKED BY:		
DRAWN BY:	M.W.C.	2/95
DO NOT SCALE DWG.		

DALLAS SEMICONDUCTOR MAXIM

TITLE: PACKAGE OUTLINE .150" SOIC 8,14&16 LD.

SIZE	FSCM NO	PART NO.	REV
A		56-G2008-001	B

SCALE N/A | SHEET 1 OF 1

Maxim/Dallas Semiconductor cannot assume responsibility for use of any circuitry other than circuitry entirely embodied in a Maxim/Dallas Semiconductor product. No circuit patent licenses are implied. Maxim/Dallas Semiconductor reserves the right to change the circuitry and specifications without notice at any time.

Maxim Integrated Products, 120 San Gabriel Drive, Sunnyvale, CA 94086 408-737-7600

© 2004 Maxim Integrated Products • Printed USA

MAXIM is a registered trademark of Maxim Integrated Products, Inc. **DALLAS** is a registered trademark of Dallas Semiconductor Corporation.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ISO²-CMOS MT8870D/MT8870D-1
Integrated DTMF Receiver

ISSUE 3 May1995

Features
- Complete DTMF Receiver
- Low power consumption
- Internal gain setting amplifier
- Adjustable guard time
- Central office quality
- Power-down mode
- Inhibit mode
- Backward compatible with MT8870C/MT8870C-1

Applications
- Receiver system for British Telecom (BT) or CEPT Spec (MT8870D-1)
- Paging systems
- Repeater systems/mobile radio
- Credit card systems
- Remote control
- Personal computers
- Telephone answering machine

Ordering Information

MT8870DE/DE-1	18 Pin Plastic DIP
MT8870DC/DC-1	18 Pin Ceramic DIP
MT8870DS/DS-1	18 Pin SOIC
MT8870DN/DN-1	20 Pin SSOP
MT8870DT/DT-1	20 Pin TSSOP

-40 °C to +85 °C

Description

The MT8870D/MT8870D-1 is a complete DTMF receiver integrating both the bandsplit filter and digital decoder functions. The filter section uses switched capacitor techniques for high and low group filters; the decoder uses digital counting techniques to detect and decode all 16 DTMF tone-pairs into a 4-bit code. External component count is minimized by on chip provision of a differential input amplifier, clock oscillator and latched three-state bus interface.

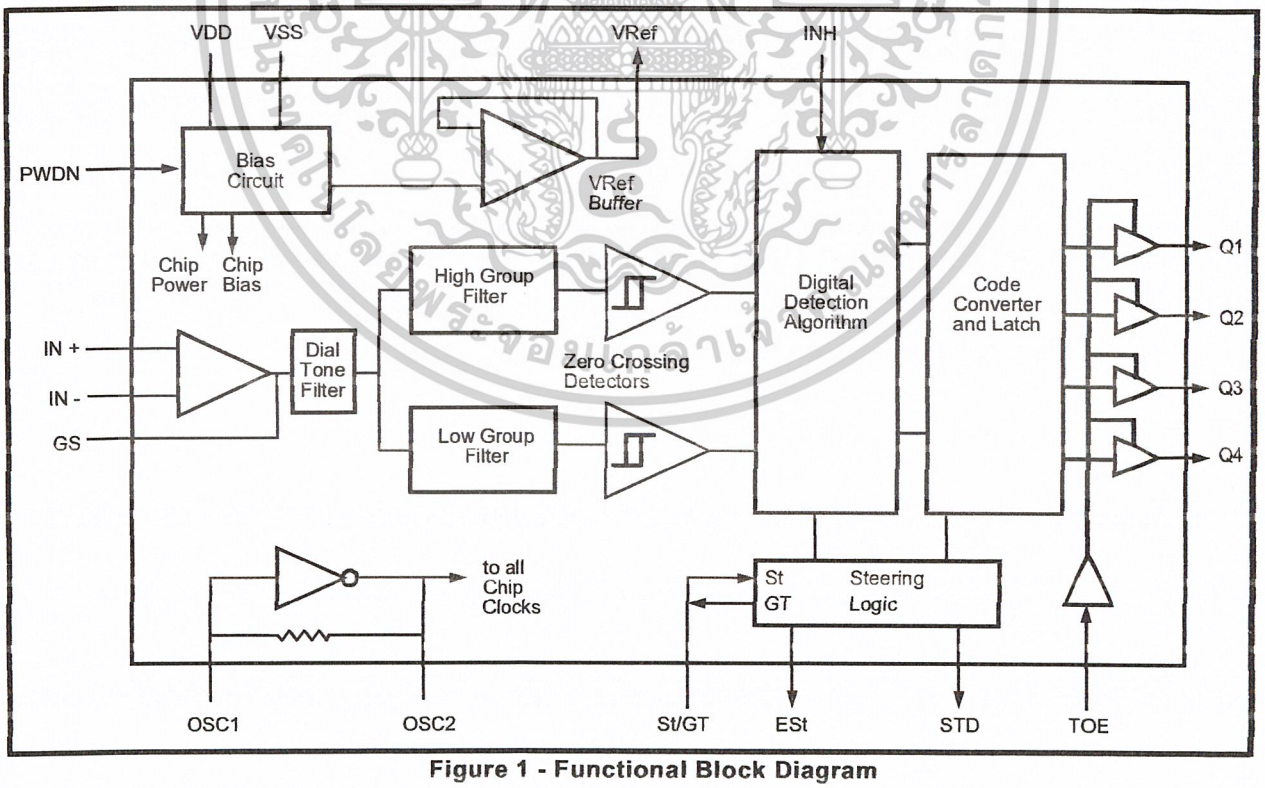


Figure 1 - Functional Block Diagram

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

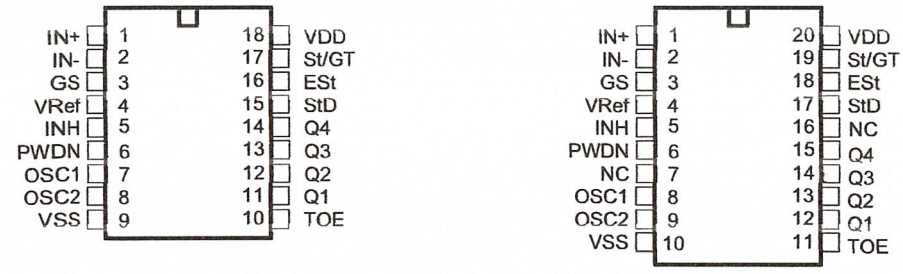


Figure 2 - Pin Connections

Pin Description

Pin #		Name	Description
18	20		
1	1	IN+	**Non-Inverting Op-Amp (Input).**
2	2	IN-	**Inverting Op-Amp (Input).**
3	3	GS	**Gain Select.** Gives access to output of front end differential amplifier for connection of feedback resistor.
4	4	V_{Ref}	**Reference Voltage (Output).** Nominally $V_{DD}/2$ is used to bias inputs at mid-rail (see Fig. 6 and Fig. 10).
5	5	INH	**Inhibit (Input).** Logic high inhibits the detection of tones representing characters A, B, C and D. This pin input is internally pulled down.
6	6	PWDN	**Power Down (Input).** Active high. Powers down the device and inhibits the oscillator. This pin input is internally pulled down.
7	8	OSC1	**Clock (Input).**
8	9	OSC2	**Clock (Output).** A 3.579545 MHz crystal connected between pins OSC1 and OSC2 completes the internal oscillator circuit.
9	10	V_{SS}	**Ground (Input).** 0V typical.
10	11	TOE	**Three State Output Enable (Input).** Logic high enables the outputs Q1-Q4. This pin is pulled up internally.
11-14	12-15	Q1-Q4	**Three State Data (Output).** When enabled by TOE, provide the code corresponding to the last valid tone-pair received (see Table 1). When TOE is logic low, the data outputs are high impedance.
15	17	StD	**Delayed Steering (Output).** Presents a logic high when a received tone-pair has been registered and the output latch updated; returns to logic low when the voltage on St/GT falls below V_{TSt}.
16	18	ESt	**Early Steering (Output).** Presents a logic high once the digital algorithm has detected a valid tone pair (signal condition). Any momentary loss of signal condition will cause ESt to return to a logic low.
17	19	St/GT	**Steering Input/Guard time (Output) Bidirectional.** A voltage greater than V_{TSt} detected at St causes the device to register the detected tone pair and update the output latch. A voltage less than V_{TSt} frees the device to accept a new tone pair. The GT output acts to reset the external steering time-constant; its state is a function of ESt and the voltage on St.
18	20	V_{DD}	**Positive power supply (Input).** +5V typical.
	7, 16	NC	No Connection.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Functional Description

The MT8870D/MT8870D-1 monolithic DTMF receiver offers small size, low power consumption and high performance. Its architecture consists of a bandsplit filter section, which separates the high and low group tones, followed by a digital counting section which verifies the frequency and duration of the received tones before passing the corresponding code to the output bus.

Filter Section

Separation of the low-group and high group tones is achieved by applying the DTMF signal to the inputs of two sixth-order switched capacitor bandpass filters, the bandwidths of which correspond to the low and high group frequencies. The filter section also incorporates notches at 350 and 440 Hz for exceptional dial tone rejection (see Figure 3). Each filter output is followed by a single order switched capacitor filter section which smooths the signals prior to limiting. Limiting is performed by high-gain comparators which are provided with hysteresis to prevent detection of unwanted low-level signals. The outputs of the comparators provide full rail logic swings at the frequencies of the incoming DTMF signals.

Decoder Section

Following the filter section is a decoder employing digital counting techniques to determine the frequencies of the incoming tones and to verify that they correspond to standard DTMF frequencies. A complex averaging algorithm protects against tone simulation by extraneous signals such as voice while providing tolerance to small frequency deviations and variations. This averaging algorithm has been developed to ensure an optimum combination of immunity to talk-off and tolerance to the presence of interfering frequencies (third tones) and noise. When the detector recognizes the presence of two valid tones (this is referred to as the "signal condition" in some industry specifications) the "Early Steering" (ESt) output will go to an active state. Any subsequent loss of signal condition will cause ESt to assume an inactive state (see "Steering Circuit").

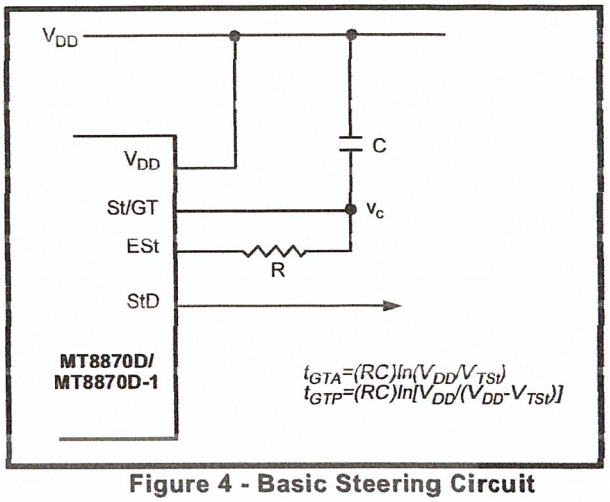


Figure 4 - Basic Steering Circuit

Steering Circuit

Before registration of a decoded tone pair, the receiver checks for a valid signal duration (referred to as character recognition condition). This check is performed by an external RC time constant driven by ESt. A logic high on ESt causes v_c (see Figure 4) to rise as the capacitor discharges. Provided signal

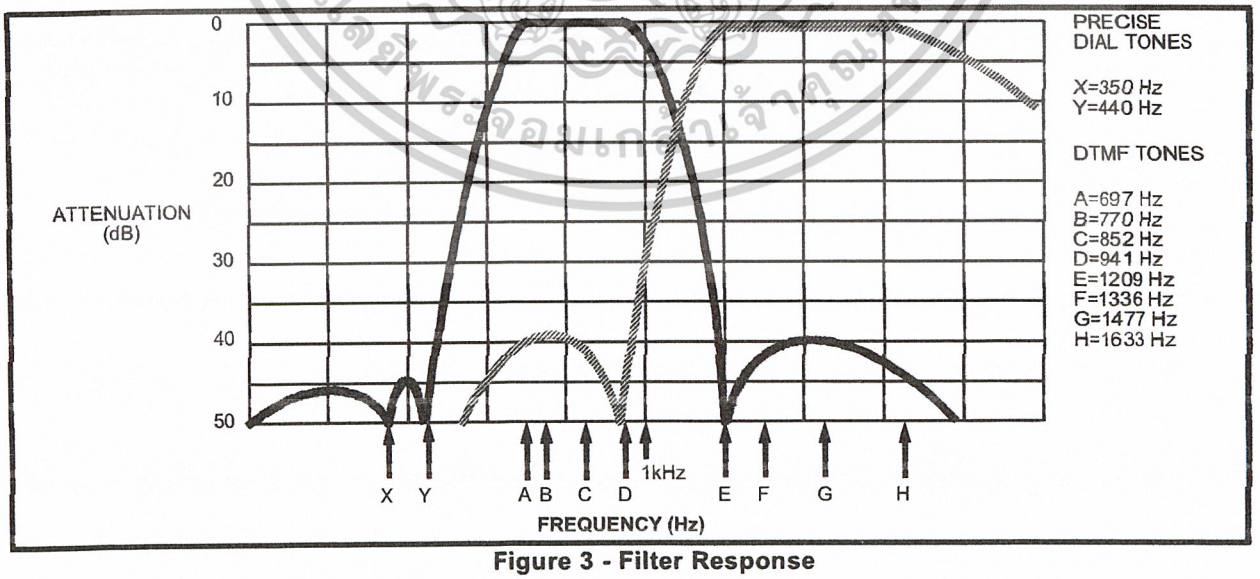


Figure 3 - Filter Response

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

condition is maintained (ESt remains high) for the validation period (t_{GTP}), v_c reaches the threshold (V_{TSt}) of the steering logic to register the tone pair, latching its corresponding 4-bit code (see Table 1) into the output latch. At this point the GT output is activated and drives v_c to V_{DD}. GT continues to drive high as long as ESt remains high. Finally, after a short delay to allow the output latch to settle, the delayed steering output flag (StD) goes high, signalling that a received tone pair has been registered. The contents of the output latch are made available on the 4-bit output bus by raising the three state control input (TOE) to a logic high. The steering circuit works in reverse to validate the interdigit pause between signals. Thus, as well as rejecting signals too short to be considered valid, the receiver will tolerate signal interruptions (dropout) too short to be considered a valid pause. This facility, together with the capability of selecting the steering time constants externally, allows the designer to tailor performance to meet a wide variety of system requirements.

Digit	TOE	INH	ESt	Q_4	Q_3	Q_2	Q_1
ANY	L	X	H	Z	Z	Z	Z
1	H	X	H	0	0	0	1
2	H	X	H	0	0	1	0
3	H	X	H	0	0	1	1
4	H	X	H	0	1	0	0
5	H	X	H	0	1	0	1
6	H	X	H	0	1	1	0
7	H	X	H	0	1	1	1
8	H	X	H	1	0	0	0
9	H	X	H	1	0	0	1
0	H	X	H	1	0	1	0
*	H	X	H	1	0	1	1
#	H	X	H	1	1	0	0
A	H	L	H	1	1	0	1
B	H	L	H	1	1	1	0
C	H	L	H	1	1	1	1
D	H	L	H	0	0	0	0
A	H	H	L	undetected, the output code will remain the same as the previous detected code			
B	H	H	L				
C	H	H	L				
D	H	H	L				

Table 1. Functional Decode Table

L=LOGIC LOW, H=LOGIC HIGH, Z=HIGH IMPEDANCE
X = DON'T CARE

Guard Time Adjustment

In many situations not requiring selection of tone duration and interdigital pause, the simple steering circuit shown in Figure 4 is applicable. Component values are chosen according to the formula:

$$t_{REC}=t_{DP}+t_{GTP}$$
$$t_{ID}=t_{DA}+t_{GTA}$$

The value of t_{DP} is a device parameter (see Figure 11) and t_{REC} is the minimum signal duration to be recognized by the receiver. A value for C of 0.1 µF is recommended for most applications, leaving R to be selected by the designer.

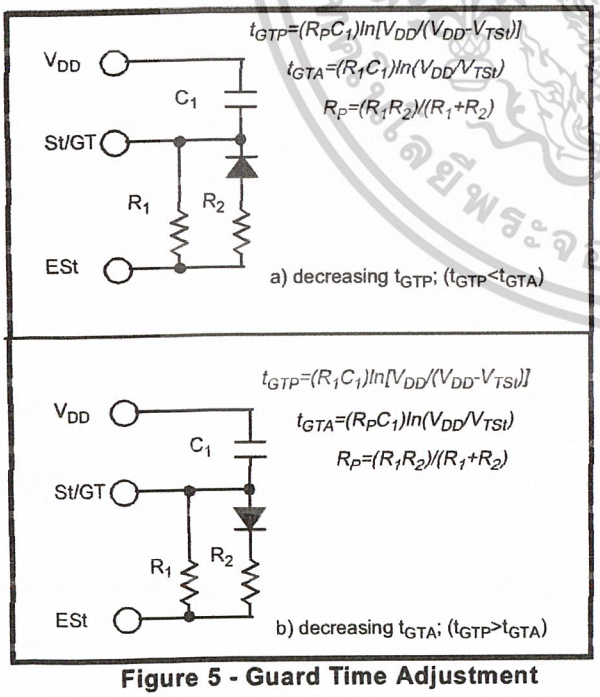


Figure 5 - Guard Time Adjustment

Different steering arrangements may be used to select independently the guard times for tone present (t_{GTP}) and tone absent (t_{GTA}). This may be necessary to meet system specifications which place both accept and reject limits on both tone duration and interdigital pause. Guard time adjustment also allows the designer to tailor system parameters such as talk off and noise immunity. Increasing t_{REC} improves talk-off performance since it reduces the probability that tones simulated by speech will maintain signal condition long enough to be registered. Alternatively, a relatively short t_{REC} with a long t_{DO} would be appropriate for extremely noisy environments where fast acquisition time and immunity to tone drop-outs are required. Design information for guard time adjustment is shown in Figure 5.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Power-down and Inhibit Mode

A logic high applied to pin 6 (PWDN) will power down the device to minimize the power consumption in a standby mode. It stops the oscillator and the functions of the filters.

Inhibit mode is enabled by a logic high input to the pin 5 (INH). It inhibits the detection of tones representing characters A, B, C, and D. The output code will remain the same as the previous detected code (see Table 1).

Differential Input Configuration

The input arrangement of the MT8870D/MT8870D-1 provides a differential-input operational amplifier as well as a bias source (V_{Ref}) which is used to bias the inputs at mid-rail. Provision is made for connection of a feedback resistor to the op-amp output (GS) for adjustment of gain. In a single-ended configuration, the input pins are connected as shown in Figure 10 with the op-amp connected for unity gain and V_{Ref} biasing the input at $^1/_2V_{DD}$. Figure 6 shows the differential configuration, which permits the adjustment of gain with the feedback resistor R_5.

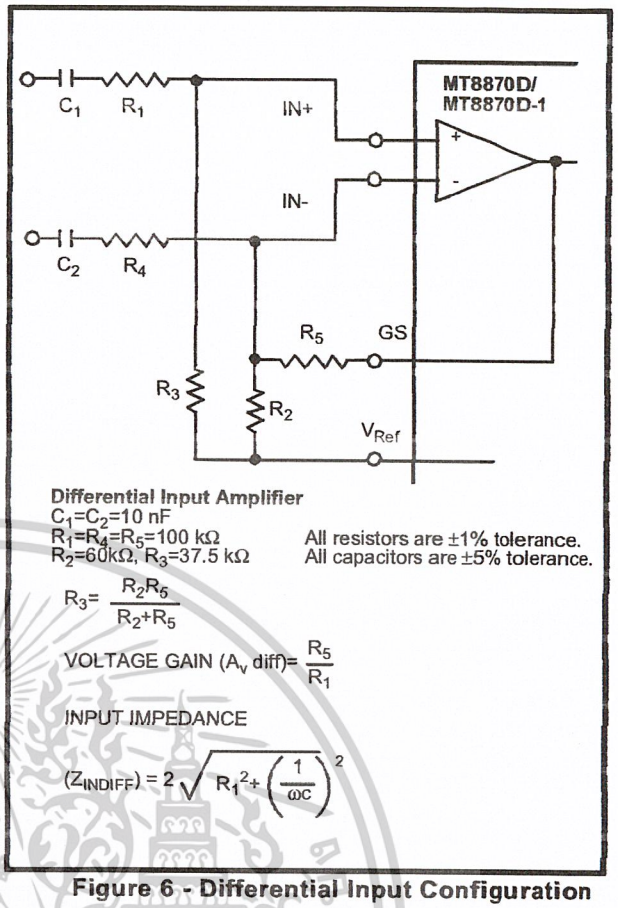


Figure 6 - Differential Input Configuration

Crystal Oscillator

The internal clock circuit is completed with the addition of an external 3.579545 MHz crystal and is normally connected as shown in Figure 10 (Single-Ended Input Configuration). However, it is possible to configure several MT8870D/MT8870D-1 devices employing only a single oscillator crystal. The oscillator output of the first device in the chain is coupled through a 30 pF capacitor to the oscillator input (OSC1) of the next device. Subsequent devices are connected in a similar fashion. Refer to Figure 7 for details. The problems associated with unbalanced loading are not a concern with the arrangement shown, i.e., precision balancing capacitors are not required.

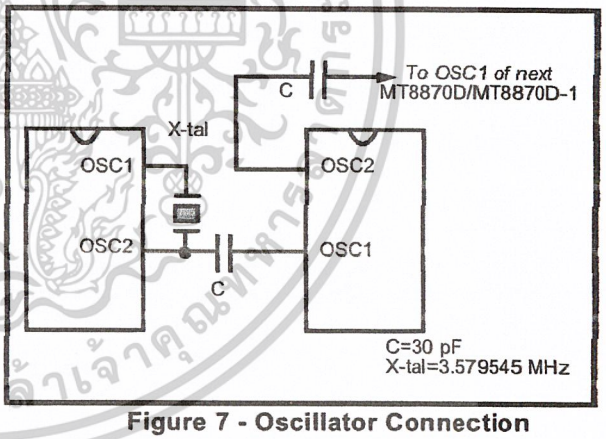


Figure 7 - Oscillator Connection

Parameter	Unit	Resonator
R1	Ohms	10.752
L1	mH	.432
C1	pF	4.984
C0	pF	37.915
Qm	-	896.37
Δf	%	±0.2%

Table 2. Recommended Resonator Specifications

Note: Qm=quality factor of RLC model, i.e., $1/2\Pi fR1C1$.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Applications

RECEIVER SYSTEM FOR BRITISH TELECOM SPEC POR 1151

The circuit shown in Fig. 9 illustrates the use of MT8870D-1 device in a typical receiver system. BT Spec defines the input signals less than -34 dBm as the non-operate level. This condition can be attained by choosing a suitable values of R_1 and R_2 to provide 3 dB attenuation, such that -34 dBm input signal will correspond to -37 dBm at the gain setting pin GS of MT8870D-1. As shown in the diagram, the component values of R_3 and C_2 are the guard time requirements when the total component tolerance is 6%. For better performance, it is recommended to use the non-symmetric guard time circuit in Fig. 8.

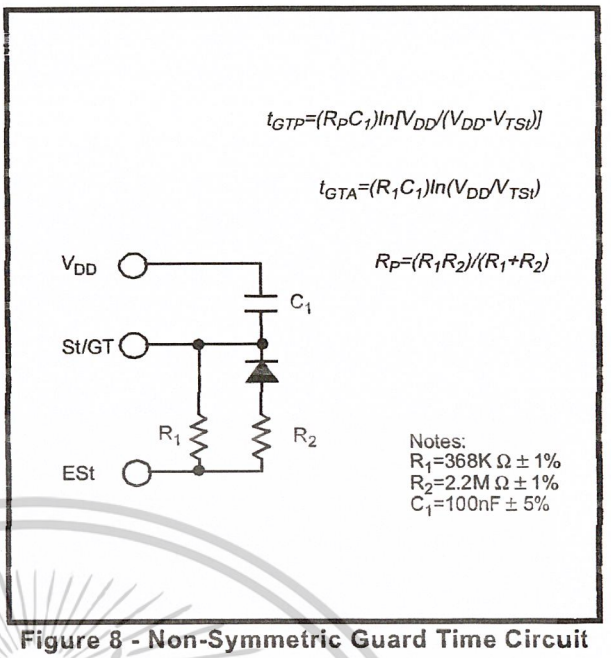


Figure 8 - Non-Symmetric Guard Time Circuit

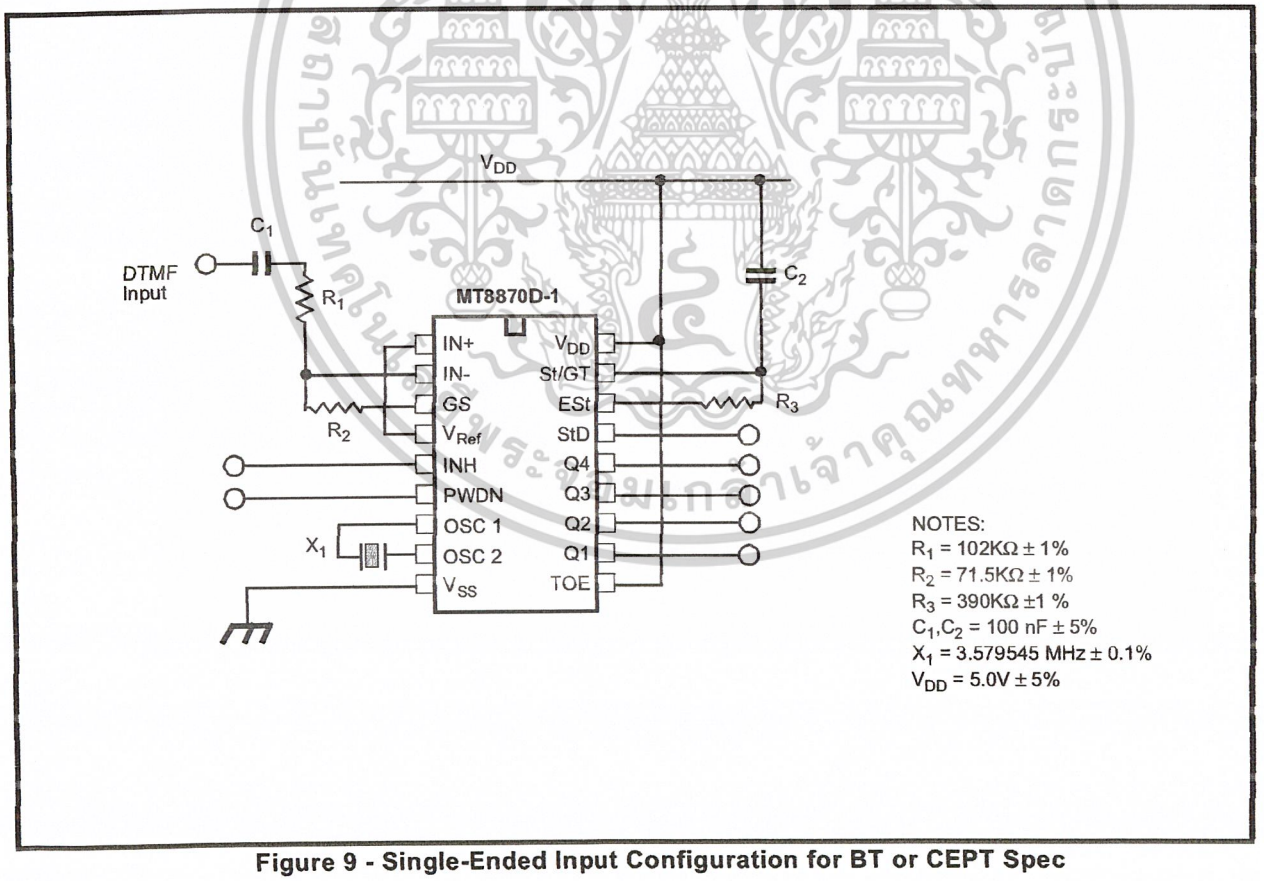


Figure 9 - Single-Ended Input Configuration for BT or CEPT Spec

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Absolute Maximum Ratings†

	Parameter	Symbol	Min	Max	Units
1	DC Power Supply Voltage	V_{DD}		7	V
2	Voltage on any pin	V_I	V_{SS}-0.3	V_{DD}+0.3	V
3	Current at any pin (other than supply)	I_I		10	mA
4	Storage temperature	T_{STG}	-65	+150	°C
5	Package power dissipation	P_D		500	mW

† *Exceeding these values may cause permanent damage. Functional operation under these conditions is not implied.*
Derate above 75 °C at 16 mW / °C. All leads soldered to board.

Recommended Operating Conditions - Voltages are with respect to ground (V_{SS}) unless otherwise stated.

	Parameter	Sym	Min	Typ‡	Max	Units	Test Conditions
1	DC Power Supply Voltage	V_{DD}	4.75	5.0	5.25	V	
2	Operating Temperature	T_O	-40		+85	°C	
3	Crystal/Clock Frequency	fc		3.579545		MHz	
4	Crystal/Clock Freq.Tolerance	Δfc		±0.1		%	

‡ Typical figures are at 25°C and are for design aid only: not guaranteed and not subject to production testing.

DC Electrical Characteristics - V_{DD}=5.0V± 5%, V_{SS}=0V, -40°C ≤ T_O ≤ +85°C, unless otherwise stated.

		Characteristics	Sym	Min	Typ‡	Max	Units	Test Conditions
1	SUPPLY	Standby supply current	I_{DDQ}		10	25	μA	PWDN=V_{DD}
2		Operating supply current	I_{DD}		3.0	9.0	mA	
3		Power consumption	P_O		15		mW	fc=3.579545 MHz
4	INPUTS	High level input	V_{IH}	3.5			V	V_{DD}=5.0V
5		Low level input voltage	V_{IL}			1.5	V	V_{DD}=5.0V
6		Input leakage current	I_{IH}/I_{IL}		0.1		μA	V_{IN}=V_{SS} or V_{DD}
7		Pull up (source) current	I_{SO}		7.5	20	μA	TOE (pin 10)=0, V_{DD}=5.0V
8		Pull down (sink) current	I_{SI}		15	45	μA	INH=5.0V, PWDN=5.0V, V_{DD}=5.0V
9		Input impedance (IN+, IN-)	R_{IN}		10		MΩ	@ 1 kHz
10		Steering threshold voltage	V_{TSt}	2.2	2.4	2.5	V	V_{DD} = 5.0V
11	OUTPUTS	Low level output voltage	V_{OL}			V_{SS}+0.03	V	No load
12		High level output voltage	V_{OH}	V_{DD}-0.03			V	No load
13		Output low (sink) current	I_{OL}	1.0	2.5		mA	V_{OUT}=0.4 V
14		Output high (source) current	I_{OH}	0.4	0.8		mA	V_{OUT}=4.6 V
15		V_{Ref} output voltage	V_{Ref}	2.3	2.5	2.7	V	No load, V_{DD} = 5.0V
16		V_{Ref} output resistance	R_{OR}		1		kΩ	

‡ Typical figures are at 25°C and are for design aid only: not guaranteed and not subject to production testing.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Operating Characteristics - V_{DD}=5.0V±5%, V_{SS}=0V, -40°C ≤ T_O ≤ +85°C ,unless otherwise stated.

Gain Setting Amplifier

	Characteristics	Sym	Min	Typ‡	Max	Units	Test Conditions
1	Input leakage current	I_{IN}			100	nA	$V_{SS} \le V_{IN} \le V_{DD}$
2	Input resistance	R_{IN}	10			MΩ	
3	Input offset voltage	V_{OS}			25	mV	
4	Power supply rejection	PSRR	50			dB	1 kHz
5	Common mode rejection	CMRR	40			dB	0.75 V ≤ V_{IN} ≤ 4.25 V biased at V_{Ref}=2.5 V
6	DC open loop voltage gain	A_{VOL}	32			dB	
7	Unity gain bandwidth	f_C	0.30			MHz	
8	Output voltage swing	V_O	4.0			V_{pp}	Load ≥ 100 kΩ to V_{SS} @ GS
9	Maximum capacitive load (GS)	C_L			100	pF	
10	Resistive load (GS)	R_L			50	kΩ	
11	Common mode range	V_{CM}	2.5			V_{pp}	No Load

MT8870D AC Electrical Characteristics - V_{DD}=5.0V ±5%, V_{SS}=0V, -40°C ≤ T_O ≤ +85°C , using Test Circuit shown in Figure 10.

	Characteristics	Sym	Min	Typ‡	Max	Units	Notes*
1	Valid input signal levels (each tone of composite signal)		-29		+1	dBm	1,2,3,5,6,9
			27.5		869	mV_{RMS}	1,2,3,5,6,9
2	Negative twist accept				8	dB	2,3,6,9,12
3	Positive twist accept				8	dB	2,3,6,9,12
4	Frequency deviation accept		±1.5% ± 2 Hz				2,3,5,9
5	Frequency deviation reject		±3.5%				2,3,5,9
6	Third tone tolerance			-16		dB	2,3,4,5,9,10
7	Noise tolerance			-12		dB	2,3,4,5,7,9,10
8	Dial tone tolerance			+22		dB	2,3,4,5,8,9,11

‡ Typical figures are at 25 °C and are for design aid only: not guaranteed and not subject to production testing.

***NOTES**

1. dBm= decibels above or below a reference power of 1 mW into a 600 ohm load.
2. Digit sequence consists of all DTMF tones.
3. Tone duration= 40 ms, tone pause= 40 ms.
4. *Signal condition consists of nominal DTMF frequencies.*
5. Both tones in composite signal have an equal amplitude.
6. Tone pair is deviated by ±1.5 %± 2 Hz.
7. Bandwidth limited (3 kHz) Gaussian noise.
8. The precise dial tone frequencies are (350 Hz and 440 Hz) ± 2 %.
9. For an error rate of better than 1 in 10,000.
10. Referenced to lowest level frequency component in DTMF signal.
11. Referenced to the minimum valid accept level.
12. Guaranteed by design and characterization.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MT8870D-1 AC Electrical Characteristics - V_{DD}=5.0V±5%, V_{SS}=0V, -40°C ≤ T_O ≤ +85°C , using Test Circuit shown in Figure 10.

	Characteristics	Sym	Min	Typ‡	Max	Units	Notes*
1	Valid input signal levels (each tone of composite signal)		-31		+1	dBm	Tested at V_{DD}=5.0V 1,2,3,5,6,9
			21.8		869	mV_{RMS}	
2	Input Signal Level Reject		-37			dBm	Tested at V_{DD}=5.0V 1,2,3,5,6,9
			10.9			mV_{RMS}	
3	Negative twist accept				8	dB	2,3,6,9,13
4	Positive twist accept				8	dB	2,3,6,9,13
5	Frequency deviation accept		±1.5%± 2 Hz				2,3,5,9
6	Frequency deviation reject		±3.5%				2,3,5,9
7	Third zone tolerance			-18.5		dB	2,3,4,5,9,12
8	Noise tolerance			-12		dB	2,3,4,5,7,9,10
9	Dial tone tolerance			+22		dB	2,3,4,5,8,9,11

‡ Typical figures are at 25 °C and are for design aid only: not guaranteed and not subject to production testing.

***NOTES**

1. dBm= decibels above or below a reference power of 1 mW into a 600 ohm load.
2. Digit sequence consists of all DTMF tones.
3. Tone duration= 40 ms, tone pause= 40 ms.
4. Signal condition consists of nominal DTMF frequencies.
5. Both tones in composite signal have an equal amplitude.
6. Tone pair is deviated by ±1.5 %± 2 Hz.
7. Bandwidth limited (3 kHz) Gaussian noise.
8. The precise dial tone frequencies are (350 Hz and 440 Hz) ± 2 %.
9. For an error rate of better than 1 in 10,000.
10. Referenced to lowest level frequency component in DTMF signal.
11. Referenced to the minimum valid accept level.
12. Referenced to Fig. 10 input DTMF tone level at -25dBm (-28dBm at GS Pin) interference frequency range between 480-3400Hz.
13. Guaranteed by design and characterization.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

AC Electrical Characteristics - V_{DD}=5.0V±5%, V_{SS}=0V, -40°C ≤ To ≤ +85°C , using Test Circuit shown in Figure 10.

		Characteristics	Sym	Min	Typ‡	Max	Units	Conditions
1	TIMING	Tone present detect time	t_{DP}	5	11	14	ms	Note 1
2		Tone absent detect time	t_{DA}	0.5	4	8.5	ms	Note 1
3		Tone duration accept	t_{REC}			40	ms	Note 2
4		Tone duration reject	$t_{\overline{REC}}$	20			ms	Note 2
5		Interdigit pause accept	t_{ID}			40	ms	Note 2
6		Interdigit pause reject	t_{DO}	20			ms	Note 2
7	OUTPUTS	Propagation delay (St to Q)	t_{PQ}		8	11	µs	TOE=V_{DD}
8		Propagation delay (St to StD)	t_{PStD}		12	16	µs	TOE=V_{DD}
9		Output data set up (Q to StD)	t_{QStD}		3.4		µs	TOE=V_{DD}
10		Propagation delay (TOE to Q ENABLE)	t_{PTE}		50		ns	load of 10 kΩ, 50 pF
11		*Propagation delay (TOE to Q DISABLE)*	t_{PTD}		*300*		*ns*	*load of 10 kΩ, 50 pF*
12	PDWN	Power-up time	t_{PU}		30		ms	Note 3
13		Power-down time	t_{PD}		20		ms	
14	CLOCK	Crystal/clock frequency	f_C	3.5759	3.5795	3.5831	MHz	
15		Clock input rise time	t_{LHCL}			110	ns	Ext. clock
16		Clock input fall time	t_{HLCL}			110	ns	Ext. clock
17		Clock input duty cycle	DC_{CL}	40	50	60	%	Ext. clock
18		Capacitive load (OSC2)	C_{LO}			30	pF	

‡ Typical figures are at 25°C and are for design aid only: not guaranteed and not subject to production testing.

*NOTES:
1. Used for guard-time calculation purposes only.
2. These, user adjustable parameters, are not device specifications. The adjustable settings of these minimums and maximums are recommendations based upon network requirements.
3. With valid tone present at input, t_{PU} equals time from PDWN going low until ESt going high.

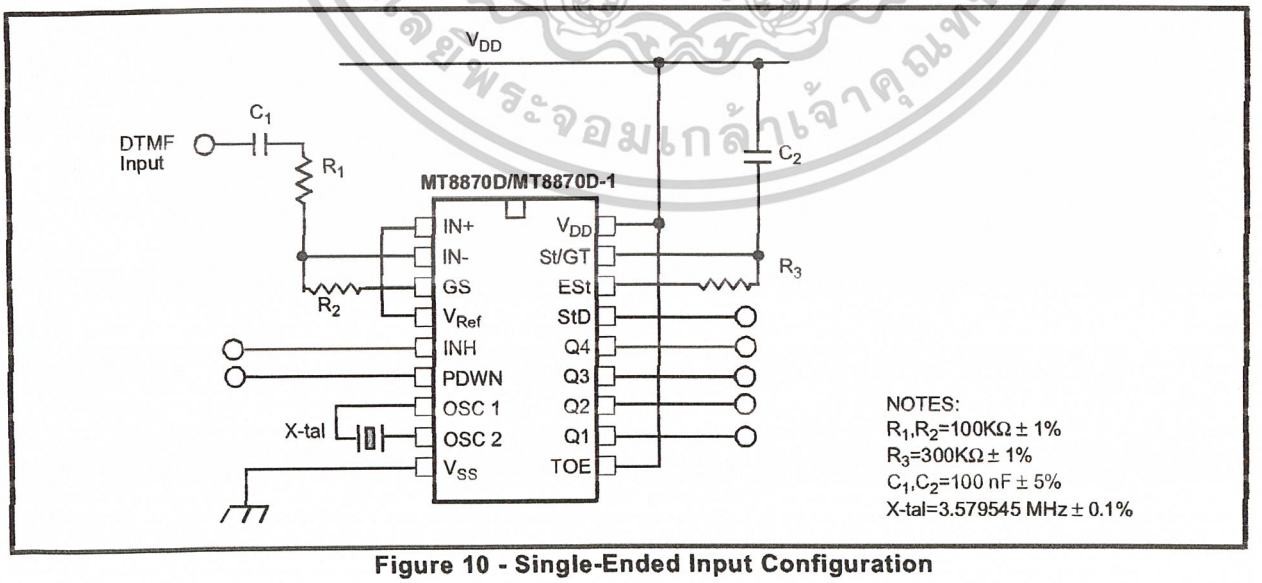
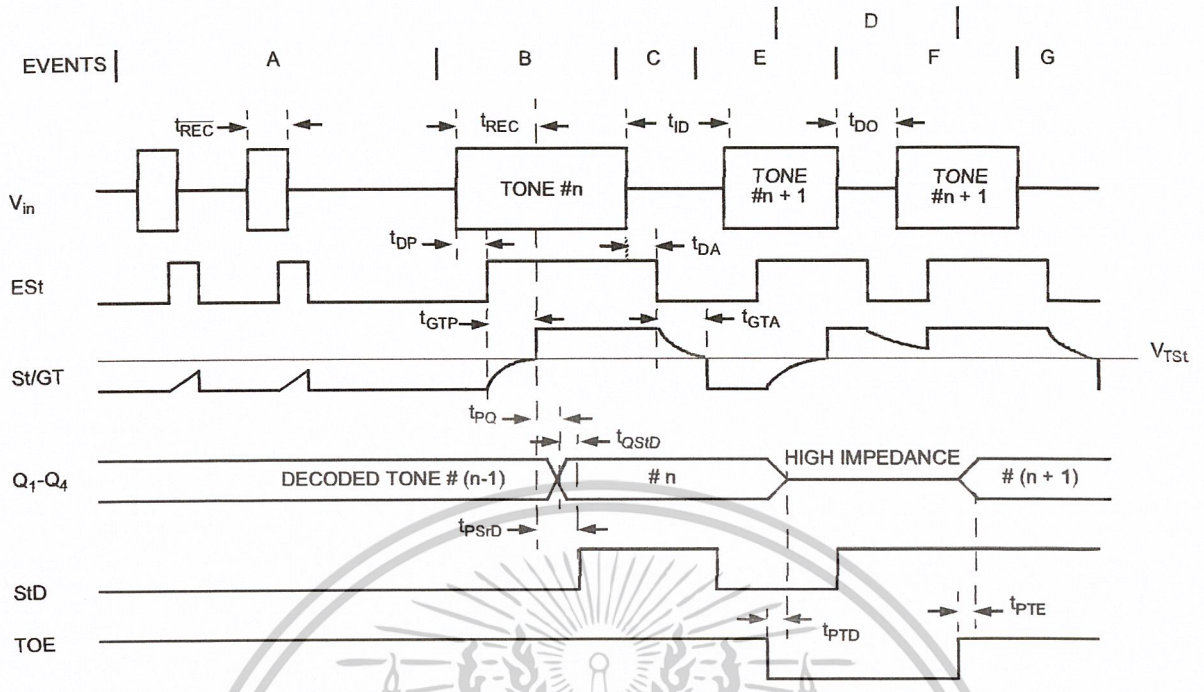


Figure 10 - Single-Ended Input Configuration

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



EXPLANATION OF EVENTS

A) TONE BURSTS DETECTED, TONE DURATION INVALID, OUTPUTS NOT UPDATED.

B) TONE #n DETECTED, TONE DURATION VALID, TONE DECODED AND LATCHED IN OUTPUTS

C) END OF TONE #n DETECTED, TONE ABSENT DURATION VALID, OUTPUTS REMIAN LATCHED UNTIL NEXT VALID TONE.

D) OUTPUTS SWITCHED TO HIGH IMPEDANCE STATE.

E) TONE #n + 1 DETECTED, TONE DURATION VALID, TONE DECODED AND LATCHED IN OUTPUTS (CURRENTLY HIGH IMPEDANCE).

F) ACCEPTABLE DROPOUT OF TONE #n + 1, TONE ABSENT DURATION INVALID, OUTPUTS REMAIN LATCHED.

G) END OF TONE #n + 1 DETECTED, TONE ABSENT DURATION VALID, OUTPUTS REMAIN LATCHED UNTIL NEXT VALID TONE.

EXPLANATION OF SYMBOLS

V_{in}	DTMF COMPOSITE INPUT SIGNAL.
ESt	EARLY STEERING OUTPUT. INDICATES DETECTION OF VALID TONE FREQUENCIES.
St/GT	STEERING INPUT/GUARD TIME OUTPUT. DRIVES EXTERNAL RC TIMING CIRCUIT.
Q_1-Q_4	4-BIT DECODED TONE OUTPUT.
StD	DELAYED STEERING OUTPUT. INDICATES THAT VALID FREQUENCIES HAVE BEEN PRESENT/ABSENT FOR THE REQUIRED GUARD TIME THUS CONSTITUTING A VALID SIGNAL.
TOE	TONE OUTPUT ENABLE (INPUT). A LOW LEVEL SHIFTS Q_1-Q_4 TO ITS HIGH IMPEDANCE STATE.
$t_{\overline{REC}}$	MAXIMUM DTMF SIGNAL DURATION NOT DETECED AS VALID
t_{REC}	MINIMUM DTMF SIGNAL DURATION REQUIRED FOR VALID RECOGNITION
t_{ID}	MAXIMUM TIME BETWEEN VALID DTMF SIGNALS.
t_{DO}	MAXIMUM ALLOWABLE DROP OUT DURING VALID DTMF SIGNAL.
t_{DP}	TIME TO DETECT THE PRESENCE OF VALID DTMF SIGNALS.
t_{DA}	TIME TO DETECT THE ABSENCE OF VALID DTMF SIGNALS.
t_{GTP}	GUARD TIME, TONE PRESENT.
t_{GTA}	GUARD TIME, TONE ABSENT.

Figure 11 - Timing Diagram

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

NOTES:

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้