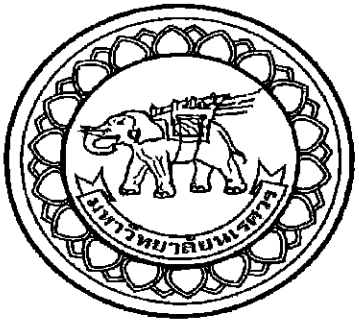


ระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์

Security Alarm System over Phone Network

นายชวาล ตาคำ รหัส 45380026

นายเอกพล ผดุง รหัส 45380182

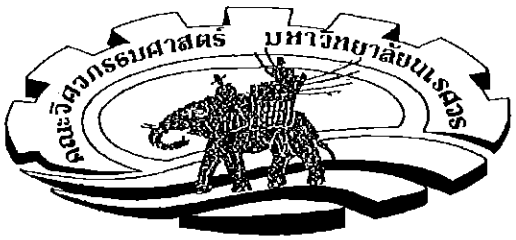
I507547x

ห้องสมุดคณะวิศวกรรมศาสตร์
วันที่รับ 15 พ.ย. 2549
เลขทะเบียน 490161
เลขเรียกหนังสือ ปร.
มหาวิทยาลัยนเรศวร ร2935
2548
e.2

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร



ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน ระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์

ผู้ดำเนินโครงงาน

นายชวาล	ตาคำ	รหัส 45380026
นายเอกพล	ผดุง	รหัส 45380182

อาจารย์ที่ปรึกษา ดร. สุรเชษฐ์ กานต์ประชา

สาขาวิชา วิศวกรรมไฟฟ้า

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2548

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

คณะกรรมการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ
(ดร. สุรเชษฐ์ กานต์ประชา)

.....กรรมการ
(ดร.สมยศ เกียรติวนิชวิไล)

.....กรรมการ
(อาจารย์พนัส นัถฤทธิ์)

หัวข้อโครงงาน ระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์

ผู้ดำเนินโครงงาน นายชวาล ตาคำ รหัส 45380026

นายเอกพล ผดุง รหัส 45380182

อาจารย์ที่ปรึกษา ดร. สุรเชษฐ์ กานต์ประชา

สาขาวิชา วิศวกรรมไฟฟ้า

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2548

.....

บทคัดย่อ

โครงงานชิ้นนี้เกิดขึ้นเนื่องจาก ความต้องการความปลอดภัยในชีวิตและทรัพย์สินของคนในปัจจุบันที่นับวันก็ยิ่งสูงขึ้น จึงเกิดแนวคิดที่จะสร้างระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์ โดยได้มีการนำไมโครคอนโทรลเลอร์มาใช้ในการควบคุมระบบ อุปกรณ์ที่ได้นำมาใช้ในการตรวจจับก็คือ อินฟราเรดเซนเซอร์ และ สวิตซ์แถบแม่เหล็ก ซึ่งคุณสมบัติพิเศษของระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์นี้คือ สามารถส่งสัญญาณแจ้งเตือนไปยังผู้อยู่อาศัยที่อาจไม่ได้อยู่ ณ ที่นั้น ๆ ให้ทราบโดยทันทีเมื่อมีผู้บุกรุก หรือแม้แต่กรณีที่ผู้อยู่อาศัยต้องการที่จะทำการปลดล็อคเพื่อเข้าบ้าน ก็สามารถทำได้โดยการโทรเข้าและกดรหัสจากภายนอกซึ่งเห็นได้ว่าเป็นวิธีที่ทำได้สะดวกอีกทั้งการติดตั้งก็ทำได้ง่ายไม่จำเป็นต้องใช้เครื่องมือหรืออุปกรณ์มากมาย ทำให้ราคาต้นทุนของระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์นี้ไม่สูง แต่สามารถนำไปใช้ในชีวิตประจำวันได้อย่างเกิดประสิทธิผล

Project title	Security Alarm System over Phone Network		
Name	Mr. Chawarn	Tacome	ID. 45380026
	Mr. Aekaphon	Padung	ID. 45380182
Project advisor	Dr. Surachet	Kanprachar	
Major	Electrical Engineering		
Department	Electrical and Computer Engineering		
Academic year	2005		

.....

Abstract

This project was made because of human are in need of to safety in a life and property. So, we have an idea for make Security Alarm System over Phone Network by bring a microcontroller for control the system, Infrared sensor and Magnetic Switch are equipment for check. A special qualification of this system, Infrared sensor and Magnetic Switch is it can send warn direct to the host who doesn't stay in this place suddenly when there is some one to break in to house or if the host want go in to home. They can use phone call and press a code from outside. It is easy method to install and unnecessary to use an equipment. So, a cost of this system is cheap and it can use effectiveness in a daily life.

กิตติกรรมประกาศ

ขอขอบพระคุณ ดร. สุรเชษฐ์ กานต์ประชา อาจารย์ที่ปรึกษา ที่คอยให้คำปรึกษาและให้ความช่วยเหลือตลอดจนคำแนะนำต่างๆในการทำโครงงานชิ้นนี้ สุดท้ายต้องขอขอบพระคุณอาจารย์ทุกท่านและเพื่อนๆ พี่ ๆทุกคนที่ยังไม่ได้เอ่ยนามที่ให้คำแนะนำและให้การสนับสนุนผู้จัดทำโครงงานให้สามารถทำโครงงานชิ้นนี้จนสำเร็จลุล่วงไปได้ด้วยดี

นายชวาล ตาคำ

นายเอกพล ผดุง

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

1.1 หลักการและเหตุผล

สภาพสังคมปัจจุบันผู้อยู่อาศัยในบ้านมีความเสี่ยงทั้งทางร่างกายและทรัพย์สิน จากพวกมิจฉาชีพ ซึ่ง ได้มีการนำเอาวิธีต่าง ๆ มาใช้ในการป้องกัน เช่น การเลี้ยงสุนัขไว้เฝ้าบ้าน การฝากบ้านไว้กับตำรวจ หรือ การจ้างพนักงานรักษาความปลอดภัย ซึ่งวิธีดังกล่าว เป็นที่นิยม แต่อาจทำให้เกิดความสิ้นเปลือง และใช้งบประมาณค่อนข้างสูง จึงได้ มีการใช้สัญญาณเตือนภัยแจ้งเตือนเมื่อมีการบุกรุกจากพวกมิจฉาชีพ

สัญญาณเตือนภัยมีหลายประเภทแล้วแต่ลักษณะการใช้งาน แต่สัญญาณเตือนภัยส่วนใหญ่จะมีราคาค่อนข้างสูงและจะมีการแจ้งเตือนในบริเวณที่มีผู้บุกรุกเท่านั้น ปัญหาที่เกิดขึ้นคือ เจ้าของทรัพย์สินไม่สามารถทราบได้ว่าขณะนี้มีการบุกรุกที่พักอาศัยของตนอยู่ เมื่อเจ้าของทรัพย์สินไม่ได้อยู่ในสถานที่นั้นจากสภาพดังกล่าวจึงเกิดแนวความคิดที่จะทำสัญญาณเตือนภัยที่สามารถแจ้งเตือนไปยังเจ้าของทรัพย์สินซึ่งไม่ได้อยู่ในสถานที่นั้น ๆ โดยเมื่อมีการบุกรุกจากพวกมิจฉาชีพ สัญญาณเตือนภัยจะทำการตรวจจับความเคลื่อนไหวและส่งสัญญาณเตือนในสถานที่นั้นแล้วแจ้งไปยังเจ้าของสถานที่ให้ทราบ ทำให้เจ้าของสถานที่ทราบทันทีว่ามีการบุกรุก และสามารถแก้ไขปัญหาได้อย่างทันท่วงที

1.2 วัตถุประสงค์

1. เพื่อสร้างระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์ที่ใช้ต้นทุนต่ำ แต่มีประสิทธิภาพสูงปลอดภัยที่มีประสิทธิภาพสูง
2. สร้างระบบสัญญาณเตือนภัยที่มีขนาดกะทัดรัด สามารถนำไปติดตั้งได้สะดวก
3. สามารถนำระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์ไปใช้งานได้จริงกับที่อยู่อาศัยหรือร้านค้าต่าง ๆ ได้อย่างมีประสิทธิภาพ

1.3 ขอบข่ายงาน

โครงงานชิ้นนี้เป็นการสร้างระบบสัญญาณเตือนภัย ซึ่งมีหัวข้อหลักที่จะทำการศึกษาเพื่อสร้างชิ้นงานดังนี้

1. ศึกษาการทำงานของวัสดุอิเล็กทรอนิกส์ เช่น อินฟราเรดเซนเซอร์ สวิตช์แถบแม่เหล็กไซเรน และ รีเลย์ เป็นต้น
2. ศึกษาการเขียนโปรแกรมควบคุมอินฟราเรดเซนเซอร์และสวิตช์แถบแม่เหล็กในการทำงาน

3. ศึกษาการเขียนโปรแกรมควบคุมบอร์ดบันทึกเสียงและบอร์ดโทรศัพท์

4. วิเคราะห์และออกแบบขั้นตอนการทำงานของระบบทั้งหมด

5. เขียนโปรแกรมควบคุมการทำงานของระบบ

6. นำอุปกรณ์หลัก ๆ แต่ละตัวคือ อินฟราเรดเซนเซอร์ บอร์ดบันทึกเสียง สวิตซ์แถบแม่เหล็ก บอร์ดโทรศัพท์ บอร์ดควบคุม รีเลย์ และ ไซเรนมาต่อเข้าด้วยกัน

7. ทดสอบการใช้งาน แก้ไขและปรับปรุง

1.4 กิจกรรมดำเนินการ

กิจกรรม	ปี 2547		ปี 2548									
	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.
1 เขียนโครงการทำงาน	↔											
2. รวบรวมข้อมูลและเอกสาร		↔										
3. วิเคราะห์และออกแบบชิ้นงาน			←——	——	—→							
4. สร้างและทดสอบชิ้นงาน						←——	——	——	—→			
5. ปรับปรุงและแก้ไขชิ้นงาน										←——	—→	
6. จัดทำเอกสารและคู่มือการใช้งาน												↔

1.5 ผลที่คาดว่าจะได้รับ

1. ได้ระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์โดยส่งสัญญาณเป็นเสียงเรียกเข้าและมีเสียงไซเรนดัง

2. ได้ฝึกการออกแบบโดยใช้ไมโครคอนโทรเลอร์ในการควบคุมระบบ

3. นำความรู้ในการดัดแปลงและออกแบบวงจรต่างๆ ไปประยุกต์ใช้ด้านอื่นได้
4. ประดิษฐ์แล้วนำไปเสนอขายเป็นรายได้เสริม
5. เป็นเอกสารอ้างอิงเพื่อใช้ในการทำงาน และค้นคว้าต่อไปได้
6. ฝึกการทำงานเป็นกลุ่มและการกำหนดระยะเวลาการทำงานได้อย่างมีประสิทธิภาพ

1.6 งบประมาณ

1. ค่าวัสดุอุปกรณ์ และเครื่องมือ	1500	บาท
2. ค่าจัดทำรูปเล่ม	500	บาท
รวมทั้งหมด	2000	บาท

บทที่ 2
หลักการและทฤษฎี

จากแนวคิดที่จะสร้างระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์ โดยประกอบด้วยอุปกรณ์หลัก ๆ คือ บอร์ดโทรศัพท์ บอร์ดบันทึกเสียง สวิตซ์แถบแม่เหล็ก และอินฟราเรดเซนเซอร์ โดยอุปกรณ์แต่ละตัวก็จะมีคุณสมบัติ หลักการทำงาน หรือแม้แต่การนำไปใช้งานที่แตกต่างกันไป ส่วนการเชื่อมต่อกับไมโครคอนโทรลเลอร์เพื่อที่จะนำมาเขียนโปรแกรมควบคุมระบบกาการทำงานของอุปกรณ์ต่าง ๆ โดยมีหลักการและทฤษฎีพื้นฐานทางไมโครคอนโทรลเลอร์ที่ควรจะทราบดังนี้

ไมโครคอนโทรลเลอร์ คือ อุปกรณ์ประเภทสารกึ่งตัวนำที่รวบรวมฟังก์ชั่นการทำงานต่างๆ ไว้ภายในตัวของมันเอง โดยมีโครงสร้างใกล้เคียงกับคอมพิวเตอร์ คือ ภายในประกอบด้วยหน่วยรับข้อมูลและโปรแกรม หน่วยประมวลผล หน่วยความจำ หน่วยแสดงผล ซึ่งส่วนประกอบเหล่านี้มีความสมบูรณ์ในตัวของมันเอง ทำให้มีขนาดเล็ก และสามารถเขียนโปรแกรมควบคุมการทำงานของอุปกรณ์ต่างๆ ที่เชื่อมต่อกับตัวมัน ง่ายต่อการนำไปประยุกต์ใช้งาน

2.1 การเชื่อมต่อกับไมโครคอนโทรลเลอร์ [1]

2.1.1 คุณสมบัติของไมโครคอนโทรลเลอร์ MCS-51

1. เป็นไมโครคอนโทรเลอร์ขนาด 8 บิต

2. มีหน่วยความจำภายในแบบแฟลชขนาด 4 กิโลไบต์ หรือ 8 กิโลไบต์ ที่โปรแกรมภายในวงจรสามารถเขียนและลบได้ถึงพันครั้ง

3. มีสายสัญญาณสำหรับต่อกับอินพุต/เอาต์พุตได้ 32 เส้น (แบบ 2 สองทิศทาง)

4. มีหน่วยความจำชั่วคราว (ROM) ภายในขนาด 128 ไบต์ หรือ 256 กิโลไบต์

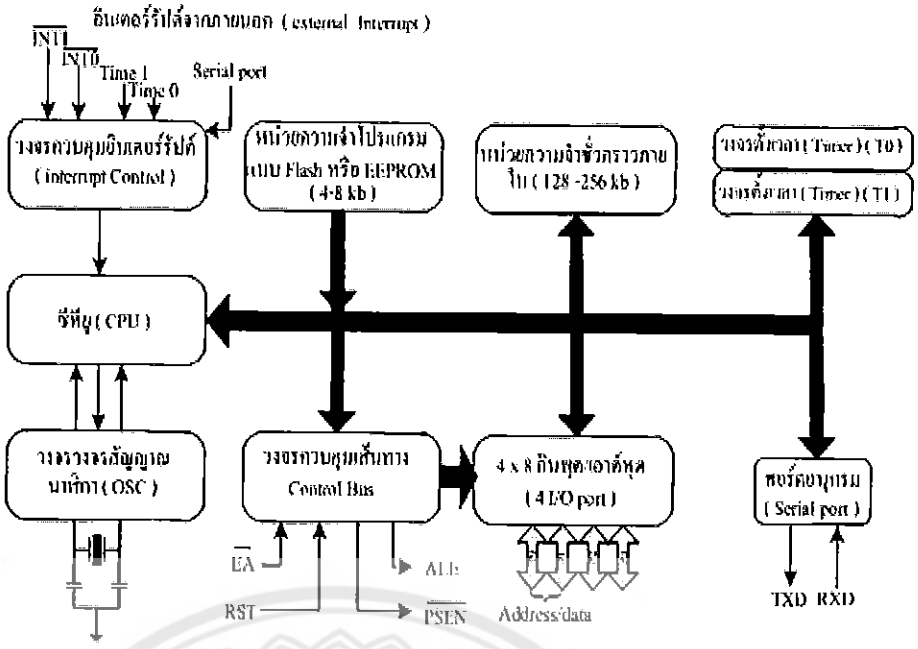
5. ใช้ความถี่สัญญาณนาฬิกาตั้งแต่ 0 Hz จนถึง 24 Hz

6. มีวงจรตั้งเวลาและนับสัญญาณเวลาขนาด 16 บิต จำนวน 2 หรือ 3 ชุด

7. มีวงจรรับสัญญาณอินเตอร์รัปได้ไม่ต่ำกว่า 6 ชนิด

8. สามารถต่อขยายหน่วยความจำภายนอกได้สูงสุด 64 กิโลไบต์

9. มีวงจรสื่อสารแบบสื่อสาร 2 ทางเต็มอัตรา และมีคำสั่งที่ใช้ภาษาแอสแซมบลีทั้งหมด 111 คำสั่ง โครงสร้างพื้นฐานของ MCS-51 มีส่วนประกอบต่างๆ ดังในรูปที่ 2.1



รูปที่ 2.1 โครงสร้างพื้นฐานของไมโครคอนโทรลเลอร์ [1]

รายการไอซีไมโครคอนโทรลเลอร์ตระกูล MCS-51 ที่ผู้ผลิตได้สร้างขึ้นมาหลายรุ่นนั้นก็เพื่อให้เหมาะสมกับการประยุกต์ใช้งานแต่ละประเภท ดังตารางที่ 2.1 แสดงจำนวนของหน่วยความจำภายใน วงจรตั้งเวลา/นับเวลา และระดับของการอินเตอร์รัปต์ของแต่ละรุ่น

ตารางที่ 2.1 ตารางแสดงรายละเอียดของไมโครคอนโทรลเลอร์ตระกูล MCS-51 [1]

ไมโครคอนโทรลเลอร์	หน่วยความจำภายใน (internal memory)		ตั้งเวลา/นับเวลา (time/counter)	สัญญาณอินเทอร์รัปต์จากภายนอก
	หน่วยความจำภายในแบบ EPROM , EEPROM	ข้อมูล RAM		
8051	4 kb × 8 Rom	128 ×8 bit	2 × 16 bit	6
8051AH	4 kb × 8 Rom	128 ×8 bit	2 × 16 bit	5
8051AH	8 kb × 8 Rom	256 × 8 bit	3 × 16 bit	6
8031AH	ไม่มี	128 ×8 bit	2 × 16 bit	5
8032AH	ไม่มี	256 × 8 bit	3 × 16 bit	5
8031	ไม่มี	128 ×8 bit	2 × 16 bit	5
8751H	4 kb × 8 Rom	128 ×8 bit	2 × 16 bit	5
8751H-12	4 kb × 8 Rom	128 ×8 bit	2 × 16 bit	5

นอกจากนี้ยังมีอีกหลายบริษัทที่ทำการผลิตไมโครคอนโทรลเลอร์ MCS-51 พร้อมทั้งมีการพัฒนาเพิ่มเติมขึ้น ในส่วนของความเร็วและหน่วยความจำภายในโดยใช้หน่วยความจำแบบแฟลช

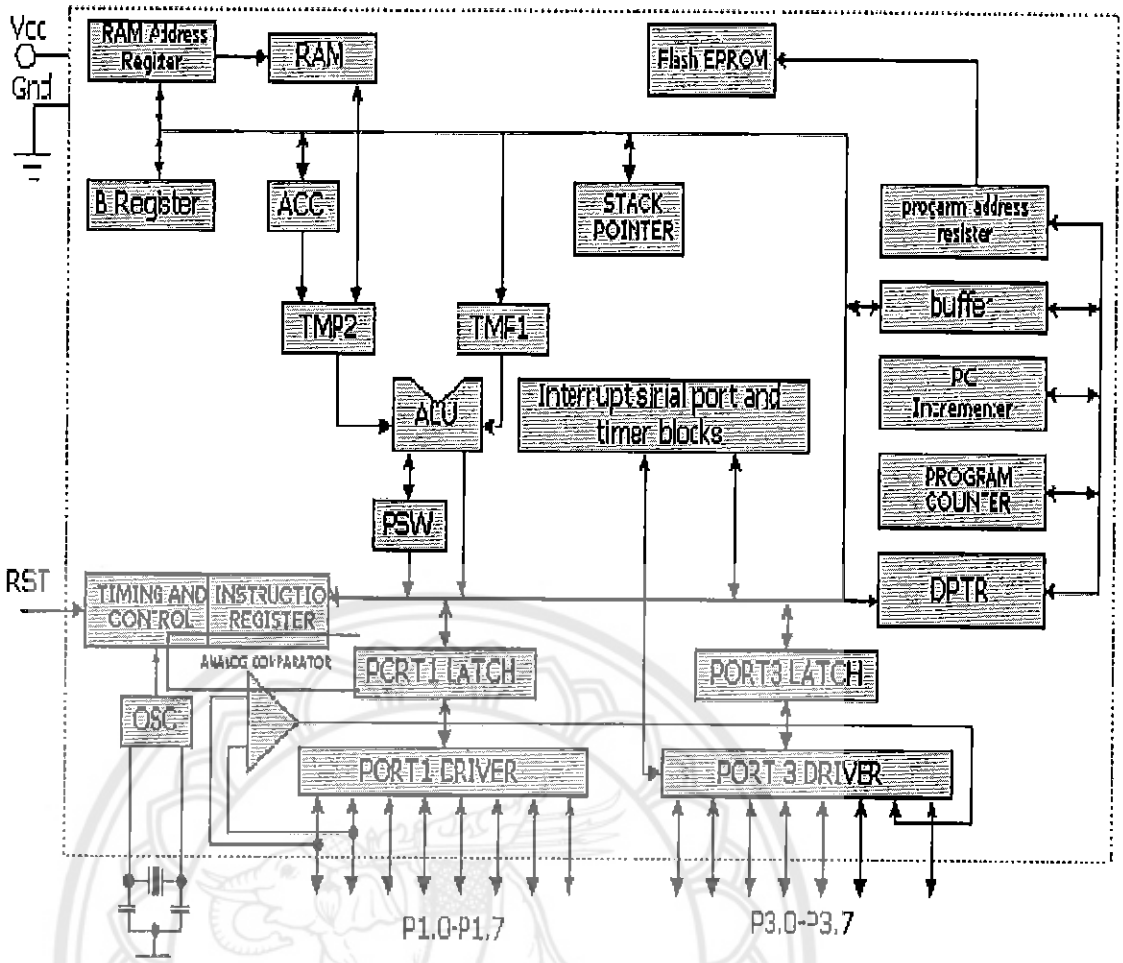
(Flash Memory) ทำให้ประยุกต์ใช้งานได้ง่ายและเป็นที่นิยมใช้กันมากขึ้น ส่วนโครงสร้างและคำสั่งของโปรแกรมก็ยังใช้เหมือนเดิม แสดงตัวอย่างดังตารางที่ 2.2

ตารางที่ 2.2 ไมโครคอนโทรลเลอร์ตระกูล MCS-51 ที่ผลิตโดยบริษัท ATMEL [1]

ไมโครคอนโทรลเลอร์	หน่วยความจำภายใน (internal memory)		ตั้งเวลา/นับเวลา (time/counter)	สัญญาณอินเทอร์รัปต์จากภายนอก
	หน่วยความจำภายในแบบ EPROM , EEPROM	ข้อมูล RAM		
AT89C1051	1 kb × 8	64 ×8 bit	2 × 16 bit	6
AT89C2051	2 kb × 8	128 ×8 bit	2 × 16 bit	6
AT89C4051	4 kb × 8	128 × 8 bit	2 × 16 bit	6
AT89C51	4 kb × 8	128 ×8 bit	2 × 16 bit	6
AT89C52	8 kb × 8	256 × 8 bit	3 × 16 bit	8
AT89S52	8 kb × 8	256 × 8 bit	3 × 16 bit	8
AT89C55	20 kb × 8	256 × 8 bit	3 × 16 bit	8
AT89S8252	8 KB × 8 (2 kb EEPROM)	256 × 8 bit	3 × 16 bit	9
AT89S53	12 kb × 8	256 × 8 bit	3 × 16 bit	9

2.1.2 โครงสร้างภายในของไมโครคอนโทรลเลอร์ตระกูล MCS-51

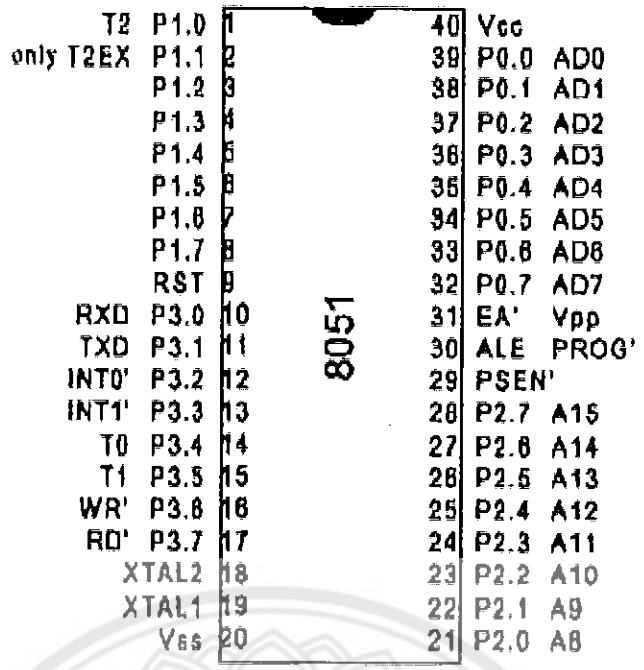
วงจรภายในของไมโครคอนโทรลเลอร์ตระกูล MCS-51 ประกอบด้วยวงจรอินพุตและเอาต์พุตทั้งหมด 4 พอร์ต แต่ละพอร์ตจะเป็น 8 บิต หน่วยความจำโปรแกรมภายใน (EPROM , EEPROM และ Flash) หน่วยความจำที่เป็นข้อมูลนั้น (RAM) ซึ่งอยู่จะในวงจรหลักของไมโครคอนโทรลเลอร์ตระกูล MCS-51 ตลอดจนวงจรการคำนวณทางคณิตศาสตร์และลอจิก (ALU) วงจรรีจีสเตอร์ทั่วไป และรีจีสเตอร์ฟังก์ชั่นการใช้งานเฉพาะ แสดงดังรูปที่ 2.2



รูปที่ 2.2 โครงสร้างภายในของไมโครคอนโทรลเลอร์ MCS-51 [1]

2.1.3 การจัดตำแหน่งขาของไมโครคอนโทรลเลอร์ตระกูล MCS-51

ไมโครคอนโทรลเลอร์ตระกูล MCS-51 ทุกเบอร์นั้นจะมีโครงสร้างและการใช้งานพื้นฐานเหมือนกันดังตัวอย่างนี้ เช่น แบบดิ๊ป (DIP) ซึ่งมีทั้งหมด 40 ขา ได้แบ่งการใช้งานออกเป็น อินพุต/เอาต์พุต (Input/Output Port) ขาสัญญาณควบคุม ขาสัญญาณกำหนดตำแหน่งหน่วยความจำ และขาสัญญาณข้อมูล ดังรูปที่ 2.3



รูปที่ 2.3 ตำแหน่งขาของไมโครคอนโทรลเลอร์ตระกูล MCS-51 [1]

ตำแหน่งของไมโครคอนโทรลเลอร์ตระกูล MCS-51และหน้าที่การทำงาน

1. P0.0-P0.7 (ขาที่ 32-39) พอร์ต 0 ทำหน้าที่เป็นสัญญาณควบคุมอุปกรณ์ภายนอกได้ 2 ทิศทาง สามารถรับข้อมูลอินพุตและส่งข้อมูลเอาต์พุตได้ มีขนาด 8 บิต การตั้งค่าให้พอร์ต 0 รับข้อมูลอินพุตทำได้โดยการส่งค่าสัญญาณ 1 ไปยังบิตที่ต้องการให้รับข้อมูลอินพุต วงจรภายในจะทำให้บิตนั้นมีค่าความต้านทานสูงและสามารถรับข้อมูลอินพุตได้ และยังใช้เป็นขาสัญญาณกำหนดตำแหน่งหน่วยหน่วยความจำ (A0-A7) และขาสัญญาณข้อมูล (D0-D7)โดยการใช้ตัวแยกสัญญาณ (D-latch 74LS373) ทำหน้าที่เป็นมัลติเพล็กซ์ (Multiplex) โดยเลือกช่วงเวลาของสัญญาณกำหนดตำแหน่งหน่วยความจำและสัญญาณข้อมูลออกจากกัน

ในขณะที่ใช้เป็นอินพุตและเอาต์พุต วงจรภายในจะไม่ม่วงจรเพิ่มกระแสไฟฟ้า (Pull up) จึงจำเป็นต้องต่อวงจรเพิ่มกระแสไฟฟ้าจากภายนอกเข้าไป

2. P1.0-P1.7 (ขาที่ 1.8) พอร์ต 1 ทำหน้าที่เป็นสัญญาณควบคุมการอุปกรณ์ภายนอกได้ 2 ทิศทาง สามรถปรับได้ทั้งอินพุตและเอาต์พุต มีขนาด 8 บิต สามารถอ้างอิงถึงการทำงานได้ที่ละบิต และวงจรภายในมีตัวต้านทานเพิ่มกระแสไฟฟ้า (Pull up) ในกรณีที่ต้องการให้รับข้อมูลอินพุตก็สามารถทำได้เหมือนพอร์ต 0

3. P2.0-P2.7 (ขาที่ 21-28) พอร์ต 2 ทำหน้าที่เป็นสัญญาณควบคุมอุปกรณ์ภายนอกได้ทั้ง 2 ทิศทาง คือ เป็นได้เป็นทั้งอินพุตและเอาต์พุต มีขนาด 8 บิต สามารถใช้เป็นขาสัญญาณที่กำหนดตำแหน่งหน่วยความจำ (A8-A15) และมีวงจรเพิ่มกระแสไฟภายใน การกำหนดให้เป็นขาอินพุตทำได้

โดยการส่งข้อมูลสถานะ 1 ไปยังบิตที่ต้องการให้เป็นอินพุต ก็จะสามารถทำการรับค่าข้อมูลอินพุตได้

4. P3.0-P3.7 (ขาที่ 10-17) พอร์ต 3 ทำหน้าที่เป็นสัญญาณควบคุมอุปกรณ์ภายนอกอินพุตและเอาต์พุต 2 ทิศทาง มีขนาด 8 บิต คุณสมบัติทั่วไปจะเหมือนกับพอร์ตอื่นๆ แต่จะมีคุณสมบัติที่ต่างออกไปคือใช้ทำหน้าที่พิเศษเป็นสัญญาณควบคุมการทำงานต่างๆของไมโครคอนโทรลเลอร์ ดังตารางที่ 2.3

ตารางที่ 2.3 หน้าที่พิเศษของพอร์ต 3 [1]

บิตของพอร์ต	สัญญาณ	หน้าที่การทำงาน
P3.0	RXD	รับข้อมูลจากพอร์ตอนุกรม (serial input port)
P3.1	TXD	ส่งข้อมูลจากพอร์ตอนุกรม (serial output port)
P3.2	$\overline{INT0}$	รับสัญญาณอินเตอร์รัปต์หมายเลข 0 (external interrupt 0)
P3.3	$\overline{INT1}$	รับสัญญาณอินเตอร์รัปต์หมายเลข 1 (external interrupt 1)
P3.4	T0	ใช้ตั้งเวลานับเวลาตัวที่ 0 (Time: 0 external input)
P3.5	T1	ใช้ตั้งเวลานับเวลาตัวที่ 1 (Time: 1 external input)
P3.6	$\overline{WR}$	เป็นสัญญาณเขียนข้อมูลหน่วยความจำหรืออุปกรณ์ภายนอก (external data memory write strobe)
P3.7	$\overline{RD}$	เป็นสัญญาณอ่านข้อมูลหน่วยความจำหรืออุปกรณ์ภายนอก (external data memory write strobe)

5. $\overline{PSEN}$ (Program Store Store Enable ขาที่ 29) ขาที่ทำงานที่สภาวะลอจิกเป็น "0" ไมโครคอนโทรลเลอร์ต้องอ่านค่าจากหน่วยความจำภายนอกที่เป็นข้อมูล โดยโปรแกรมจะเก็บในหน่วยความจำถาวร (ROM , EPROM , EEPROM) ส่วนมากใช้ต่อเป็นขาเลือกทำงาน (Enable: $\overline{OE}$) แต่ถ้าไมโครคอนโทรลเลอร์ใช้หน่วยความจำภายใน ขานี้ก็จะไม่ได้ใช้งาน และมีค่าลอจิกเป็น "1"

6. ขา ALE (Address Latch Enable ขาที่ 30) ทำหน้าที่ควบคุมการทำงานของสัญญาณกำหนดตำแหน่งกับสัญญาณข้อมูล โดยใช้การเลือกเส้นทาง (data select หรือ multiplex) โดยปกติเมื่อไมโครคอนโทรลเลอร์ทำงานจะส่งสัญญาณกำหนดตำแหน่งออกมาก่อน พร้อมกับส่งสัญญาณให้ขา ALE ทำงาน เพื่อเลือกให้สัญญาณกำหนดตำแหน่ง (A0-A7) ผ่านไอซี (74LS373) ที่ทำหน้าที่เลือกเส้นทางถ้าส่งสัญญาณข้อมูลออกมา ไอซี (74LS373) จะไม่ทำงาน ข้อมูลก็จะถูกส่งไปที่สายสัญญาณข้อมูล

7. ขา $\overline{EA}$ (External Access ขาที่ 31) ทำหน้าที่เลือกการทำงานของหน่วยความจำ ถ้ามีค่าลอจิกเป็น "1" หมายถึง ใช้ข้อมูลจากหน่วยความจำภายในตัวไมโครคอนโทรลเลอร์ แต่ถ้ามีค่าลอจิกเป็น "0" หมายถึง ใช้ข้อมูลจากหน่วยความจำภายนอก

8.ขา RST (Reset ขาที่ 9) ทำหน้าที่เริ่มต้นการทำงานใหม่ของไมโครคอนโทรลเลอร์ การทำงานที่ค่าลอจิก "1" นี้จะทำให้ไมโครคอนโทรลเลอร์เริ่มต้นทำงานที่ตำแหน่ง 0000 เพื่ออ่านข้อมูลโปรแกรมและจัดระบบการทำงาน

8. ขาสัญญาณนาฬิกา (ขาที่ 18-19) ซึ่งจะทำหน้าที่เป็นตัวกำหนดสัญญาณนาฬิกาให้กับไมโครคอนโทรลเลอร์ใช้เป็นฐานเวลาในการทำงาน โดยจะใช้แผ่นผลึก (crystal) ที่มีความถี่ตั้งแต่ 0-24 เมกกะเฮิรตซ์ (MHz) ร่วมกับตัวเก็บประจุขนาด 20-33 pF

9. แหล่งจ่ายไฟ (Power supply) ขาที่ 20 จะเป็นขากราวด์ (Ground) และขาที่ 40 จะเป็นแหล่งจ่ายไฟบวกให้กับไมโครคอนโทรลเลอร์ ซึ่งใช้แหล่งจ่ายไฟขนาดไม่เกิน 5 โวลต์

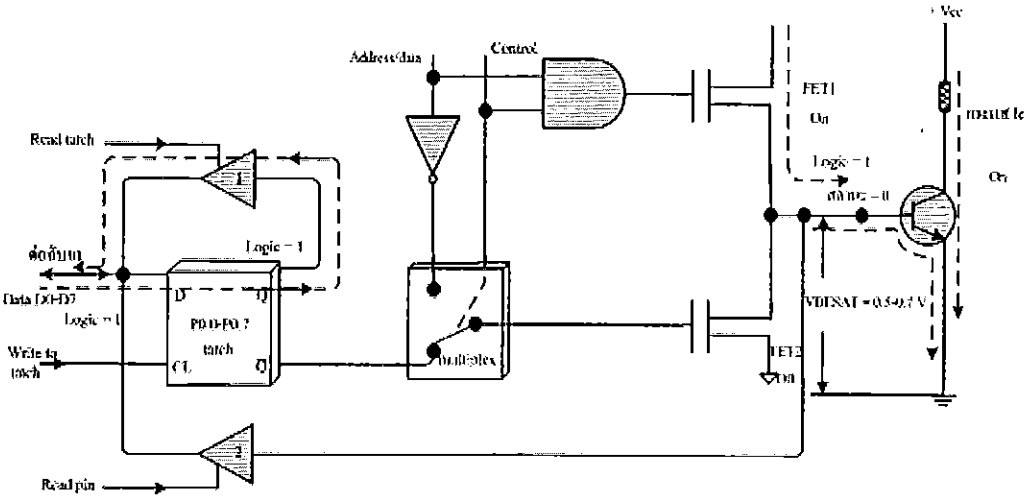
2.1.4 โครงสร้างและการทำงานของพอร์ตอินพุต/เอาต์พุต (I/O Structure)

วงจรอินพุตและเอาต์พุตของพอร์ตไมโครคอนโทรลเลอร์ตระกูล MCS-51ที่ศึกษานี้เป็นตัวอย่างวงจรการทำงานของพอร์ตละบิต ในส่วนของพอร์ตเอาต์พุตที่ค้างสถานะ (Latch) จะใช้วงจร ฟลิปฟลอป ซึ่งรับข้อมูลจากสายสัญญาณข้อมูลภายใน (internal data bus) โดยสัญญาณที่เขียนนี้จะไปที่เอาต์พุตของไมโครคอนโทรลเลอร์

การอ่านข้อมูลจากอินพุตของแต่ละพอร์ตมีการทำงาน 2 วิธีคือ การอ่านข้อมูลจากภายนอกโดยตรง ซึ่งใช้เป็นสัญญาณควบภายในที่อ่านจากขา (read pin) ที่ส่งมาจากไมโครคอนโทรลเลอร์ โดยข้อมูลจะผ่านบัฟเฟอร์ตัวที่ 2 เข้าไปที่สายสัญญาณข้อมูลภายใน และวิธีที่ 2 เป็นการอ่านค่าข้อมูลอินพุตจากเอาต์พุตของวงจรดีฟลิปฟลอบที่ขา Qโดยใช้สัญญาณการควบคุมการอ่านข้อมูลที่ค้างอยู่ (read leach) จากไมโครคอนโทรลเลอร์ โดยข้อมูลจะผ่านบัฟเฟอร์ตัวที่ 1 เข้าไปยังสายสัญญาณข้อมูลภายใน

การทำงานของอินพุต/เอาต์พุต

ในกรณีที่นี้จะทำการอ่านข้อมูลอินพุตจากเอาต์พุตของวงจรดีฟลิปฟลอปที่ขา Q โดยจะใช้สัญญาณควบคุมการอ่านข้อมูลที่ค้างไว้อยู่ ซึ่งเป็นการอ่านข้อมูลที่ถูกเขียนไว้แล้ว (read-modifi-write) สาเหตุที่ใช้วิธีนี้เนื่องจากในบางกรณีไมโครคอนโทรลเลอร์ตระกูล MCS-51ไมโครคอนโทรลเลอร์อาจเกิดการสับสนของสถานะข้อมูลที่ขาของแต่ละพอร์ต และเมื่อนำสัญญาณแต่ละบิตของพอร์ตไปต่อกับวงจรภายนอกที่ใช้งานทรานซิสเตอร์ การควบคุมต้องใช้ไบแอสที่ขาเบสทรานซิสเตอร์ ถ้าไมโครคอนโทรลเลอร์ส่งสัญญาณที่ขามีสถานะเป็น 1 ออกไปไบแอส และทำให้ทรานซิสเตอร์ทำงาน แรงดันไฟฟ้าระหว่างขาเบส (base) กับขาอิมิตเตอร์ (emitter) จะมีค่าประมาณ 0.5-0.7 โวลต์ V_{BE} (sat) ซึ่งมีค่าเทียบสถานะ "0" ไมโครคอนโทรลเลอร์จึงแยกไม่ออก ดังรูปที่ 2.4



รูปที่ 2.4 พอร์ตและวงจรทรานซิสเตอร์ในกรณีเอาต์พุตมีสถานะลอจิกเป็น "1" [1]

2.1.5 คำสั่งในการใช้งานของพอร์ตอินพุต/เอาต์พุต

การอ่านค่าของข้อมูลอินพุตจากสภาวะสัญญาณที่ค้างอยู่ที่ขา Q โดยใช้สัญญาณควบคุมการอ่านข้อมูลที่ค้างอยู่ ไมโครคอนโทรลเลอร์ตระกูล MCS-51 จะมีคำสั่งในการใช้งานทั้งหมด 11 คำสั่ง นอกเหนือจากนั้นจะเป็นการอ่านค่าจากพอร์ตโดยตรง ดังตารางที่ 2.4

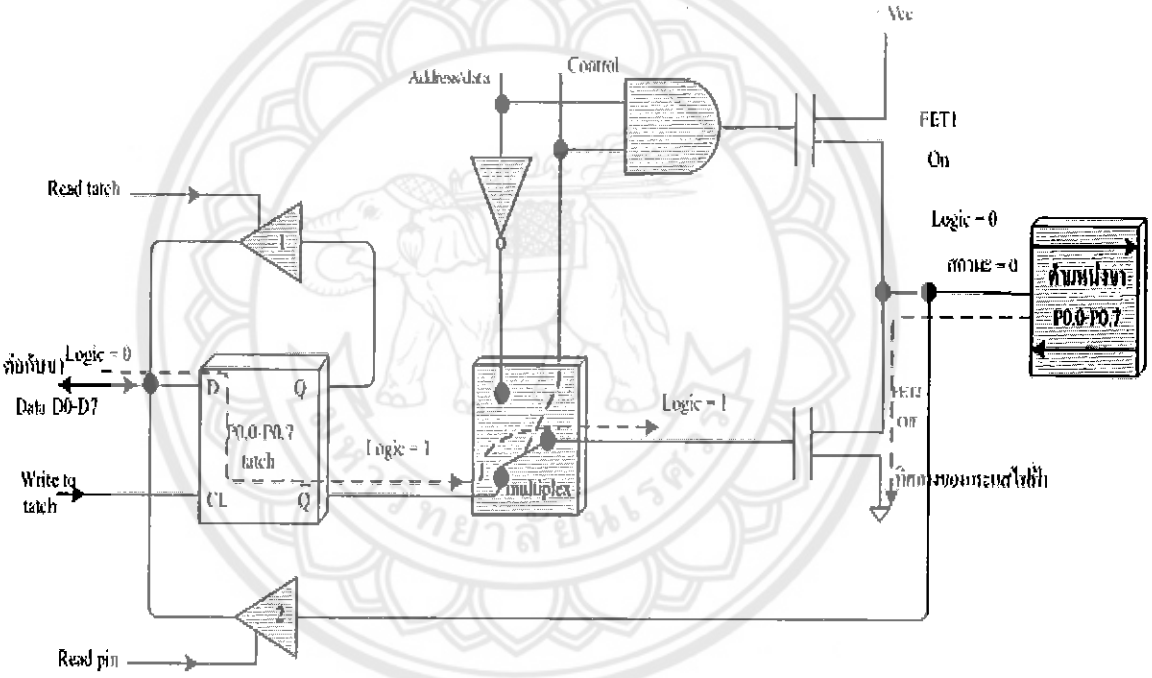
ตารางที่ 2.4 คำสั่งที่ใช้ในการอ่านข้อมูลที่ค้างอยู่ [1]

คำสั่ง	การทำงาน
ANL	การกระทำลอจิก AND ทั่วไป ANL P1,A
ORL	การกระทำลอจิก OR ทั่วไป และ ORL,P1,A
XRL	การกระทำลอจิก EX-OR ทั่วไป และ XRL P3,A
JBC	กระโดดไปถ้าบิต = 1 และยกเลิกบิตทั่วไป และกระโดดไปเมื่อ P1, = 1
CPL	การกลับค่าบิตทั่วไป (จาก 1 →0, 0 →1) และพอร์ต 3 บิต 0
INC	เพิ่มค่าอีกค่าหนึ่ง ข้อมูลทั่วไป และค่าที่อยู่พอร์ต 2
DEC	ลดค่าอีกค่าหนึ่ง ข้อมูลทั่วไป และค่าที่อยู่พอร์ต 2
DJNZ	ลดค่า ถ้าไม่เท่ากับ 0 กระโดดไปตำแหน่งที่กำหนดและใช้พอร์ต 3
MOV,PXY,C	การเคลื่อนย้ายบิตตัวทดระหว่างบิตของพอร์ต
CLR PX,Y	การยกเลิก (0) บิตของพอร์ต
SET PX,T	การกำหนดบิตเป็น 1 ของพอร์ต

2.1.6 ตำแหน่งของพอร์ต 0

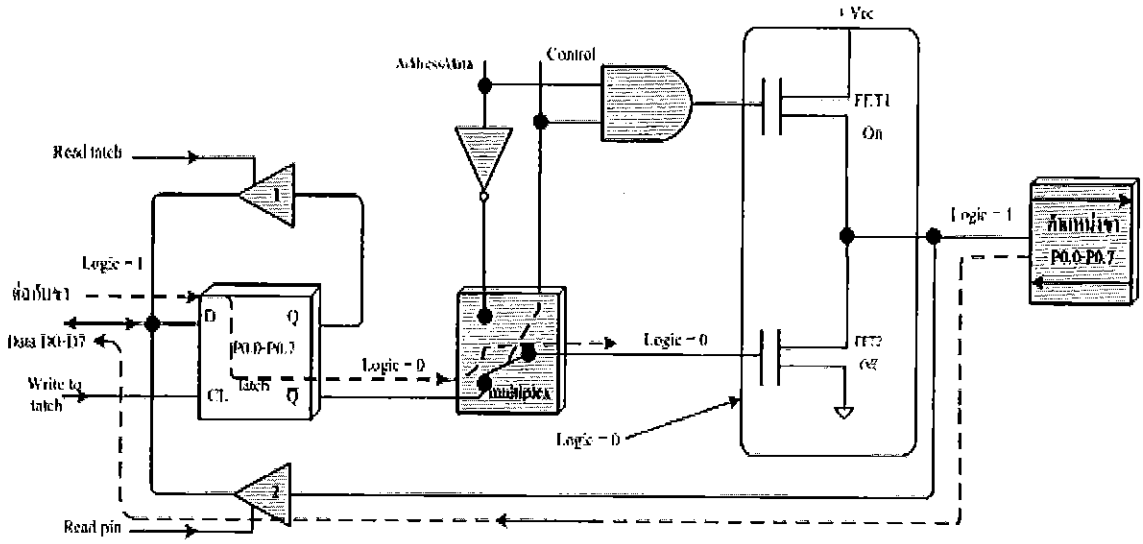
พอร์ต 0 (P0.1-P0.7) นอกจากเป็นพอร์ต/เอาต์พุตทั่วไป ยังเป็นสายสัญญาณกำหนดตำแหน่งหน่วยความจำ (A0-A7) และสายสัญญาณข้อมูล โดยใช้สัญญาณควบคุม (ALE) เลือกทำงาน และไม่มีวงจรตัวต้านทานเพิ่มกระแสไฟฟ้าภายใน แต่จะใช้เฟตตัวที่ 1 (FET) ทำหน้าที่แทน โดยให้เอาต์พุตมีสถานะลอจิกเป็น "1" ในขณะติดต่อกับหน่วยความจำภายนอก ในกรณีอื่นนอกเหนือจากนี้เฟตตัวนี้จะไม่ทำงาน (off)

การทำงานของวงจรพอร์ต 0 เมื่อต้องการส่งข้อมูลที่มีสถานะลอจิกเป็น "0" ออกไปแสดงผลภายนอกก็สามารถส่งข้อมูลออกไปได้เลยโดยไม่ต้องกำหนดค่าใดๆ ข้อมูลจะผ่านวงจร ฟลิปฟลอปออกไปทางขา $\overline{Q}$ โดยมีสถานะลอจิกเป็น "1" และผ่านวงจรเลือกสัญญาณไปไบอัสให้เฟตตัวที่ 2 ทำงาน (ON) ผลก็คือจะได้ค่าสถานะลอจิกทางเอาต์พุตเป็น "0" ดังรูปที่ 2.5



รูปที่ 2.5 การทำงานของพอร์ต 0 เมื่อส่งข้อมูลที่มีสถานะลอจิกเป็น "0" [1]

การทำงานของพอร์ต 0 เมื่อเป็นอินพุตคือ จะส่งข้อมูล 1 (FFH) ไปที่พอร์ต 0 ทุกบิต และให้เฟตทั้งสองตัวหยุดทำงาน (OFF) ผลที่เกิดขึ้นคือ พอร์ต 0 เหมือนถูกตัดออกจากวงจรทุกบิต แต่ละบิตจึงสามารถใช้เป็นขาอินพุตได้โดยการอ่านข้อมูลจากภายนอกโดยตรงบัฟเฟอร์ตัวที่ 2 ดังรูปที่ 2.6

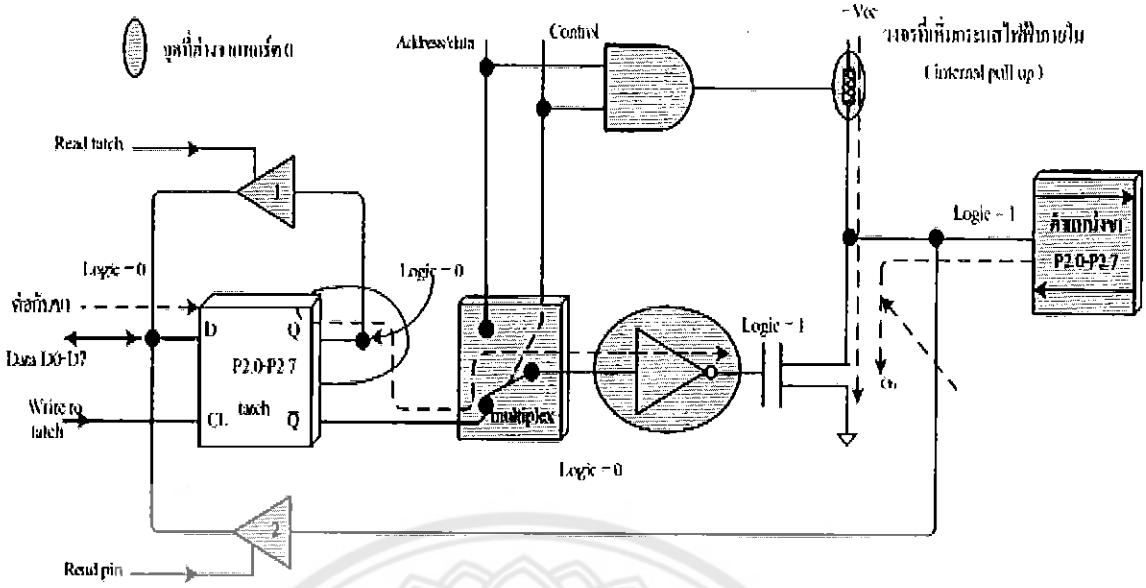


รูปที่ 2.6 การทำงานของพอร์ต 0 เมื่อเป็นอินพุต [1]

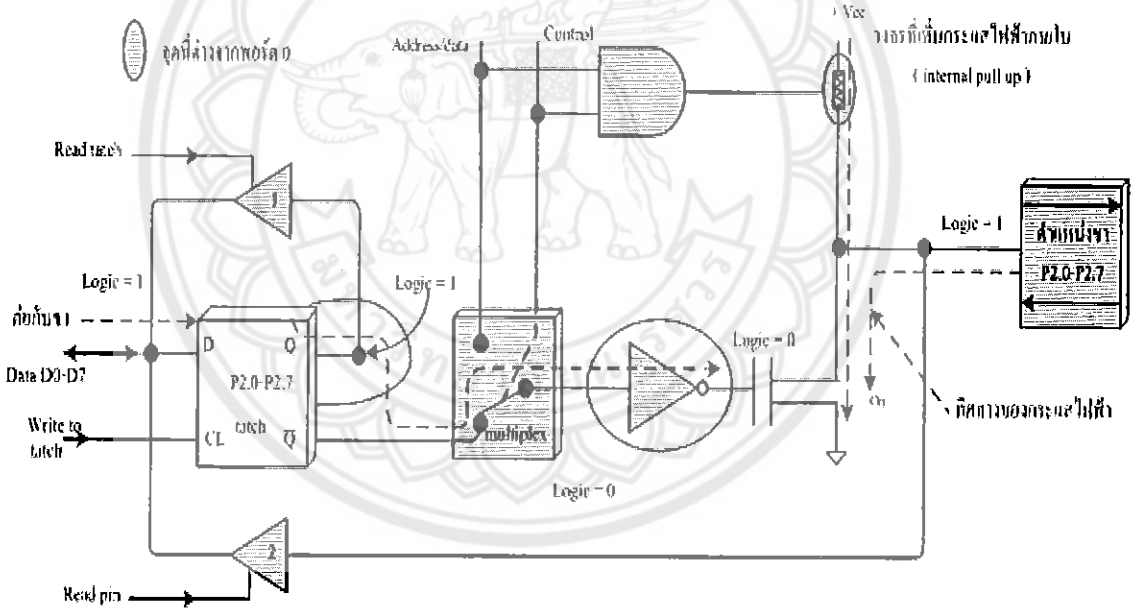
2.1.7 ตำแหน่งของพอร์ต 2

พอร์ต 2 (P2.0-P2.7) มีการทำงานและโครงสร้างคล้ายกับพอร์ต 0 นอกจากเป็นอินพุต/เอาต์พุตทั่วไปแล้ว ยังเป็นสายสัญญาณกำหนดตำแหน่งหน่วยความจำ (A8-A15) โดยใช้สัญญาณควบคุม ALE เลือกทำงาน โดยมีวงจรตัวต้านทานเพิ่มกระแสไฟฟ้าภายใน วงจรดีฟลิปฟลอปต่อสัญญาณออกที่ขา Q และมีวงจรอินเวอร์เตอร์ ต่อกับขาไบแอสของเฟต

การทำงานเป็นเอาต์พุตของพอร์ต 2 จะมีลักษณะคล้ายกันกับพอร์ต 0 เมื่อต้องการส่งข้อมูล 0 ออกไปแสดงผลภายนอก ก็สามารถส่งข้อมูลไปออกได้เลยโดยไม่ต้องกำหนดค่าใดๆ ข้อมูลก็จะผ่านวงจรดีฟลิปฟลอปออกไปทางขา Q โดยจะมีสถานะลอจิกเป็น "1" เข้าไปไบแอสให้เฟตทำงาน (ON) กระแสไฟฟ้าจากแหล่งจ่ายไฟก็จะไหลผ่านลงกราวด์ (Ground) ผลก็คือ จะได้ค่าสถานะทางเอาต์พุตเป็น "0" ในกรณีที่ส่งค่าข้อมูลเป็น 1 ก็จะมีลักษณะการทำงานเหมือนกัน ต่างที่สถานะของข้อมูล ดังรูปที่ 2.7-2.8



รูปที่ 2.7 การทำงานของพอร์ต 2 เมื่อส่งข้อมูลออกที่มีค่าเป็น 0 [1]

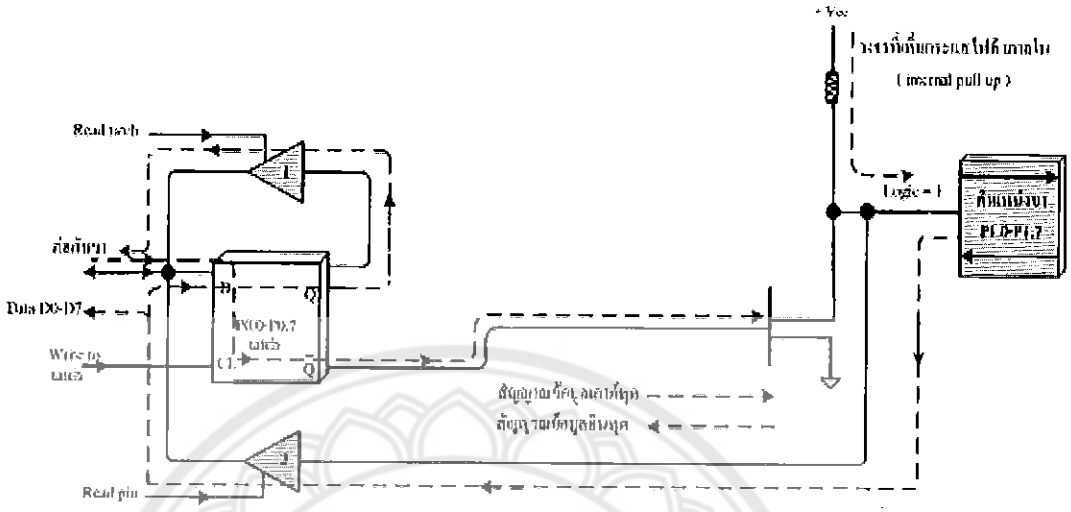


รูปที่ 2.8 การทำงานของพอร์ต 2 เมื่อส่งข้อมูลออกที่มีค่าเป็น 1 [1]

กรณีที่ต้องการให้แต่ละบิตเป็นอินพุต โดยการส่งข้อมูลที่มีค่าเป็น 1 ไปยังแต่ละบิต ทำให้เฟตไม่ทำงาน (Off) และมีค่าความต้านทานสูงมาก ซึ่งสามารถใช้อ่านค่าข้อมูลอินพุตจาก ภายนอกได้โดยตรง

2.1.8 ตำแหน่งของพอร์ต 1

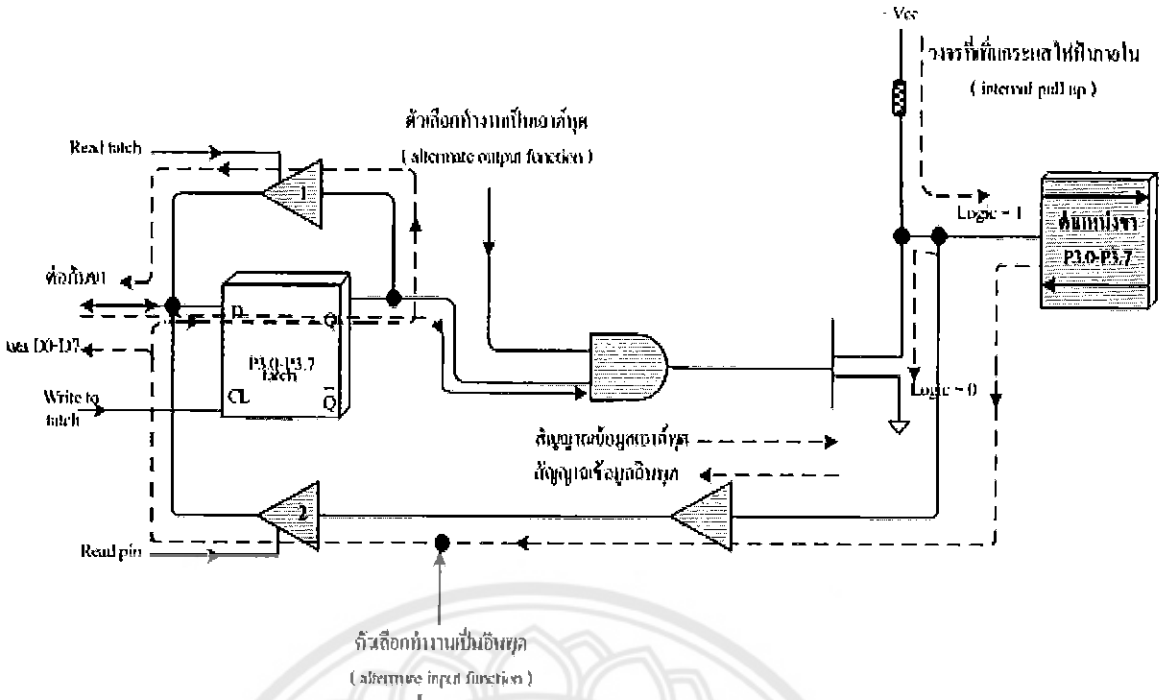
พอร์ต 1 (P1.0-P1.7) เป็นวงจรพื้นฐานที่มีการทำงานเป็นอินพุต/เอาต์พุตทั่วไป และมีส่วนประกอบของวงจรน้อยกว่าพอร์ตอื่นๆ การทำงานเป็นเอาต์พุตและอินพุตก็จะมีลักษณะเหมือนกับพอร์ตอื่นๆ ดังรูปที่ 2.9



รูปที่ 2.9 การทำงานของพอร์ต 1 เมื่อใช้เป็นเอาต์พุตและอินพุต [1]

2.1.9 ตำแหน่งของพอร์ต 3

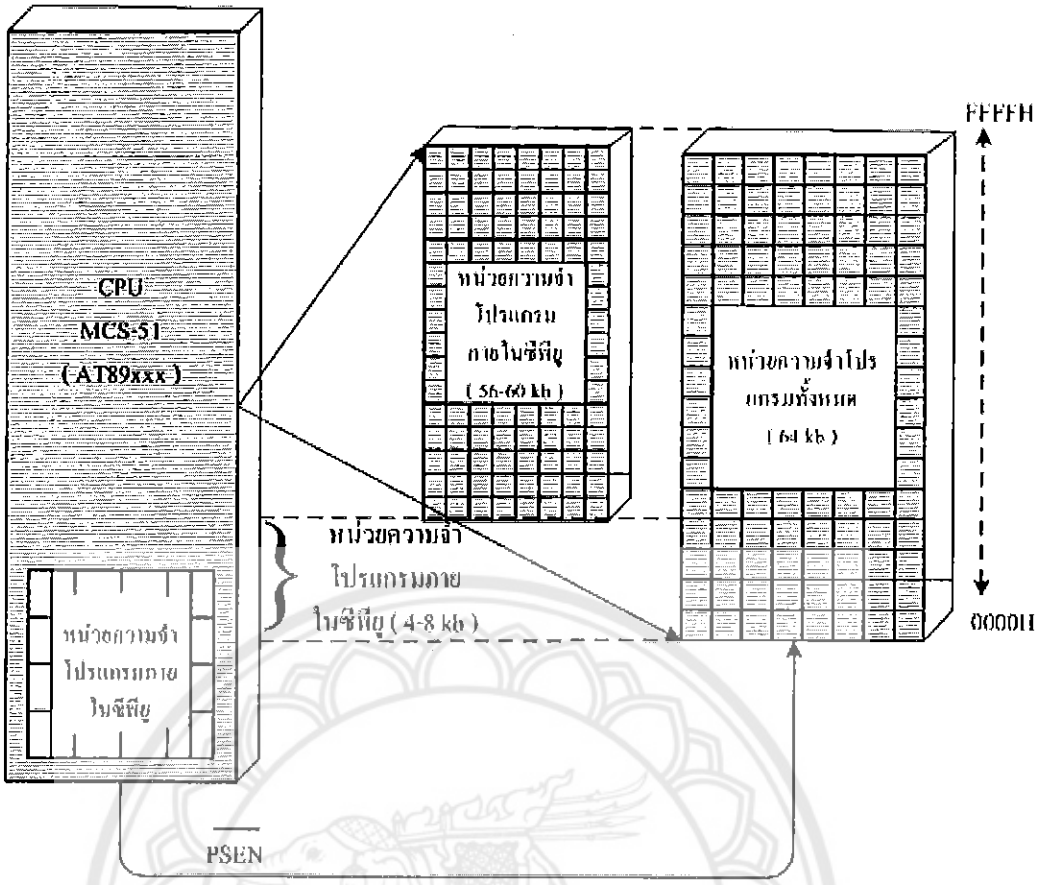
พอร์ต 3 (P3.0-P3.7) นอกจากใช้เป็นพอร์ตเอาต์พุต/อินพุตแล้ว จะมีการใช้งานเหมือนกับport อื่นๆ และยังทำหน้าที่พิเศษอีกหลายอย่าง เช่น เป็นพอร์ตในการสื่อสารข้อมูล พอร์ตควบคุมการติดต่อหน่วยความจำกับอุปกรณ์ภายนอกและวงจรนับเวลา/ตั้งเวลา พร้อมทั้งเป็นอินพุตการทำงานแบบอินเตอร์รัปของไมโครคอนโทรลเลอร์ ดังนั้นจึงมีสัญญาณควบคุมเพิ่มขึ้นทั้งเอาต์พุตและอินพุต ดังรูปที่ 2.10



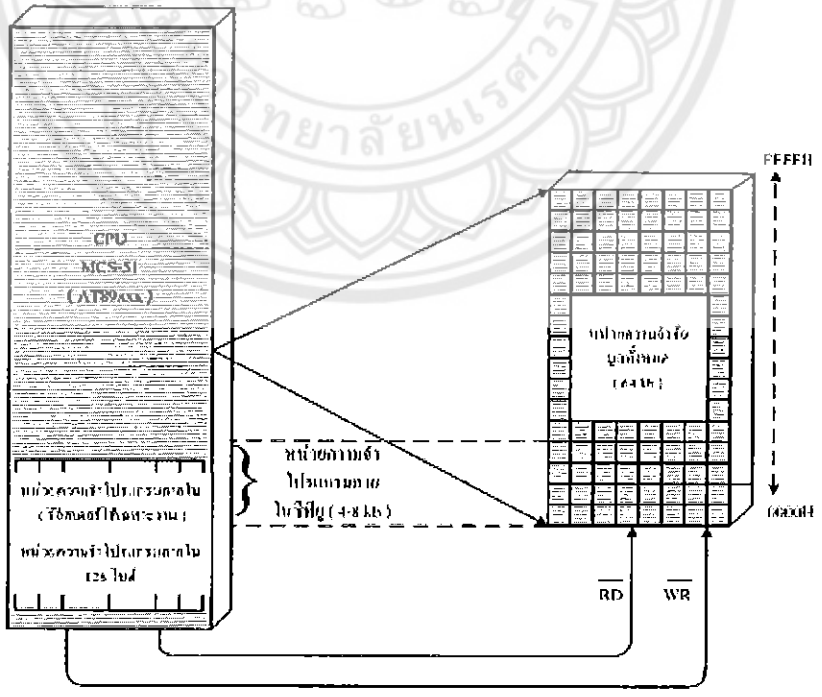
รูปที่ 2.10 การทำงานของพอร์ต 3 [1]

2.1.10 หน่วยความจำของไมโครคอนโทรลเลอร์ MCS-51

ไมโครคอนโทรเลอร์ MCS-51 ได้ออกแบบการจักการหน่วยความจำแต่ละประเภทแยกจากกันและกำหนดการทำงานเป็นแบบเฉพาะ คือ หน่วยความจำโปรแกรม "(program memory หรือ code memory)" และหน่วยความจำชั่วคราว (RAM) เรียกว่า " หน่วยความจำข้อมูล (data memary) ซึ่งมีขนาดความจุ 64 กิโลไบต์เท่ากัน แต่จะถูกแยกการทำงานโดยคำสั่งทางซอฟแวร์และโครงสร้างทางฮาร์ดแวร์ ดังรูปที่ 2.11-2.12



รูปที่ 2.11 การจัดการหน่วยความจำโปรแกรม (program memory) [1]

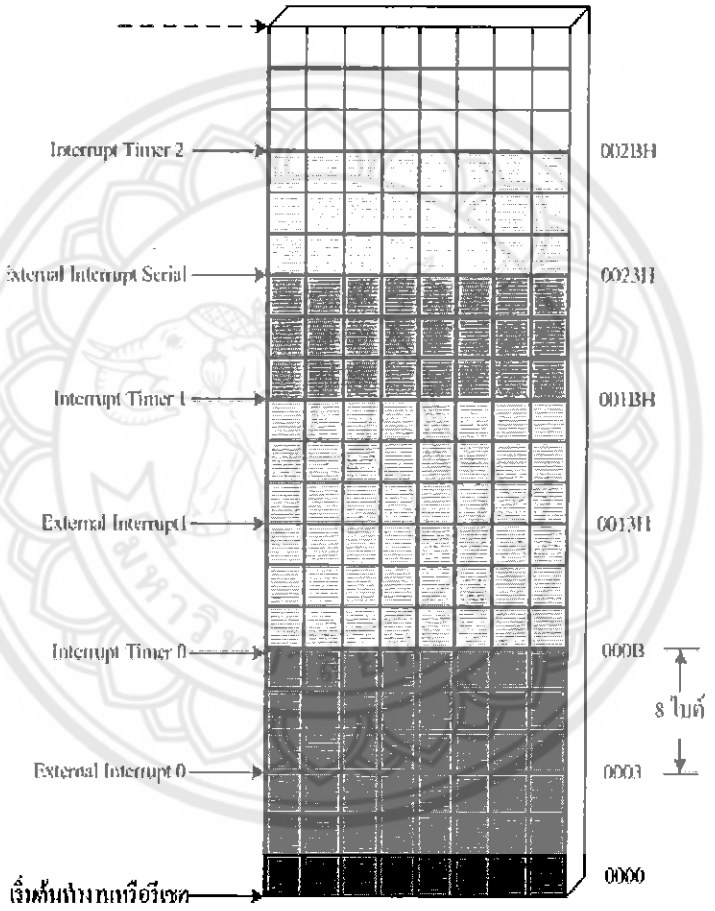


รูปที่ 2.12 การจัดการหน่วยความจำข้อมูล (data memory) [1]

2.1.11 หน่วยความจำโปรแกรม (Program Memory)

ไมโครคอนโทรลเลอร์ MCS-51 ที่เริ่มต้นทำงานใหม่ หรือเมื่อรีเซตใหม่จะเริ่มทำงานที่ตำแหน่ง 0000 ทุกครั้ง โดยจะอ่านข้อมูลที่ตำแหน่ง 0000 แปลความหมายและปฏิบัติตามคำสั่ง ซึ่งเป็นตำแหน่งหน่วยความจำโปรแกรม (program memory) ที่ใช้เก็บประแกรมเบื้องต้นในการทำงานของไมโครคอนโทรลเลอร์ (monitor program) หรือระบบปฏิบัติการ (BIOS) ที่อาจอยู่ภายในหรือภายนอกไมโครคอนโทรลเลอร์ก็ได้

หลังจากโปรแกรมเริ่มทำงานจะมีการกำหนดตำแหน่งหน่วยความจำโปรแกรมเพื่อรับการอินเตอร์รัปต์ซึ่งมี 6 ประเภท แต่ละประเภทมีขนาด 8 ไบต์ ดังรูปที่ 2.13



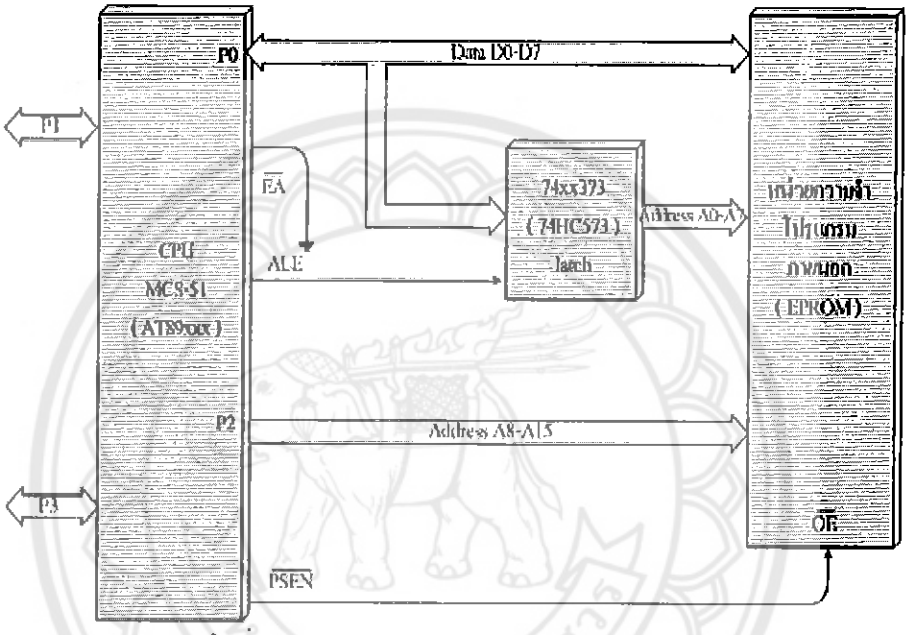
รูปที่ 2.13 หน่วยความจำโปรแกรมที่ตำแหน่งเริ่มต้นการทำงานและการอินเตอร์รัปต์ [1]

2.1.12 หน่วยความจำโปรแกรมภายนอก

งานควบคุมบางอย่างยังต้องใช้หน่วยความจำโปรแกรมเป็นจำนวนมาก ซึ่งหน่วยความจำโปรแกรมภายในอาจไม่เพียงพอ ก็สามารถใช้ไอซีหน่วยความจำมาต่อขยายได้ ซึ่งจะเริ่มจากตำแหน่งสุดท้ายของหน่วยความจำโปรแกรมภายใน เช่น ถ้ามีหน่วยความจำโปรแกรมภายในขนาด

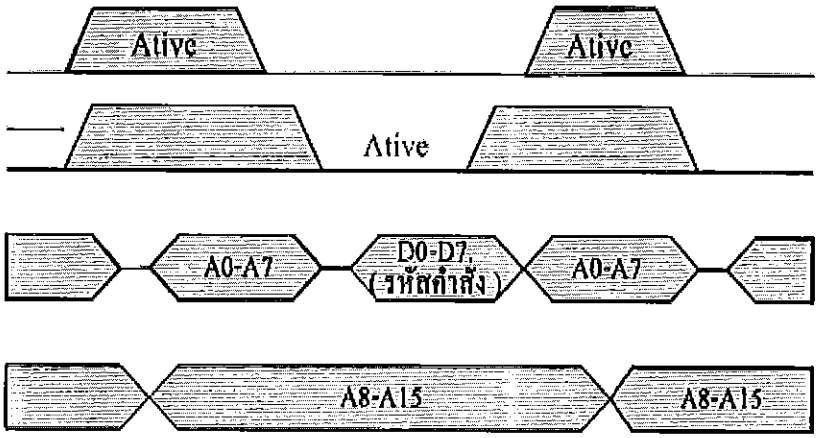
4 กิโลไบต์ มีตำแหน่งหน่วยความจำข้อมูลอยู่ระหว่าง 0000-0FFFH การต่อหน่วยความจำภายนอกต้องเริ่มต้นที่ตำแหน่ง 1000H-FFFH

การที่จะต่อวงจรหน่วยความจำโปรแกรมภายนอกจะใช้พอร์ต 0 เป็นสายสัญญาณของข้อมูล (D0-D7) และสายสัญญาณกำหนดตำแหน่งหน่วยความจำ (A0-A7)โดยใช้เกตทีทีแอล 74XX373 (74HC573) ทำหน้าที่แยกสัญญาณการทำงานทั้งสองออกจากกันด้วยสัญญาณควบคุม ALE และใช้พอร์ต 2 ทำหน้าที่เป็นสัญญาณกำหนดตำแหน่งความจำ (A8-A15) ให้ครบ 16 บิต โดยใช้ขาควบคุม $\overline{\text{PSEN}}$ ทำหน้าที่ควบคุมการทำงานของไอซีหน่วยความจำ ดังรูปที่ 2.14



รูปที่ 2.14 วงจรหน่วยความจำโปรแกรมภายนอก [1]

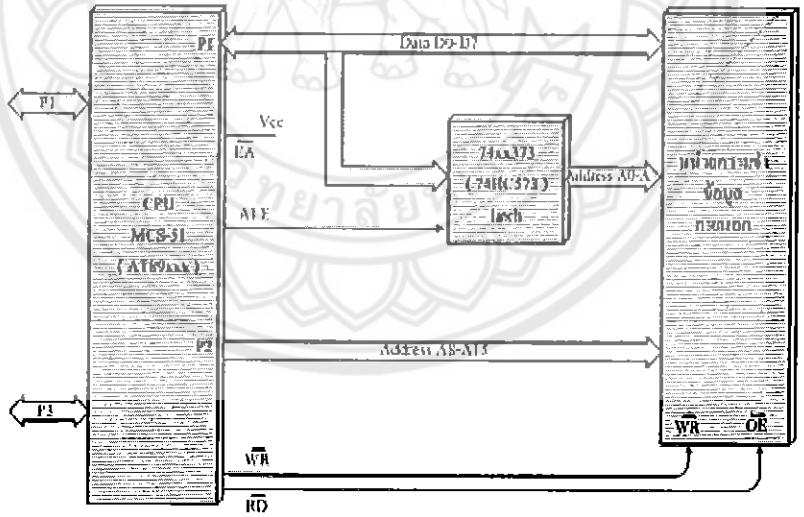
การอ่านค่าจากหน่วยความจำโปรแกรมภายนอก เริ่มไมโครคอนโทรลเลอร์ได้รับคำสั่งให้อ่านค่า สัญญาณ ALE จะมีสถานะเป็น "1" เพื่อให้สัญญาณกำหนดตำแหน่งพอร์ต 0 และพอร์ต 2 ทำงานและส่งค่าออกมา ในขณะที่ขาสัญญาณ $\overline{\text{PSEN}}$ ก็จะทำงานและมีสภาวะลอจิกเป็น "0" จากนั้นไมโครคอนโทรลเลอร์จะอ่านค่ารหัสคำสั่งผ่านมายังพอร์ต 0 (D0-D7) จะเห็นว่าที่พอร์ต 0 มีช่วงคาบเวลาของสัญญาณกำหนดตำแหน่ง (A0-A7) ในการอ่านไม่มาก เพราะมีสัญญาณข้อมูล (D0-D7) ใช้เส้นทางร่วมกัน ส่วนพอร์ต 2 ใช้เป็นสัญญาณกำหนดตำแหน่ง (A8-A15) ซึ่งมีช่วงคาบเวลายาวกว่าพอร์ต 0 ดังรูปที่ 2.15



รูปที่ 2.15 การอ่านข้อมูลหน่วยความจำโปรแกรมภายนอก [1]

2.1.13 หน่วยความจำข้อมูล (data memory)

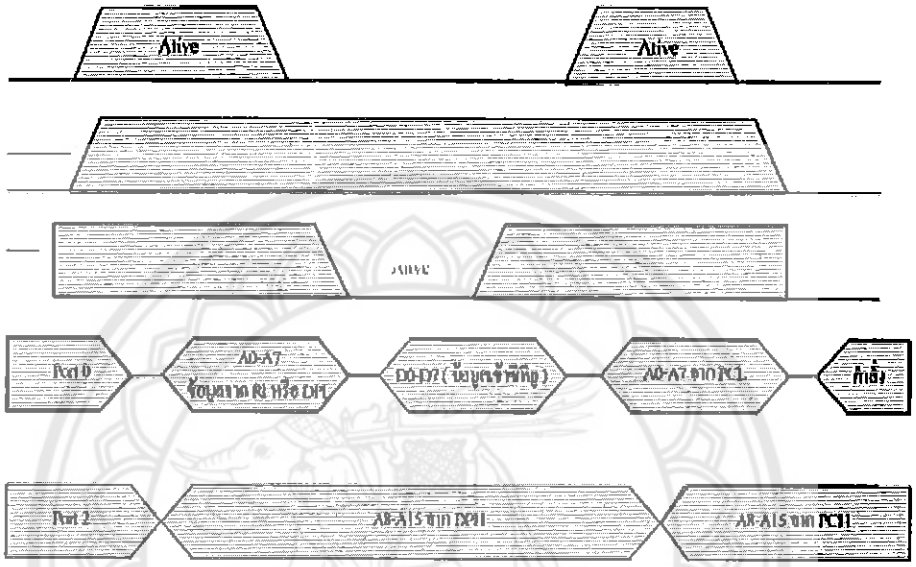
การใช้งานจะมีอยู่ 2 แบบ คือ หน่วยความจำข้อมูลที่อยู่ภายในกับหน่วยความจำข้อมูลที่อยู่ภายนอกไมโครคอนโทรลเลอร์ สำหรับไมโครคอนโทรลเลอร์ตระกูล MCS-51 (AT89XXX) ขนาดของหน่วยความจำทั้งสองรวมกันจะไม่เกิด 64 กิโลไบต์ การติดต่อข้อมูลเพื่อทำการอ่าน/เขียนข้อมูลทำได้โดยใช้คำสั่ง MOVX เท่านั้น การต่อวงจรภายนอกจะใช้สัญญาณควบคุมการอ่านและการเขียนข้อมูล $\overline{RD}$ และ $\overline{WR}$ ดังรูปที่ 2.16



รูปที่ 2.16 วงจรหน่วยความจำข้อมูลภายนอก [1]

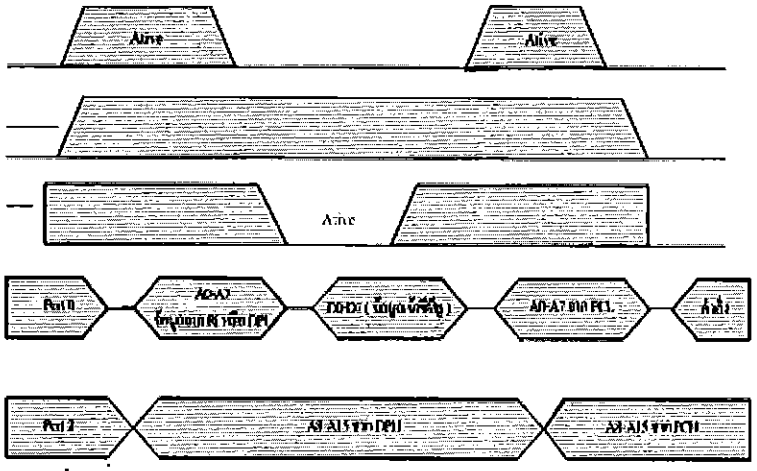
การอ่านหน่วยความจำข้อมูลจากหน่วยความจำภายนอกจะต้องใช้สัญญาณควบคุม ALE ซึ่งทำงานที่สภาวะลอจิก "1" ทำหน้าที่เลือกสัญญาณข้อมูล (D0-D7) กับตำแหน่งหน่วยความจำ (A0-A7) ของพอร์ต 0 และสัญญาณควบคุมการอ่าน ($\overline{RD}$) จะทำงานที่สภาวะลอจิกเป็น "0"

เมื่อสัญญาณ ALE มีสถานะลอจิกเป็น "1" ทำให้สัญญาณกำหนดตำแหน่ง (A0-A7) ผ่านไอซี 74XX373 ไปกำหนดตำแหน่งที่หน่วยความจำ พอร์ต 2 ก็จะส่งสัญญาณออกไปกำหนดตำแหน่ง (A8-A15) ออกไปพร้อมกัน แต่มีคาบเวลามากกว่า จากนั้นสัญญาณการการอ่านข้อมูล ($\overline{RD}$) จะมีค่าสถานะลอจิกเป็น "0" ไมโครคอนโทรลเลอร์ทำการอ่านข้อมูลจากหน่วยความจำที่กำหนดตำแหน่ง (A0-A15) เข้ามาเก็บในรีจีสเตอร์ต่างๆ จากนั้นโปรแกรมนับตำแหน่ง (PC) ก็จะส่งค่าไปตำแหน่งถัดไป ดังรูปที่ 2.17



รูปที่ 2.17 การอ่านข้อมูลจากหน่วยความจำข้อมูลภายนอก [1]

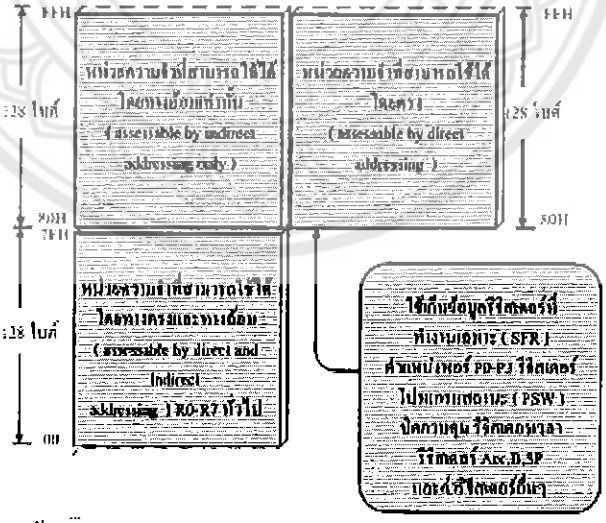
การเขียนข้อมูลไปยังหน่วยความจำข้อมูลภายนอกจะมีลักษณะการทำงานคล้ายกับการอ่านข้อมูล แต่มีข้อแตกต่างคือ ใช้สัญญาณควบคุมในการเขียนข้อมูล ($\overline{WR}$) ซึ่งช่วงเวลาในการเขียนข้อมูลจะใช้เวลามากกว่าการอ่านข้อมูล เพราะต้องรอการทำงานของไอซีหน่วยความจำที่ทำงานช้ากว่าไมโครคอนโทรลเลอร์ ดังนั้นสัญญาณการเขียน ($\overline{WR}$) กับสายสัญญาณข้อมูล (D0-D7) จึงมีช่วงเวลาทำงานมากกว่า ดังรูปที่ 2.18



รูปที่ 2.18 การเขียนข้อมูลไปยังหน่วยความจำข้อมูลภายนอก [1]

2.1.14 หน่วยความจำข้อมูลภายใน

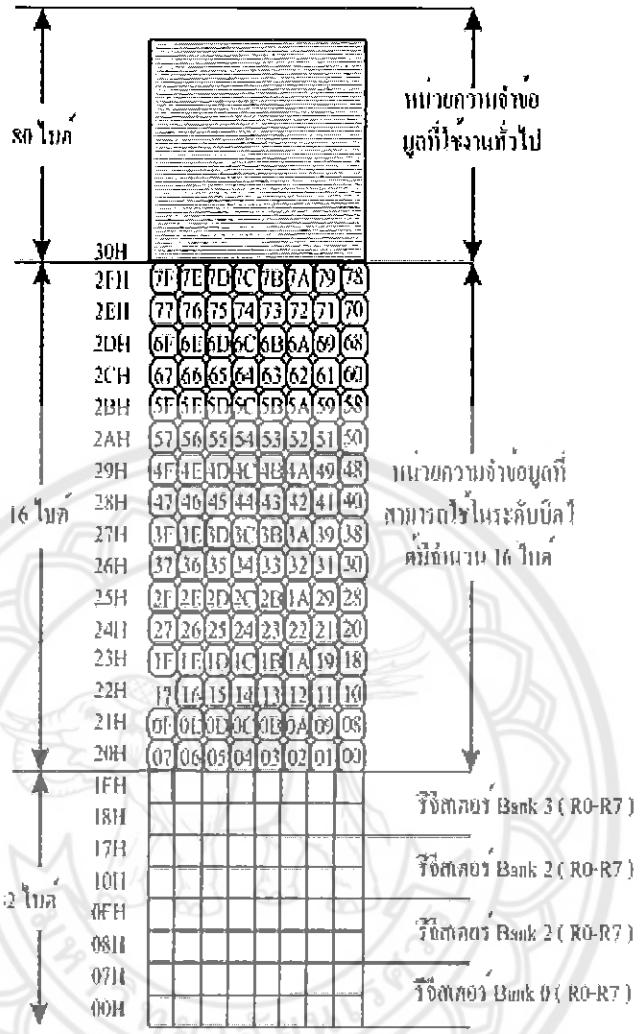
โครงสร้างภายในของไมโครคอนโทรลเลอร์ตระกูล MCS-51 (AT89XXX) จะมีหน่วยความจำข้อมูล (RAM) ขนาด 128 – 256 ไบต์ ขึ้นอยู่กับรุ่นของไมโครคอนโทรเลอร์ การใช้งานจะมีการแบ่งออกเป็นส่วนใหญ่ๆ คือหน่วยความจำตั้งแต่ตำแหน่ง 00-7FH (ส่วน = 128 ไบต์) และหน่วยความจำที่ตำแหน่ง 80H-FFH นอกจากนี้ยังได้มีการจัดระบบหน่วยความจำให้สามรถทำงานได้ 2 สภาวะ โดยใช้เป็นหน่วยความจำทั่วไปที่ใช้ได้โดยทางอ้อม และใช้เป็นรีจีสเตอร์ที่ทำงานเฉพาะ (Specical Function Register = SFR) อีกส่วนหนึ่ง ซึ่งดูเสมือนว่าหน่วยความจำภายในส่วนนี้จะมีขนาดทั้งหมด 384 กิโลไบต์ ดังรูปที่ 2.19



รูปที่ 2.19 การจัดพื้นที่งานของหน่วยความจำข้อมูลภายในไมโครคอนโทรลเลอร์ [1]

การจัดพื้นที่ใช้งานของหน่วยความจำข้อมูลที่อยู่ภายในถือว่าเป็นพื้นฐานสำคัญของการเรียนการเขียนโปรแกรมภาษาแอสเซมบลี ในตำแหน่ง 128 ไบต์ส่วนล่าง (00-7FH) ได้มีการจัดแบ่งหน้าที่แต่ละ

ตำแหน่งออกเป็นส่วนย่อยๆ ซึ่งแต่ละส่วนได้กำหนดหน้าที่ให้ทำงานเฉพาะ เพื่อให้การเขียนโปรแกรมสามารถเรียกใช้ได้ง่ายขึ้น ดังรูปที่ 2.20



รูปที่ 2.20 พื้นที่ใช้งานของหน่วยความจำภายในที่ 128 ไบต์ส่วนล่าง [1]

พื้นที่หน่วยความจำ 00-1FH มีขนาด 32 ไบต์ ได้กำหนดเป็นค่ารีจีสเตอร์ R0-R7 ขนาด 8 บิต จำนวน 4 ชุด (Bank 0 – Bank 3) เป็นรีจีสเตอร์ที่ใช้งานทั่วไป การเรียกใช้งานแต่ละชุดทำได้โดยกำหนดบิตในรีจีสเตอร์ PSW (Program Status Word) การรีเซตหรือเริ่มต้นทำงานใหญ่ทุกครั้งจะกำหนดไว้ที่ตำแหน่งรีจีสเตอร์ชุดที่ 1 (Bank 0)

พื้นที่หน่วยความจำ 20H – 2FH มีขนาด 16 ไบต์ ออกแบบมาให้สามารถใช้ได้ถึงระดับบิต (bit addressable) ซึ่งเป็นหน่วยที่เล็กที่สุดของข้อมูล สามารถกระทำทางลอจิก (SET , CLEAR , AND และ OR) ได้ ทำให้ควบคุมการทำงานได้ละเอียดมากยิ่งขึ้นและในส่วนที่เหลืออีก 80 ไบต์นั้น (30H-7FH) ก็สามารถใช้งานได้ทั่วไป แต่ได้เป็นระดับไบต์ ซึ่งส่วนหนึ่งจะสำรองไว้เก็บข้อมูล สแตก (Stack Pointer = SP) ที่ใช้เก็บข้อมูลตำแหน่งที่กระโดดไปทำงานในส่วนของโปรแกรมย่อย

2.1.15 หน่วยความจำรีจิสเตอร์

หน่วยความจำรีจีสเตอร์ (80H-FFH) ซึ่งเป็นส่วนที่เข้าถึงข้อมูลได้โดยตรง และเป็นส่วนสำคัญเพราะทำหน้าที่เป็นรีจีสเตอร์ที่ใช้งานเฉพาะเป็นการทำงานหลักของไมโครคอนโทรลเลอร์ เช่น การคำนวณการตั้งเวลานับเวลา การอินเตอร์รัปต์ การส่งข้อมูลแบบอนุกรม และ รีจีสเตอร์อื่นๆ ที่จำเป็น ดังรูปที่ 2.21

รูปที่ 2.21 พื้นที่งานของหน่วยความจำภายในส่วนบน (80H-FFH) [2]

การกำหนดพื้นที่หน่วยความจำในที่ใช้เป็นตำแหน่งหน่วยความจำและใช้ชื่อของรีจิสเตอร์ ในไมโครคอนโทรลเลอร์ตระกูล MCS-51 (AT89XXX) ได้ออกแบบค่ารีจีสเตอร์บางตัวให้สามารถเข้าถึงได้ในระดับบิตและไบต์ ดังตารางที่ 2.5

4900164

ตารางที่ 2.5 ชื่อและตำแหน่งรีจิสเตอร์ในไมโครคอนโทรลเลอร์ตระกูล MCS-51 [1]

สัญลักษณ์	ชื่อ	ตำแหน่ง (address)
*Acc	Accumulator	0EOH
*B	B Register	OF0H
*PSW	Program Status Word	ODOH
SP	Stack Pointer	81H
DPTR	Data Pointer 2 ไบต์ DPH,DPL	
DPL	ไบต์ต่ำ (Low byte)	82H
DPH	ไบต์ต่ำ (High byte)	83H
*P0	Port 0	80H
*P1	Port 1	90H
*P2	Port 2	0A0H
*P3	Port 3	0B0H
*IP	Interrupt Priority Control	0B8H
*IE	Interrupt Enable Control	0A8H
TMOD	Timer/Counter Mode Control	89H
*TCON	Timer/Counter Control	88H
*+T2CON	Timer/Counter 2 Control	0C8H
TH0	Timer/Counter 0 High byte	8CH
TL0	Timer/Counter 0 Low byte	8AH
TH1	Timer/Counter 1 High byte	8DH
TL1	Timer/Counter 1 Low byte	8BH
+TH2	Timer/Counter 2 High byte	0CDH
+TL2	Timer/Counter 2 High byte	0CCH
+RCAP2H	T/C 2 Capture Reg. High byte	0CBH
+RCAP2L	T/C 2 Capture Reg. Low byte	0CAH
*SCON	Serial Control	98H
SBUF	Serial Data Buffer	99H
PCON	Power Control	87H

หมายเหตุ * เป็นรีจีสเตอร์ที่สามารถเข้าถึงระดับบิต + มีเฉพาะรุ่น (80xx52) (AT89xx52)

2.1.16 การทำงานของรีจีสเตอร์

รีจีสเตอร์ A (Accumulator = Acc)

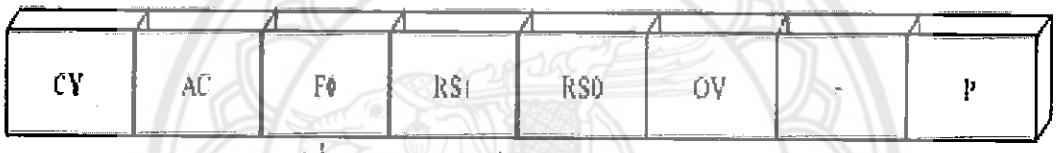
เป็นรีจีสเตอร์ขนาด 8 บิต สามารถเข้าถึงได้ระดับบิต ทำหน้าที่หลักในการคำนวณทางคณิตศาสตร์และลอจิก เก็บผลลัพธ์จากการประมวล และสามารถเก็บข้อมูลขนาด 8 บิตได้ อยู่ที่ตำแหน่งหน่วยความจำ E0H

รีจีสเตอร์ B (B Register)

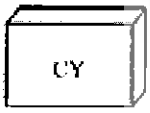
เป็นรีจีสเตอร์ขนาด 8 บิต ใช้เก็บข้อมูลทั่วไป และยังทำหน้าที่พิเศษคือ เก็บข้อมูลตัวกระทำการคูณหรือการหาร เช่น รีจีสเตอร์ A เก็บค่าตัวตั้ง และรีจีสเตอร์ B เป็นตัวคูณหรือหาร สามารถเข้าถึงได้ระดับ อยู่ที่ตำแหน่งหน่วยความจำ 0F0H

รีจีสเตอร์ PSW (Program Status Word)

ทำหน้าที่เก็บสภาวะการงานของคำสั่ง มีขนาด 7 บิต อยู่ที่ตำแหน่งหน่วยความจำ 0D0H ซึ่งมีการเก็บสภาวะการทำงานของแต่ละบิตดังต่อไปนี้



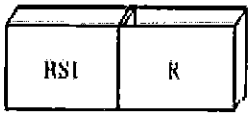
รูปที่ 2.22 รีจีสเตอร์ PSW (Program Status Word) [3]



(Carry Flag) ทำหน้าที่เก็บค่าตัวทดเมื่อมีการกระทำทางคณิตศาสตร์และลอจิก ถ้าผลลัพธ์จากการประมวลผลที่ได้มีค่าเกิน 8 บิต (0FFH) บิตที่เกินก็จะเก็บไว้ที่บิต CY ซึ่งเป็นบิตทด

(Auxiliary Carry Flag) เป็นตัวช่วยเมื่อมีการคำนวณ และเกิดการยืมค่าหรือเกิดทดขึ้นระหว่างบิตที่ 3 กับบิตที่ 4 ของค่ารีจีสเตอร์ บิตนี้จะมีค่าลอจิกเป็น "1" ส่วนมากใช้กับระบบเลขฐานสอง (Binary Code Decimal)

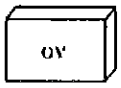
(Flag 0) ทำหน้าที่เป็นตัวเก็บสถานะทั่วไป ถ้ามีการตั้งค่าสถานะนี้ไว้ เมื่อมีการกระทำตามคำสั่งที่มีผลต่อสถานะ (Flag) ก็จะไม่มีผลกระทบต่อค่าที่กำหนดไว้



(Register Select 1,0) ทำหน้าที่เลือกใช้ตำแหน่งของรีจีสเตอร์ R0-R7 มีอยู่ 4 ตำแหน่ง ตำแหน่งละ 8 ตัว (R0-R7) สามารถเลือกใช้ได้โดยการตั้งค่าบิตดังตารางที่ 10.13

ตารางที่ 2.6 ตำแหน่งรีจีสเตอร์ R0-R7 [3]

RS1	RS1	ตำแหน่ง R0-R7
0	0	Bank 0 ตำแหน่ง 00H-07H
0	1	Bank 1 ตำแหน่ง 08H-0FH
0	0	Bank 2 ตำแหน่ง 10H-17H
0	0	Bank 3 ตำแหน่ง 18H-1FH



(Overflow) เป็นค่าแสดงสถานะเมื่อมีการคำนวณหรือการกระทำทางลอจิก ผลลัพธ์ที่ได้จะมีค่าเกินกว่าที่ตำแหน่งหน่วยความจำจะรับได้ ทำให้สถานะนี้เป็น "1" และยังสามารถใช้เป็นค่าแสดงผลลัพธ์ที่เป็นลบ

บิตนี้ได้ใช้งาน ผู้ใช้สามารถนำไปใช้งานได้

(Parity Flag) ใช้ตรวจสอบค่าที่เก็บอยู่ในรีจีสเตอร์ A แต่ละบิตมีค่าบิตเป็น 1 รวมกันเป็นจำนวนคู่หรือคี่ ถ้ามีจำนวนเป็นคู่ ค่าสถานะนี้จะเป็น "1" แต่ถ้าเป็นจำนวนคี่จะมีค่าสถานะเป็น "0"

รีจีสเตอร์สแตก (SP = Stack Pointer)

เป็นรีจีสเตอร์ขนาด 8 บิต มีตำแหน่งอยู่ที่ 81H ใช้เก็บตำแหน่งตัวชี้ค่าสแตก (เข้าที่หลังออกก่อน) และใช้เก็บค่าตำแหน่งของโปรแกรมหลัก เมื่อไมโครคอนโทรเลอร์กระโดดไปทำงานโปรแกรมย่อยเสร็จ ไมโครคอนโทรลเลอร์จะกลับมาทำงานที่โปรแกรมหลักอีกที่ ก่อนที่จะกระโดดไปไมโครคอนโทรลเลอร์จะเก็บค่าตำแหน่งไว้ที่สแตกก่อน ในกรณีการเขียนโปรแกรมย่อยหลายๆ โปรแกรมค่าสแตกก็จะมีการเก็บค่าตำแหน่งมากขึ้น พื้นที่ที่ต้องใช้งานก็เพิ่มมากขึ้น ค่ารีจีสเตอร์สแตกสามารถกำหนดตำแหน่งใหม่ได้แต่ถ้าไม่กำหนดตำแหน่งไมโครคอนโทรลเลอร์ตระกูล MCS-51 จะกำหนดตำแหน่งไว้ที่ 07H

รีจีสเตอร์เป็นตัวนับขั้นตอนในโปรแกรม (PC = Program Counter)

เป็นรีจีสเตอร์ขนาด 16 บิต มีหน้าที่บอกตำแหน่งการทำงานของไมโครคอนโทรลเลอร์แต่ละคำสั่งจะมีขนาดความยาวโค้ดคำสั่งต่างกัน ซึ่งมีความสำคัญมากในการตรวจสอบการเขียนของโปรแกรมให้ไมโครคอนโทรลเลอร์ประมวลผล และ ปฏิบัติตามลำดับขั้นตอน

แผนงานที่ได้วางไว้รีจีสเตอร์ตัวนับขั้นตอนในโปรแกรมจะเป็นตัวเก็บค่าตำแหน่งทุกตำแหน่งที่ไมโครคอนโทรลเลอร์ทำงานอยู่ปัจจุบัน

รีจีสเตอร์ตัวชี้ข้อมูล (DPTR = Pointer)

เป็นรีจีสเตอร์ขนาด 8 บิต 2 ชุด ประกอบด้วยรีจีสเตอร์ไบต์ต่ำ (DPH) อยู่ที่ตำแหน่ง 82H และ 83H สามารถใช้ได้ 8 บิต และใช้รวมเป็นขนาด 16 บิต ทำหน้าที่กำหนดตำแหน่งข้อมูลในหน่วยความจำและอุปกรณ์ภายนอกที่ไมโครคอนโทรลเลอร์ต้องการติดต่อ

รีจีสเตอร์พอร์ต (Port Register)

ไมโครคอนโทรลเลอร์ MCS-51 ได้กำหนดให้ติดต่อทุกอย่างกระทำผ่านพอร์ต ซึ่งมีอยู่ 4 พอร์ต มีขนาดพอร์ต 8 บิต คือ พอร์ต 0 (P0 = 80H) พอร์ต 1 (P1 = 90H) พอร์ต 2 (P2 = A0H) และพอร์ต 3 (P3 = B0H) ทุกพอร์ตสามารถใช้อินพุตและเอาต์พุตที่ควบคุมได้ระดับบิต

รีจีสเตอร์บัฟเฟอร์ข้อมูลอนุกรม (SBUF = Serial Data Buffer)

เป็นรีจีสเตอร์ที่ทำหน้าที่รับส่งข้อมูลแบบอนุกรม มีขนาด 8 บิต อยู่ที่ตำแหน่ง 99H ซึ่งปกติแล้ว ใช้การติดต่อข้อมูลอนุกรม มีบัฟเฟอร์ 2 ชุด คือ บัฟเฟอร์สำหรับส่งข้อมูล (transmit buffer register) ผ่านไปยังขา TXD และบัฟเฟอร์สำหรับรับข้อมูล(receive buffer register) ผ่านมายังขา RXD เมื่อต้องการส่งข้อมูลอนุกรมก็สามารถเขียนข้อมูลไปยังรีจีสเตอร์ SBUF และข้อมูลจะถูกส่งไปยังบัฟเฟอร์เพื่อให้ไมโครคอนโทรลเลอร์ส่งข้อมูลออกไปภายนอก

รีจีสเตอร์เกี่ยวกับเวลา (Timer Register)

เป็นรีจีสเตอร์ขนาด 16 บิต แบ่งเป็นรีจีสเตอร์คู่ คือ ไบต์ต่ำและไบต์สูง ใช้เก็บค่าของตัวนับเวลา (counter) ภายในไมโครคอนโทรลเลอร์ เพื่อใช้เป็นฐานเวลา หรือจับเวลานับจำนวนสัญญาณนาฬิกา (pulse) ในไมโครคอนโทรลเลอร์ตระกูล MCS-51 ประกอบด้วย TH0 TH1 , TL0 TL1 หรือ TL2 TH2

รีจีสเตอร์แคปเจอร์ (Capture Register)

เป็นรีจีสเตอร์ขนาด 16 บิต คือ RCAP2L ไบต์ต่ำ และ RCAP2H ไบต์สูง ใช้ร่วมกับ TL2 TH2 ทำหน้าที่ให้ไมโครคอนโทรเลอร์ตรวจจับเวลาการเปลี่ยนแปลงสถานะลอจิกที่ขา T2EX เพื่อใช้วัดคาบเวลา ความถี่ และการเปลี่ยนแปลงสัญญาณนาฬิกาที่ขา T2EX

รีจีสเตอร์ควบคุม (Control Register)

เป็นรีจีสเตอร์ที่ใช้ควบคุมการทำงานส่วนต่างๆ ของไมโครคอนโทรเลอร์ ซึ่งประกอบด้วยรีจีสเตอร์ต่อไปนี้

รีจีสเตอร์ PCON (Power Control Register)

ทำหน้าควบคุมการทำงานของไมโครคอนโทรลเลอร์ โดยการกำหนดให้สัญญาณนาฬิกาหยุดทำงาน ทำให้ส่วนต่างๆ ภายในไมโครคอนโทรลเลอร์หยุดทำงานด้วย ซึ่งเป็นการลดพลังงานเมื่อไม่ต้องการให้ไมโครคอนโทรลเลอร์ทำงาน (sleep mode)

รีจีสเตอร์ SCON (Serial Control Register)

เป็นรีจีสเตอร์ที่ทำหน้าที่ควบคุมวงจรการสื่อสารแบบอนุกรมภายในไมโครคอนโทรลเลอร์

รีจีสเตอร์ TCON , T2CON และ TMOD T2MOD

ใช้ควบคุมการทำงานของวงจรจับเวลา/นับเวลาภายในไมโครคอนโทรลเลอร์

รีจีสเตอร์ IE และ IP (Interrupt Enable Control และInterrupt Priority Control)

เป็นรีจีสเตอร์ที่ควบคุมการอินเตอร์รัปต์ของไมโครคอนโทรลเลอร์มี IE (Interrupt Enable) เป็นรีจีสเตอร์สนองการทำงานของสัญญาณอินเตอร์รัปต์ และรีจีสเตอร์ IP (Priority Interrupt) ทำหน้าที่จัดลับดับความสำคัญของสัญญาณตอบสนองการอินเตอร์รัปต์ของไมโครคอนโทรลเลอร์ของไมโครคอนโทรลเลอร์

ในกรณีเริ่มต้นการทำงานใหม่ของไมโครคอนโทรลเลอร์หรือรีเซตเครื่อง ค่ารีจิสเตอร์ต่างๆ ที่อยู่ภายในไมโครคอนโทรลเลอร์จะถูกเซตค่าใหม่ ดังตารางที่ 2.7

ตารางที่ 2.7 ค่ารีจีสเตอร์ต่างๆ เมื่อเริ่มต้นทำงานของไมโครคอนโทรลเลอร์ตระกูล MCS-51 [1]

สัญลักษณ์	ชื่อ	เลขฐานสอง
*Acc	Accumulator	00000000
*B	B Register	00000000
*PSW	Program Status Word	00000000
SP	Stack Pointer	00000111
DPTR	Data Pointer 2 ไบต์ (DPH , DPL)	
DPL	ไบต์ต่ำ (Low byte)	00000000
DPH	ไบต์สูง (High byte)	00000000
*P0	Port 0	11111111
*P1	Port 1	11111111
*P3	Port 2	11111111
*IP	Port 3	11111111

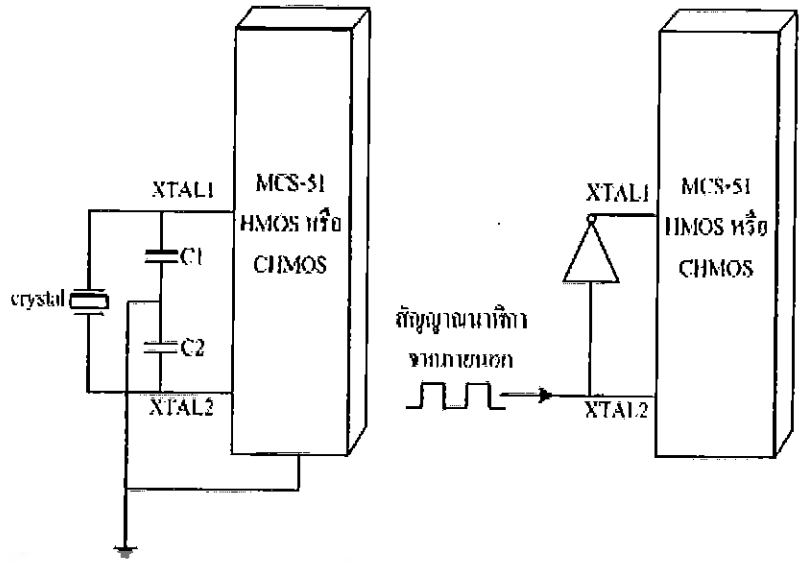
หมายเหตุ * รีจีสเตอร์ที่สามารถเข้าถึงระดับบิต + มีใช้เฉพาะรุ่น 80x52 (AT89xx52) x ไม่ได้กำหนด

ตารางที่ 2.7ค่ารีจีสเตอร์ต่างๆเมื่อเริ่มต้นทำงานของไมโครคอนโทรลเลอร์ตระกูล MCS-51(ต่อ) [1]

*IE	Interrupt Priority Control	8051 XXX00000 , 8052 XX00000
TMOD	Interrupt Enable Control	8051 0XX00000 , 8052XX00000
*TCON	Timer /Counter Mode Control	00000000
*+T2CON	Timer /Counter Control	00000000
TH0	Timer /Counter 2 Control	00000000
TH1	Timer /Counter 1 High byte	00000000
TL	Timer /Counter 1 Low byte	00000000
+TH2	Timer /Counter 2 High byte	00000000
+TL2	Timer /Counter 2 Low byte	00000000
+RCAP2H	T/C 2 Capture Reg. High byte	00000000
+RCAP2L	T/C 2 Capture Reg. Low byte	00000000
*SCON	Serial Control	00000000
SBUF	Serial Data Buffer	ไม่ได้กำหนด
PCON	Power Control	0XXXXXXX

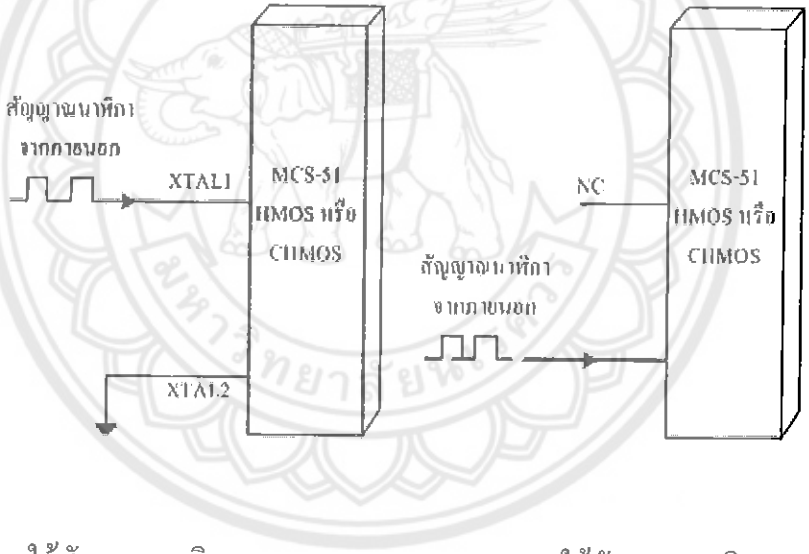
2.1.17 วงจรกำเนิดสัญญาณนาฬิกา

วงจรกำเนิดสัญญาณนาฬิกาของไมโครคอนโทรลเลอร์ตระกูล MCS-51 (AT89XXX) สามารถใช้ต่อได้หลายวิธี เช่น ใช้ผลึกคริสตัล (Crystal) ต่อกับขา XTAL1 และ XTAL พร้อมกับตัวเก็บประจุ หรือใช้สัญญาณนาฬิกาจากภายนอกต่อเข้าที่ XTAL และ XTAL2 ดังรูปที่ 2.22



ก. ใช้วงจรเดียว ข. ใช้สัญญาณนาฬิกาจากภายนอก

รูปที่ 2.23 การต่อสัญญาณนาฬิกาให้กับไมโครคอนโทรลเลอร์ตระกูล MCS-51 [1]



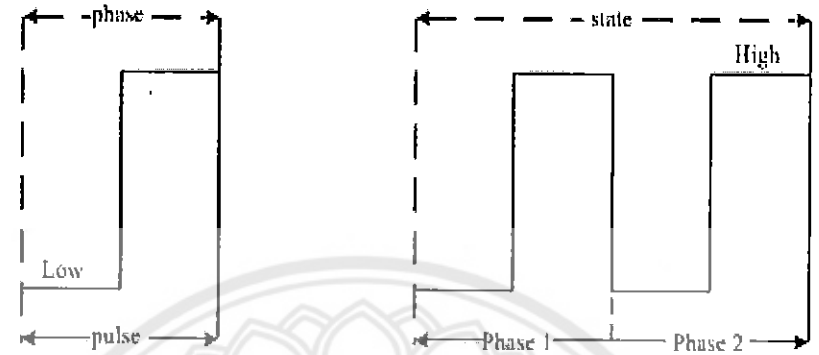
ค. ใช้สัญญาณนาฬิกาจากภายนอก ง. ใช้สัญญาณนาฬิกาจากภายนอก

รูปที่ 2.24 การต่อสัญญาณนาฬิกาให้กับไมโครคอนโทรลเลอร์ตระกูล MCS-51 (ต่อ) [1]

การประมวลผลตามคำสั่งของไมโครคอนโทรลเลอร์มีลำดับการทำงานอยู่ 3 ขั้นตอน คือ การอ่านและตรวจสอบข้อมูล (Fetch) การถอดรหัสข้อมูล (decode) และการปฏิบัติตามคำสั่ง (execute) การกระทำทั้ง 3 ขั้นตอนนี้เรียกว่า 1 รอบการทำงานของคำสั่งไมโครคอนโทรลเลอร์ (machine Cycle) ซึ่งแต่ละขั้นตอนจะใช้สัญญาณนาฬิกาเป็นตัวกำหนดมาตรฐานเวลาในการทำงานของวงจรในไมโครคอนโทรลเลอร์

2.1.18 รูปแบบของสัญญาณนาฬิกา MCS-51

ไมโครคอนโทรลเลอร์ตระกูล MCS-51 ที่ใช้สัญญาณนาฬิกา 1 รอบการทำงานคำสั่งจะมี 6 สภาวะการทำงาน (State) ซึ่งแต่ละสภาวะการทำงานจะใช้ 2 คาบหรือจังหวะ (pnase or pulse) ดังรูปที่ 2.24



ก. คาบเวลา 1 Phase ข. คาบเวลา 1 state

รูปที่ 2.25 สัญญาณนาฬิกาที่ 1 phase และ 1 state [1]

สัญญาณนาฬิกาที่ใช้เป็นฐานเวลาในการทำงานของไมโครคอนโทรลเลอร์ตระกูล MCS-51 ในรูปที่ 11.0 (ก) เป็นคาบเวลาหนึ่งเฟส ซึ่งประกอบด้วยช่วงสัญญาณลอจิก "0" (Low) และลอจิก "1" (High) ซึ่งได้มาจากวงจรกำหนดสัญญาณนาฬิกา ถ้าใช้คริสตัลเป็นตัวกำหนดความถี่ ก็สามารถใช้ค่าความถี่ของคริสตัลมาคำนวณหาค่าเวลาในการทำงานได้ เช่น ไมโครคอนโทรลเลอร์ตระกูล MCS-51 (AT89XXX) สามารถใช้ความถี่สัญญาณนาฬิกาได้ตั้งแต่ 0-24 เมกกะเฮิรตซ์

ถ้าไมโครคอนโทรลเลอร์ใช้สัญญาณนาฬิกาที่มีค่าความถี่ 12 เมกกะเฮิรตซ์ สามารถคำนวณหาค่าคาบเวลาในการทำงานได้ดังต่อไปนี้

สูตรในการคำนวณ

$$T = \frac{1}{F} \qquad \text{สมการที่ 2.1}$$

เมื่อ T = ค่าคาบเวลา: เฟส (phase)

F = ค่าความถี่ของคริสตัล = 12 เมกกะเฮิรตซ์ (MHz)

ดังนั้น $T = \frac{1}{12\text{MHz}}$

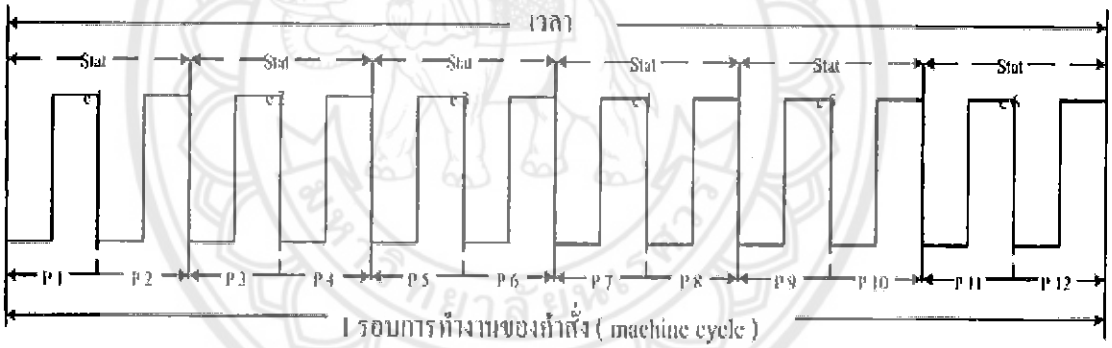
$= 0.08333 \times 10^{-6}$ วินาที

$T = 83.33$ นาโนวินาที

สัญญาณคาบเวลา 1 เฟสใช้เวลา 83.33 นาโนวินาที ในการจัดฐานเวลาการทำงานไมโครคอนโทรลเลอร์ MCS-51 กำหนดให้สัญญาณคาบเวลา 2 เฟส เท่ากับ 1 สภาวะการทำงาน (State) ดังรูปที่ 11.2 (ข) ดังจะได้ค่าเวลาในการทำงานดังนี้

S (state) $= 2 \times T = 2 \times 83.33 = 166.66$ นาโนวินาที

ไมโครคอนโทรลเลอร์ได้กำหนดค่าเวลาในการทำงานของ 1 รอบการทำงานของคำสั่ง (Machine cycle) ให้มีค่าเป็น 6 สภาวะการทำงาน หรือ 12 คาบเวลา ดังรูปที่ 2.25



รูปที่ 2.26 การทำงาน 1 รอบการทำงานของคำสั่ง (Machine cycle) [1]

ดังนั้น ใน 1 รอบการทำงานของคำสั่ง (Machine cycle) จะใช้เวลาการทำงานทั้งหมด

$T = 12 \times 83.33$ นาโนวินาที

$= 999.96$ นาโนวินาที

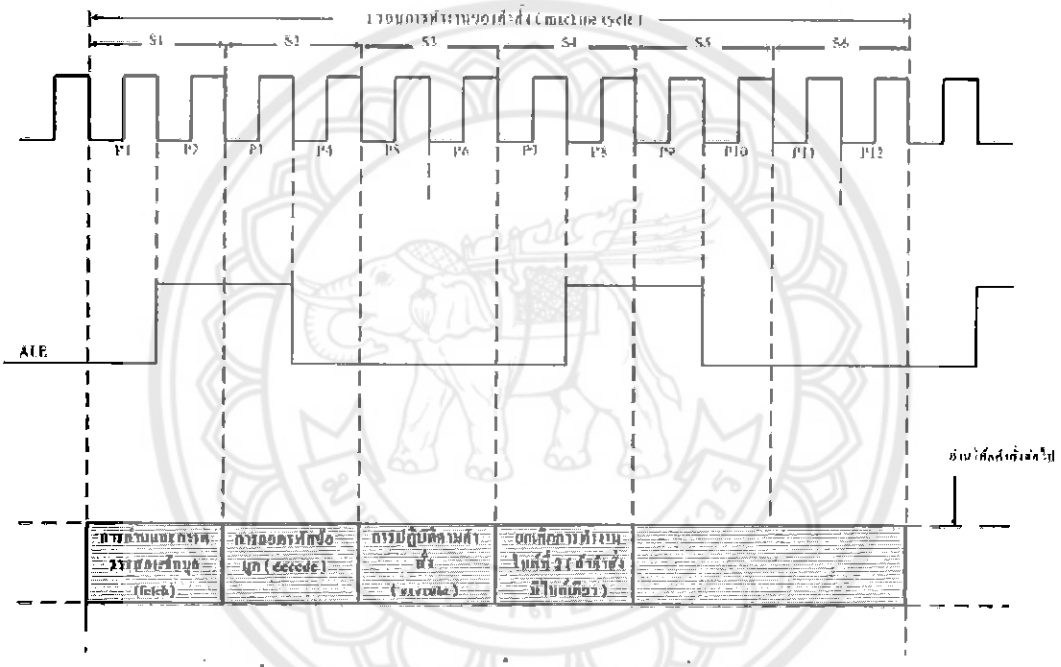
$= 1$ ไมโครวินาที

เวลาที่ใช้ทำงานทั้งหมด $= 1$ ไมโครวินาที

ค่าเวลาที่ได้จากการคำนวณนี้ทำให้ทราบค่าเวลาในการทำงานแต่ละคำสั่งและสามารถนำไปใช้ในการออกแบบการเขียนโปรแกรมควบคุมวงจรอุปกรณ์ภายนอกที่เหมาะกับชนิดของอุปกรณ์ เช่น แอลอีดี (LED) สวิตช์ และรีเลย์ เป็นต้น

2.1.19 คาบเวลาในการทำงานของคำสั่ง

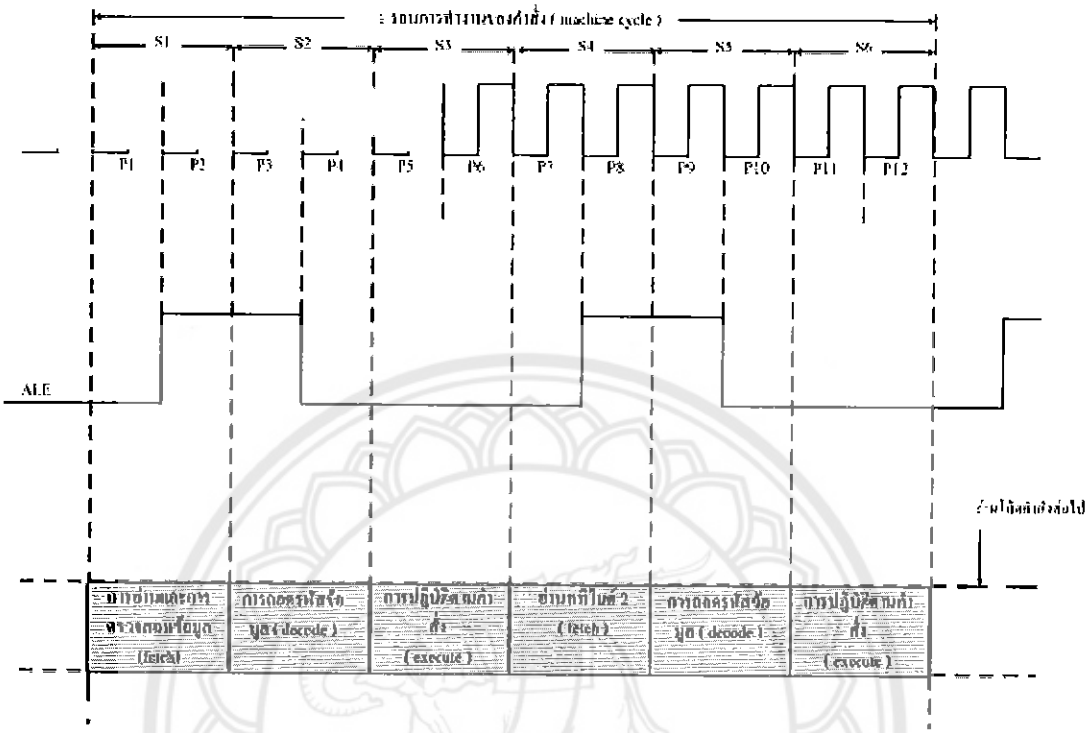
เมื่อได้รับคำสั่งให้ทำงาน หน้าที่ของไมโครคอนโทรลเลอร์จะเริ่มจากการอ่านและตรวจสอบข้อมูล การถอดรหัสข้อมูล และการปฏิบัติตามคำสั่ง ซึ่งแต่ละคำสั่งจะมีคาบเวลาในการทำงานที่แตกต่างกันขึ้นอยู่กับชนิดของคำสั่ง คำสั่งของไมโครคอนโทรลเลอร์ MCS-51 จะมีรอบการทำงานของคำสั่งตั้งแต่ 1-4 คำสั่ง คำสั่งที่ใช้คาบเวลาน้อยที่สุด คือ 1 รอบการทำงาน หรือ 12 คาบเวลา ส่วนมากเป็นคำสั่งในการทำงานเคลื่อนย้ายข้อมูลภายใน ส่วนการเคลื่อนย้ายข้อมูลภายนอกจะใช้ 2 รอบการทำงาน หรือ 24 คาบเวลาและคำสั่งที่ใช้คาบเวลามากที่สุด คือ 48 คาบเวลา เป็นคำสั่งในการคำนวณทางคณิตศาสตร์และลอจิกรูปที่ 2.26 แสดงตัวอย่างการทำงานตามคำสั่ง 1 รอบการทำงาน



รูปที่ 2.27 การทำงาน 1 รอบการทำงานของคำสั่งขนาด 1 ไบต์ [1]

รูปที่ 2.25 เป็นการทำงานของคำสั่งที่ใช้ภาษาแอสเซมบลีของไมโครคอนโทรลเลอร์ตระกูล MCS-51 ซึ่งมีขนาด 1 ไบต์ ใช้เวลา 1 รอบการทำงาน หรือ 12 เฟส ได้แก่ คำสั่ง INC A , INC Rn , DEC A , MOV Rn,A เป็นต้น จะเห็นว่าสภาวะที่ 1 (S1) เป็นเวลาในการอ่านและตรวจสอบข้อมูล สภาวะที่ 2 (S2) เป็นการถอดรหัสคำสั่งให้ไมโครคอนโทรลเลอร์เข้าใจ และสภาวะที่ 3 (S3) จะเป็นการปฏิบัติตามคำสั่ง และ ในสภาวะที่ 4 (S4) จะเริ่มอ่านไบต์ต่อไปถ้าคำสั่งนั้นมีขนาด 2 ไบต์ หลังจากเวลานี้ไปจนถึงสภาวะที่ 6 จะเป็นการสิ้นสุดการทำงานของคำสั่ง

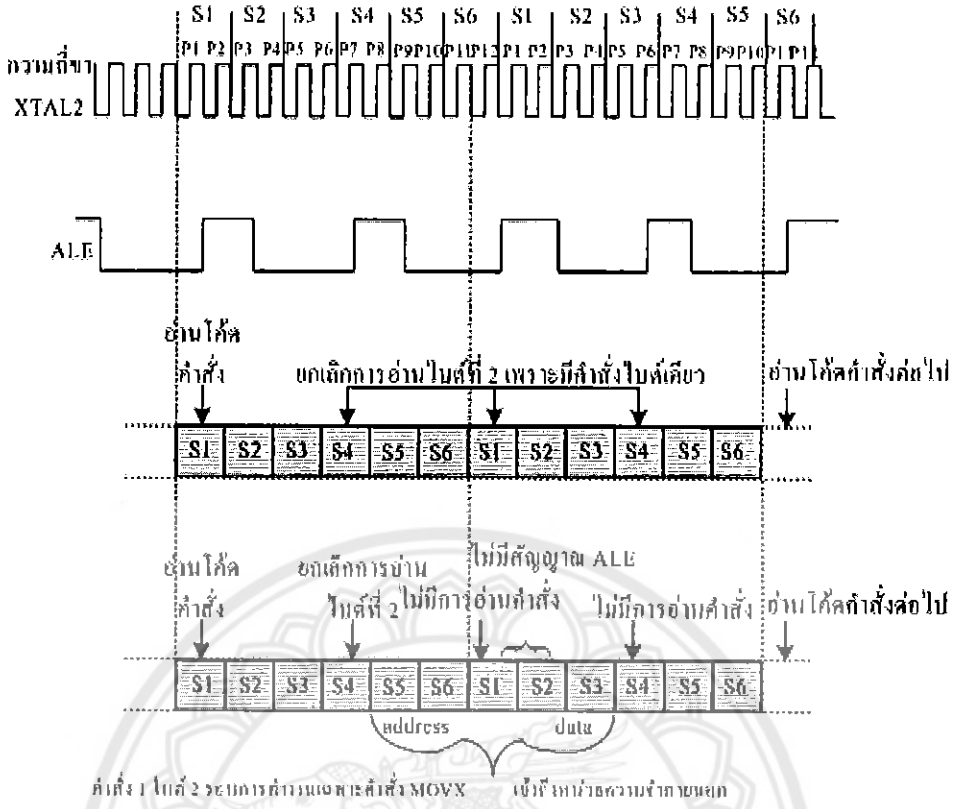
การกระทำคำสั่งที่มีขนาด 2 ไบต์ (โค้ดคำสั่ง (Code) และตำกระทำ (operand) และใช้เวลา 1 รอบการทำงาน เช่น คำสั่ง ADD A,#data , SUBB A,#data , MOV A,#data , MOV Rn,#data เป็นต้น การทำงานจะเป็นดังรูปที่ 2.27



รูปที่ 2.28 การทำงาน 1 รอบการทำงานของคำสั่งขนาด 2 ไบต์ [1]

การทำงาน 1 รอบการทำงานของคำสั่งขนาด 2 ไบต์ จะเห็นว่ามีการใช้สภาวะการทำงานทั้งหมด 6 สภาวะ คือ สภาวะที่ 1-3 เป็นการประมวลโค้ดคำสั่งของไบต์แรก และสภาวะที่ 4-6 เป็นการประมวลผลของไบต์ที่ 2 และสิ้นสุดการทำงานที่สภาวะที่ 6 จากนั้นจะเป็นการขึ้นรอบของการทำงานคำสั่งใหม่ต่อไป

การทำงานของคำสั่งที่มีขนาด 1 ไบต์ จะใช้ 2 รอบการทำงานหรือ 24 คาบเวลา ซึ่งมีข้อแตกต่างที่ขึ้นอยู่กับแต่ละคำสั่ง เช่น คำสั่ง INC DPTR กับ MOVX เป็นคำสั่งที่มีขนาดของโค้ดคำสั่งและรอบการทำงานที่เท่ากัน ดังรูปที่ 2.28

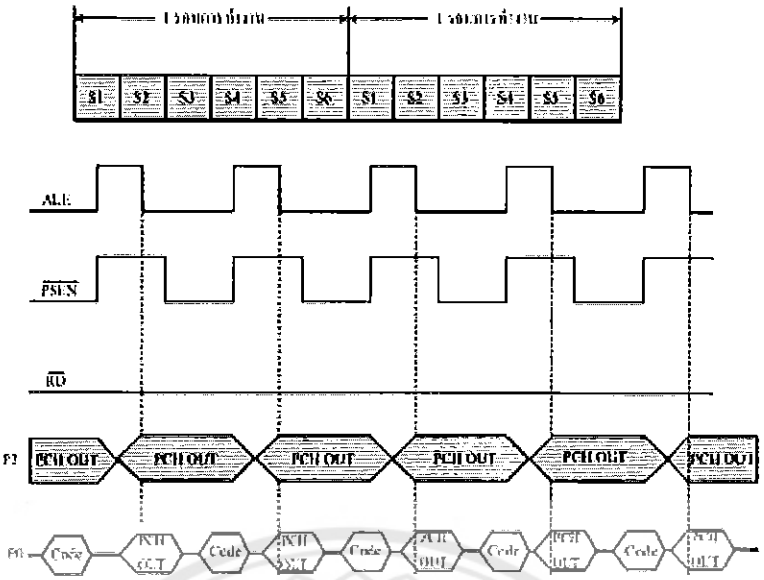


รูปที่ 2.29 การทำงาน 2 รอบการทำงานของคำสั่งขนาด 1 ไบต์ [1]

การทำงานของคำสั่ง 1 ไบต์ 2 รอบการทำงานที่ไม่ใช้คำสั่ง MOVX จะมีการอ่านโค้ดคำสั่งในสภาวะที่ 1 จากนั้นจะทำการประมวลผลจนถึงสภาวะที่ 6 (S6) ของรอบที่ 2 โดยไม่มีการอ่านโค้ดอีก ถ้าเป็นคำสั่ง MOVX การปฏิบัติงานตามคำสั่งจะไม่มีผลต่อการติดต่อกับหน่วยความจำภายในหรือภายนอกของไมโครคอนโทรลเลอร์

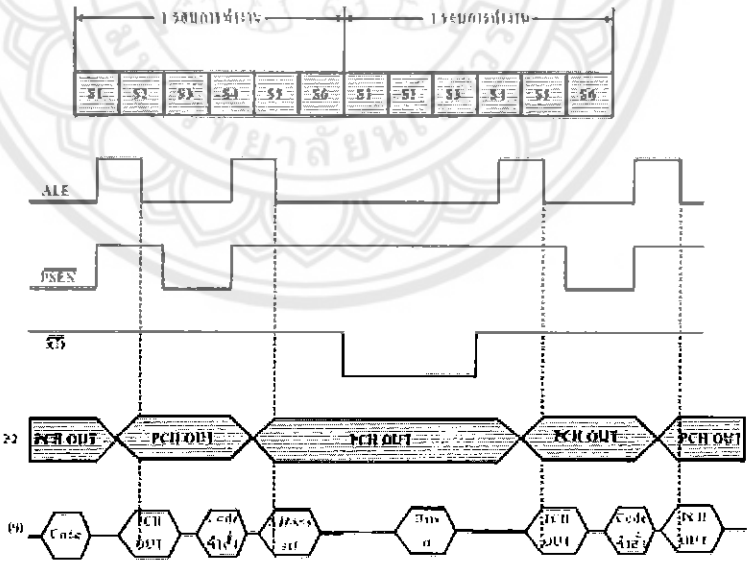
2.1.20 คาบเวลาในการติดต่อกับหน่วยความจำโปรแกรมภายนอก

เวลาในการเข้าถึงข้อมูลของหน่วยความจำโปรแกรมที่ต่ออยู่ภายนอกจะใช้สัญญาณ ALE และ $\overline{\text{PSEN}}$ ในการเลือกและอ่านข้อมูลโปรแกรม ซึ่งสัญญาณทั้งสองจะทำงาน 2 ครั้งต่อ 1 รอบการทำงานของคำสั่ง ซึ่งการอ่านแต่ละครั้งจะต้องใช้ 2 รอบการทำงาน และคำสั่งที่ใช้จะต้องเกี่ยวกับ รีจีสเตอร์ตัวนับขั้นตอนในโปรแกรม (PC หรือ program counter) ช่วงค่ากำหนดตำแหน่งไบต์สูง (A8-A15) จะส่งผ่านไปยังพอร์ต 2 ที่ทำหน้าที่ควบคุมและกำหนดค่าไบต์ต่ำ (A0-A7) พร้อมสัญญาณข้อมูล และพอร์ต 0 ซึ่งจังหวะเวลาการทำงานเป็นดังรูปที่ 2.29



รูปที่ 2.30 คาบเวลาในการติดต่อกับหน่วยความจำโปรแกรมภายนอก [1]

ในกรณีที่เป็นเวลาในการอ่านข้อมูลของหน่วยความจำข้อมูลที่ต่ออยู่ภายนอกโดยใช้คำสั่ง MOVX สัญญาณ $\overline{\text{PSEN}}$ จะไม่ถูกใช้งาน และสัญญาณ $\overline{\text{RD}}$ จะทำหน้าที่แทน ซึ่งจะเห็นว่าสัญญาณ ALE ในช่วงที่ไมโครคอนโทรลเลอร์ทำการอ่านข้อมูลจะไม่ทำงาน ดังรูปที่ 2.30



รูปที่ 2.31 เวลาในการอ่านข้อมูลของหน่วยความจำข้อมูลโดยใช้คำสั่ง MOVX [1]

ความสัมพันธ์ระหว่างเวลาในการอ่านข้อมูลจากภายนอก ของหน่วยความจำโปรแกรมและหน่วยความจำข้อมูลจะใช้ 2 รอบการทำงานเท่ากัน แต่จะมีการใช้คำสั่งที่ต่างกันคือ หน่วยความจำ

โปรแกรมจะไม่ใช้คำสั่ง MOVX แต่หน่วยความจำข้อมูลจะใช้คำสั่ง MOVX ซึ่งรูปแบบการทำงานของสัญญาณต่างๆ ให้พิจารณาจากรูปที่ 2.28 และ 2.29

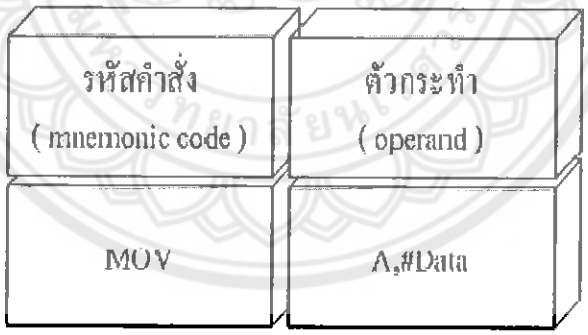
2.1.21 ชุดคำสั่งของไมโครคอนโทรลเลอร์ตระกูล MCS-51

ไมโครคอนโทรลเลอร์ตระกูล MCS-51 มีชุดคำสั่งทั้งหมด 111 คำสั่ง และแบ่งหน้าที่การทำงานออกเป็นกลุ่มตามลักษณะของงานที่ทำเพื่อให้ผู้เรียนสามารถศึกษาได้สะดวกขึ้น ซึ่งประกอบด้วยกลุ่มคำสั่งต่อไปนี้

1. กลุ่มคำสั่งทางคณิตศาสตร์ (Arithmetic instruction)
2. กลุ่มคำสั่งการกระทำลอจิก (Logical instruction)
3. กลุ่มสั่งการเคลื่อนย้ายข้อมูล (data transfer instruction)
4. กลุ่มคำสั่งการจัดการข้อมูลระดับบิต (bit manipulated instruction)
5. กลุ่มคำสั่งกระโดด (branch instruction)

รูปแบบการเขียนคำสั่ง

ภาษาแอสแซมบลีของไมโครคอนโทรลเลอร์ MCS-51 จะมีรูปแบบมาตรฐานคล้ายกับ ไมโครคอนโทรลเลอร์ทั่วๆ ไป ประกอบด้วย 2 ส่วนใหญ่ๆ คือ ส่วนที่เป็นรหัสคำสั่ง (mnemonic code) ส่วนที่เป็นตัวกระทำ (operand) ดังรูปที่ 2.31



รูปที่ 2.32 รูปแบบการเขียนภาษาแอสแซมบลี [4]

รหัสคำสั่ง (mnemonic code) คือ MOV เป็นคำสั่งให้ไมโครคอนโทรลเลอร์รู้หน้าที่การทำงานและการย้ายข้อมูลจากค่าตัวกระทำ (operand) คือ A,#data และนำเอาข้อมูลจากที่กำหนดโดยตรงมาเก็บไว้ในรีจีสเตอร์ A ที่อยู่ภายในไมโครคอนโทรลเลอร์

การคำนวณหาค่าเวลาในการทำงานของแต่ละคำสั่งของไมโครคอนโทรลเลอร์ทำให้สามารถทราบถึงเวลาในการทำงานของคำสั่งในโปรแกรมทั้งหมด เพื่อกำหนดและออกแบบวงจรควบคุมอุปกรณ์ภายนอกให้ทำงานสัมพันธ์กัน โดยเฉพาะค่าหน่วงเวลาการทำงานของอุปกรณ์ต่างๆ ซึ่งการ

ทำงานในแต่ละคำสั่งจะใช้เวลาประมวลผลที่แตกต่างกัน ทุกคำสั่งจะมีค่าเวลาของรอบการทำงานที่แน่นอน และสามารถคำนวณหาค่าเวลาของรอบการทำงานได้จากวิธีการต่อไปนี้

สูตรในการคำนวณ

$$Time = Mc \times \frac{12}{f_{cystal}}$$ สมการที่ 2.2

Time = เวลาที่ใช้ประมวลผลคำสั่งทั้งหมด (วินาที)

Mc = เวลาของรอบการทำงานของคำสั่ง

f_{cystal} = ค่าความถี่ของคริสตอลที่ต่อในวงจรกำเนิดสัญญาณนาฬิกาให้กับไมโครคอนโทรลเลอร์ตระกูล MCS-51

2.1.22 ค่าตัวแปรที่กำหนดให้คำสั่ง

ในชุดคำสั่งของไมโครคอนโทรลเลอร์ตระกูล MCS-51 จะใช้ค่าตัวแปรหรืออักษรย่อต่างๆ ในการกำหนดตำแหน่งการทำงาน ตารางรูปที่ 2.8

ตารางที่ 2.8 ค่าตัวแปรที่กำหนดในคำสั่ง [4]

สัญลักษณ์	รายละเอียดการทำงานของตำแหน่ง
Rn	รีจีสเตอร์แบงก์ R0-R7 ในตำแหน่งที่กำหนดปัจจุบัน
direct	ข้อมูล 8 บิต เป็นตำแหน่งหน่วยความจำข้อมูล (RAM) เข้าถึงได้โดยตรง (0-127) หรือ รีจีสเตอร์ SFR (128-256)
@Ri	ข้อมูล 8 บิต เป็นตำแหน่งหน่วยความจำข้อมูล (RAM) ภายใน เข้าถึงได้โดยตรง (0-256) โดยอ้างผ่านรีจีสเตอร์ R0 และ R1
#data	ค่าคงที่ขนาด 8 บิต เป็นการกำหนดข้อมูลโดยตรงในคำสั่ง
#data 16	ค่าคงที่ขนาด 16 บิต เป็นการกำหนดข้อมูลโดยตรงในคำสั่ง
Add 16	ข้อมูล 16 บิต เป็นค่าตำแหน่งการกระโดดไปของไมโครคอนโทรลเลอร์ในหน่วยความจำโปรแกรม ซึ่งสามารถกระโดดไปที่ระยะทาง 64 กิโลไบต์ ใช้กับคำสั่ง LCALL , LJMP
Add 11	ข้อมูล 11 บิต เป็นค่าตำแหน่งการกระโดดไปของไมโครคอนโทรลเลอร์ในหน่วยความจำโปรแกรม ซึ่งสามารถกระโดดไปที่ระยะทาง 64 กิโลไบต์ ใช้กับคำสั่ง ACALL , AJMP
rel	ข้อมูล 8 บิต เป็นการอ้างอิงระยะทางของไมโครคอนโทรลเลอร์ และจะกระโดดไปทำงานในระยะทางอ้างอิงตั้งแต่ -128 ถึง + 127 ไบต์
bit	เป็นตำแหน่งบิตที่เข้าถึงได้โดยตรงของหน่วยความจำข้อมูลภายใน หรือรีจีสเตอร์ SFR (Special Function Register)

2.1.23 กลุ่มคำสั่งทางคณิตศาสตร์ (Arithmetic Instruction)

เป็นชุดคำสั่งที่ทำงานเกี่ยวกับการคำนวณทางคณิตศาสตร์ การบวก การลบ การคูณ การหาร การเพิ่มค่าและลดค่า ซึ่งจะใช้รีจีสเตอร์ Acc และ B เป็นตัวทำงานหลัก ส่วนคำสั่งการลดค่าและเพิ่มค่ากระทำได้กับรีจีสเตอร์ทั่วไปและหน่วยความจำ ดังตารางที่ 2.9

ตารางที่ 2.9 กลุ่มคำสั่งทางคณิตศาสตร์ [4]

รหัสคำสั่ง	รายละเอียดของคำสั่ง	รหัสไบต์	จำนวนควบเวลา
ADD A,Rn	บวกค่าในรีจีสเตอร์ Acc ด้วยรีจีสเตอร์ Rn	1	12
ADD A,direct	บวกค่าในรีจีสเตอร์ Acc ด้วยรีจิสเตอร์ direct	2	12
ADD A,@Ri	บวกค่าในรีจีสเตอร์ Acc ด้วย RAM ภายในผ่านรีจีสเตอร์ Ri	1	12
ADD A,#data	บวกค่าข้อมูลกับรีจีสเตอร์ Acc	2	12
ADDC A,Rn	บวกค่ารีจีสเตอร์ Acc ด้วยรีจีสเตอร์ Rn พร้อมบิตตัวทด	1	12
ADDC A,Direct	บวกค่ารีจีสเตอร์ Acc ด้วยค่าใน direct พร้อมบิตตัวทด	2	12
ADD A,@Ri	บวกค่าใน RAM ภายในกับรีจีสเตอร์ Acc พร้อมบิตตัวทด	1	12
ADDC A,#data	บวกค่าข้อมูลกับรีจีสเตอร์ Acc พร้อมบิตตัวทด	1	12
SUBB A,Rn	ลบค่าในรีจีสเตอร์ Acc ด้วยรีจีสเตอร์ Rn พร้อมบิตตัวทด	1	12
SUBB A,Direct	ลบค่าในรีจีสเตอร์ Acc ด้วยค่าใน direct พร้อมบิตตัวทด	1	12
SUBB A,@Ri	ลบค่าในรีจีสเตอร์ Acc ด้วยค่า RAM ภายใน พร้อมบิตตัวทด	2	12
SUBB A,#data	ลบค่าในรีจีสเตอร์ Acc ด้วยค่าข้อมูล พร้อมบิตตัวทด	1	12
INC A	เพิ่มค่าข้อมูลใน Acc ขึ้น 1 ค่า	1	12
INC Rn	เพิ่มค่าข้อมูลใน Rn ขึ้น 1 ค่า	2	12
INC Direct	เพิ่มค่าข้อมูลใน direct ขึ้น 1 ค่า	1	12
INC @Ri	เพิ่มค่าใน direct RAM ผ่านรีจีสเตอร์ Ri ขึ้น 1 ค่า	1	12
DEC A	ลดค่าในรีจีสเตอร์ Acc ลง 1 ค่า	1	12
DEC Rn	ลดค่าใน Rn ลง 1 ค่า	1	12
DEC @Ri	ลดค่าใน direct RAM ผ่านรีจีสเตอร์ Ri ลง 1 ค่า	1	12
INC DPTR	เพิ่มค่าในรีจีสเตอร์ DPTR ขึ้น 1 ค่า	1	24
MUL AB	คูณค่าในรีจีสเตอร์ Acc ด้วยรีจีสเตอร์ B	2	48
DLV AB	หารค่ารีจีสเตอร์ Acc ด้วยรีจีสเตอร์ B	2	48
DA A	ปรับค่าในรีจีสเตอร์ Acc ให้เป็นฐาน 10	1	12

2.1.24 กลุ่มคำสั่งการกระทำลอจิก (Logical Instruction)

กลุ่มคำสั่งที่ทำงานเกี่ยวกับการกระทำทางลอจิกเบื้องต้น ได้แก่ วงจรลอจิก AND , OR , NOT ,X-OR การหมุนและเลื่อนค่าในรีจีสเตอร์ การแลกเปลี่ยนของค่าระหว่างรีจีสเตอร์และหน่วยความจำ ดังตารางที่ 2.10

ตารางที่ 2.10 กลุ่มคำสั่งการกระทำลอจิก [4]

รหัสคำสั่ง	รายละเอียดของคำสั่ง	รหัสไบต์	จำนวนคาบเวลา
ANL A,Rn	AND ค่ารีจีสเตอร์ Acc ด้วยรีจีสเตอร์ Rn	1	12
ANL A.direct	AND ค่ารีจีสเตอร์ Acc ด้วยด้วยค่าใน direct	2	12
ANL A,@Ri	AND ค่ารีจีสเตอร์ Acc ด้วยค่าในหน่วยความจำผ่านค่ารีจีสเตอร์ Ri (R0,R1)	1	12
ANL A,#data	AND ค่ารีจีสเตอร์ Acc ด้วยค่าข้อมูลโดยตรง	2	12
ANL direct,A	AND ค่าใน direct ด้วยค่าในรีจีสเตอร์ Acc	2	12
ANL direct,#data	AND ค่าใน direct ด้วยค่าข้อมูลโดยตรง	3	12
ORL A,Rn	ORL ค่ารีจีสเตอร์ Acc ด้วยรีจีสเตอร์ Rn	1	12
ORL A,direct	ORL ค่ารีจีสเตอร์ Acc ด้วยด้วยค่าใน direct	2	12
ORL A,@Ri	ORL ค่ารีจีสเตอร์ Acc ด้วยค่าในหน่วยความจำผ่านค่ารีจีสเตอร์ Ri (R0,R1)	1	12
ORL A,#data	ORL ค่ารีจีสเตอร์ Acc ด้วยค่าข้อมูลโดยตรง	2	12
ORL direct,A	ORL ค่าใน direct ด้วยค่าในรีจีสเตอร์ Acc	2	12
ORL direct,#data	ORL ค่าใน direct ด้วยค่าข้อมูลโดยตรง	3	12
XRL A,Rn	XRL ค่ารีจีสเตอร์ Acc ด้วยรีจีสเตอร์ Rn	1	12
XRL A,direct	XRL ค่ารีจีสเตอร์ Acc ด้วยด้วยค่าใน direct	2	12
XRL A,@Ri	XRL ค่ารีจีสเตอร์ Acc ด้วยค่าในหน่วยความจำผ่านค่ารีจีสเตอร์ Ri (R0,R1)	1	12
XRL A,#data	XRL ค่ารีจีสเตอร์ Acc ด้วยค่าข้อมูลโดยตรง	2	12
XRL direct,A	XRL ค่าใน direct ด้วยค่าในรีจีสเตอร์ Acc	2	12
XRL direct,#data	XRL ค่าใน direct ด้วยค่าข้อมูลโดยตรง	3	24
CLR A	ทำให้ค่ารีจีสเตอร์ Acc มีค่าเป็น 0 ทุกบิต	1	12
CPL A	กลุ่มค่าในรีจีสเตอร์ Acc ให้มีค่าจาก 0 $\rightarrow$ 1 และจาก 1 $\rightarrow$ 0	1	12
RL A	หมุนค่าในรีจีสเตอร์ Acc จาก 7 ไปบิต 0	1	12
RLC A	หมุนค่าในรีจีสเตอร์ Acc จาก 7 ผ่านบิตตัวทดไปบิต 0	1	12
RR A	หมุนค่าในรีจีสเตอร์ Acc จาก 0 ไปบิต 7	1	12
RRC A	หมุนค่าในรีจีสเตอร์ Acc จาก 0 ผ่านบิตตัวทดไปบิต 7	1	12
SWAP A	สลับค่าบิตภายในรีจีสเตอร์ Acc จาก (A_{3-0}) $\leftrightarrow$ (A_{7-4})	1	12

2.1.25 กลุ่มคำสั่งการเคลื่อนย้ายข้อมูล (Data Transfer Instruction)

คำสั่งการเคลื่อนย้ายข้อมูลเป็นการคัดลอกข้อมูลระหว่างรีจีสเตอร์กับรีจีสเตอร์ รีจีสเตอร์กับหน่วยความจำทั้งภายในและภายนอก และหน่วยความจำกับหน่วยความจำ ตารางรูปที่ 2.11

ตารางที่ 2.11 กลุ่มคำสั่งการเคลื่อนย้ายข้อมูล [4]

รหัสคำสั่ง	รายละเอียดของคำสั่ง	รหัสไบต์	จำนวนคาบเวลา
MOV A,Rn	คัดลอกข้อมูลจากรีจีสเตอร์ Rn ไปที่รีจีสเตอร์ Acc	1	12
MOV A,direct	คัดลอกข้อมูลจาก direct ไปที่รีจีสเตอร์ Acc	2	12
MOV A,@Ri	คัดลอกข้อมูลจากหน่วยความจำ RAM ไปที่รีจีสเตอร์ Acc	1	12
MOV A,#data	คัดลอกข้อมูลโดยตรงไปที่รีจีสเตอร์ Acc	2	12
MOV Rn,A	คัดลอกข้อมูลจากรีจีสเตอร์ Acc ไปที่รีจีสเตอร์ Rn	1	12
MOV Rn,direct	คัดลอกข้อมูลจาก direct ไปที่รีจีสเตอร์ Rn	2	24
MOV Rn,#data	คัดลอกข้อมูลจากข้อมูลโดยตรงไปที่รีจีสเตอร์ Rn	2	12
MOV direct,A	คัดลอกข้อมูลจากรีจีสเตอร์ Acc ไปที่ direct	2	12
MOV direct,Rn	คัดลอกข้อมูลจากรีจีสเตอร์ Rn ไปที่ direct	2	24
MOV direct,direct	คัดลอกข้อมูลจาก direct ไปที่ direct	3	24
MOV direct,@Ri	คัดลอกข้อมูลจากหน่วยความจำ RAM ไปที่ direct	2	24
MOV direct,#data	คัดลอกข้อมูลจากข้อมูลโดยตรงไปที่ direct	3	24
MOV @Ri,A	คัดลอกข้อมูลจากรีจีสเตอร์ Acc ไปที่หน่วยความจำ RAM	1	12
MOV @Ri.direct	คัดลอกข้อมูลจาก direct ไปที่หน่วยความจำ RAM	2	24
MOV @Ri,#data	คัดลอกข้อมูลจากข้อมูลไปที่หน่วยความจำ RAM	2	24
MOV DPTR,#DATA 16	คัดลอกข้อมูล 16 บิต จากข้อมูลไปที่รีจีสเตอร์ DPTR	3	24
MOVC A,@A+DPTR	คัดลอกข้อมูลจากตำแหน่ง A+DPTR ไปที่รีจีสเตอร์ Acc	1	24
MOVC A,@A+PC	คัดลอกข้อมูลจากตำแหน่ง A+PC ไปที่รีจีสเตอร์ Acc	1	24
MOVX @Ri,A	คัดลอกข้อมูลจาก RAM ภายนอกไปที่รีจีสเตอร์ Acc	1	24
MOVX @DPTR,A	คัดลอกข้อมูลจากรีจีสเตอร์ Acc ไปที่รีจีสเตอร์ DPTR กำหนด	1	24
PUSH DIRECT	นำข้อมูล 8 บิต ไปเก็บในสแตก (SP)	2	24
POP DIRECT	นำข้อมูล 8 บิต ออกเก็บในสแตก (SP)	2	24
XCH A,Rn	แลกเปลี่ยนข้อมูลระหว่างรีจีสเตอร์ Acc กับรีจีสเตอร์ Rn	1	12
XCH A,DIRECT	แลกเปลี่ยนข้อมูลระหว่างรีจีสเตอร์ Acc กับหน่วยความจำ RAM	2	12
XCH A,@Ri	แลกเปลี่ยนข้อมูลระหว่างรีจีสเตอร์ Acc กับรีจีสเตอร์ Ri กำหนด ตำแหน่ง	1	12
XCHD A,@Ri	แลกเปลี่ยนบิต 3-0 ระหว่างรีจีสเตอร์ Acc กับตำแหน่ง direct	1	12

2.1.26 กลุ่มคำสั่งการจัดการข้อมูลระดับบิต (Bit Manipulated Instruction)

เป็นกลุ่มของคำสั่งที่ทำหน้าที่เกี่ยวกับหน่วยความจำที่เข้าถึงระดับบิตโดยตรง (รีจีสเตอร์ หน่วยความจำภายใน) โดยการเปลี่ยนแปลงค่าข้อมูลระดับบิต การทำให้มีค่าสภาวะลอจิกเป็น "0" หรือ "1" การกลับค่าบิต ดังตารางที่ 2.12

ตารางที่ 2.12 กลุ่มคำสั่งการจัดการข้อมูลระดับบิต [4]

รหัสคำสั่ง	รายละเอียดของคำสั่ง	รหัสไบต์	จำนวนคาบเวลา
CLR C	ทำให้บิตตัวทดมีค่าเป็น 0	1	12
CLR bit	ทำให้บิตที่กำหนดมีค่าเป็น 0	2	12
SETB C	กำหนดให้บิตตัวทดมีค่าเป็น 1	1	12
SETB bit	ทำให้บิตที่กำหนดมีค่าเป็น 1	2	12
CPL	กลับค่าในบิตตัวทดจาก 0 → 1 , 1 → 0	1	12
CPL bit	กลับค่าในบิตที่กำหนดตัวทดจาก 0 → 1 , 1 → 0	2	24
ANL C,bit	AND บิตตัวทดด้วยบิตที่กำหนด	2	24
ANL C,/bit	AND บิตตัวทดด้วยบิตที่กำหนดกลับค่า	2	12
ORL C,bit	OR บิตตัวทดด้วยบิตที่กำหนด	2	24
ORL C,/bit	OR บิตตัวทดด้วยบิตที่กำหนดกลับค่า	2	24
MOV C,bit	คัดลอกบิตที่กำหนดไปบิตตัวทด	2	24
MOV bitj,C	คัดลอกบิตตัวทดไปบิตที่กำหนด	2	24
JC rel	กระโดดไปที่ตำแหน่ง rel เมื่อบิตตัวทดมีค่าเป็น 1	2	24
JNC rel	กระโดดไปที่ตำแหน่ง rel เมื่อบิตตัวทดมีค่าไม่เป็น 1	2	24
JB bit,rel	กระโดไปเมื่อบิตที่กำหนดมีค่าเป็น 1	3	24
JNB bit,rel	กระโดไปเมื่อบิตที่กำหนดมีค่าไม่เป็น 1	3	24
JBC bit,rel	กระโดดไปเมื่อบิตกำหนดเป็น 1 และทำให้บิตนั้นเป็น 0	3	24

2.1.27 กลุ่มคำสั่งการกระโดด (Branch Instruction)

เป็นกลุ่มคำสั่งที่ทำหน้าที่เกี่ยวกับการกระโดดไปที่ตำแหน่งที่โปรแกรมกำหนด โดยมีลักษณะการกระโดดอยู่ 2 แบบ คือ การกระโดดโดยไม่มีเงื่อนไข เมื่อพบคำสั่งก็กระโดดไปตำแหน่งที่

กำหนดทันที่และการกระโดดโดยมีเงื่อนไข เมื่อไมโครคอนโทรลเลอร์พบคำสั่งจะต้องตรวจสอบเงื่อนไขก่อนว่าเป็นไปตามที่กำหนดหรือไม่ ถ้าถูกต้องจึงกระโดดไปที่ตำแหน่งที่กำหนด

การกระโดดไปทำงานก็มีรูปแบบคือ กระโดดไปโดยเป็นการเรียกใช้โปรแกรมย่อย โดยจะต้องกลับมาทำงานได้ก็ต่อเมื่อเสร็จจากการทำงานในโปรแกรมย่อย (ACALL, LCALL) และกระโดดไปทำงานโดยไม่กลับมา (SJMP, LJMP) ระยะทางการกระโดดก็มีให้เลือกเป็นระยะแบบสั้น (256 ไบต์ หรือ 2 กิโลไบต์) ระยะ (64 กิโลไบต์) ดังตารางที่ 2.13

ตารางที่ 2.13 กลุ่มคำสั่งการกระโดด [4]

รหัสคำสั่ง	รายละเอียดของคำสั่ง	รหัสไบต์	จำนวนคาบเวลา
ACALL add11	เรียกโปรแกรมย่อยที่ addr11	2	24
LCALL addr16	เรียกโปรแกรมย่อยที่ addr16	3	24
RET	กลับจากโปรแกรมย่อยไปโปรแกรมหลัก	1	24
RETI	กลับจากการเรียกใช้อินเตอร์รัปต์	1	24
AJMP addr11	กระโดดทำงานที่กำหนดตำแหน่งด้วย addr11	2	24
LJMP addr16	กระโดดทำงานที่กำหนดตำแหน่งด้วย addr16	3	24
SJMP rel	กระโดดไปทำงานที่กำหนดตำแหน่งด้วย rel	2	24
JMP @A+DPTR	กระโดดไปทำงานที่ตำแหน่ง A+DPTR	1	24
JZ rel	กระโดดไปทำงานที่ rel เมื่อค่าในรีจีสเตอร์ Acc = 00	2	24
JNZ rel	กระโดดไปทำงานที่ rel เมื่อค่าในรีจีสเตอร์ Acc ≠ 00	2	24
CJNE A,direct,rel	ถ้า direct ≠ Acc กระโดดไปที่ rel	3	24
CJNE A,#data,rel	ถ้า #data ≠ Acc กระโดดไปที่ rel	3	24
CJNE Rn,#data,rel	ถ้า direct ≠ Rn กระโดดไปที่ rel	3	24
CJNE @Ri,#data,rel	ถ้า direct ≠ @Ri กระโดดไปที่ rel	3	24
DJNZ Rn,rel	ถ้า Rn-1 ≠ 0 กระโดดไปที่ rel	2	24
DJNZ direct,rel	ถ้า direct – 1 ≠ 0 กระโดดไปที่ rel	3	24
NOP	เพิ่มค่า PC ขึ้น 1 ค่า (ไม่มีการทำงานอย่างอื่น)	1	12

2.2 บอร์ดโทรศัพท์ E12-SLC (Telephone line interface) [7]

E12-SLC คือ Application Board ซึ่งออกแบบให้ใช้งานร่วมกับ Controller Board สำหรับงานประยุกต์ที่ต้องควบคุมระบบโดยผ่านคู่สายโทรศัพท์ เช่น ระบบบันทึกข้อมูลการใช้โทรศัพท์

ระบบแจ้งเหตุร้ายอัตโนมัติ ระบบรับ-ส่งข้อมูลผ่านคู่สายโทรศัพท์ด้วยรหัส DTMF (Dual Tone Modulation Frequency) หรือระบบประยุกต์อื่น ๆ

2.2.1 คุณสมบัติ

1. ไอซีเบอร์ MT8888C เพื่อทำหน้าที่เชื่อมตอสัญญาณระหว่างคู่สายโทรศัพท์กับ Controller Board รวมถึงการสร้างสัญญาณ DTMF เพื่อส่งออกไปยังคู่สายโทรศัพท์

2. มี LED แสดงสถานะสัญญาณ RINGGING ซึ่งเป็นสัญญาณเรียกเข้า ซึ่งต่อใช้งานร่วมกับ INT0

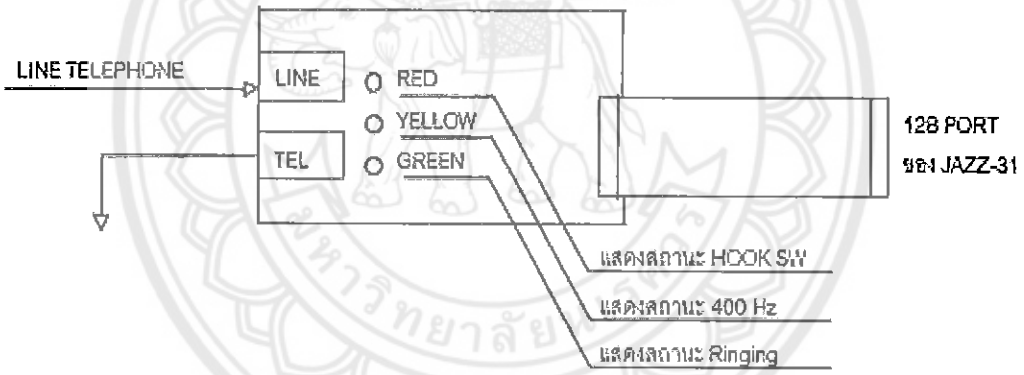
3. มี LED แสดงสถานะสัญญาณ การยกหู-วางหูต่อใช้งานร่วมกับ INT1

4. มี LED แสดงสถานการณ์ตรวจจับสัญญาณ Busy Tone , Dial Tone, และ Ring back ที่ความถี่ 400-425 Hz

5. สามารถต่อใช้งานร่วมกับ Controller Board ผ่านทาง Port 1 มาตรฐานศิลาได้ทันที

6 . รับไฟเลี้ยง 5VDC จาก Controller Board

7. SIZE PCB 2x3.9 INCH



รูปที่ 2.33 การต่อใช้งานร่วมกับ Single Board JAZZ-31 [7]

2.2.2 การใช้งาน

1. ต่อสายให้ครบตามรูป

2. เสียบ ADAPTOR 9VDC ที่ JAZZ-31 สังเกตว่า LED สีเหลืองสว่าง

3. เข้าสู่ PROGRAM สื่อสาร XATLK เพื่อ DOWN LOAD PROGRAM E12_SLC.HEX เข้าเครื่อง JAZZ-31 จากนั้นสั่ง RUN PROGRAM ที่ ADDRESS 8100H

4. ที่เครื่อง JAZZ-31 จะแสดง 'E12-SLC' และตามด้วย ' SEL 0-4 ' ซึ่งหมายถึงให้เลือก Menu ที่ต้องการทดสอบซึ่งรายละเอียดของแต่ละ Menu จะเป็นดังนี้

2.2.3 รายละเอียดของแต่ละเมนู

- Menu 0 สำหรับการทดสอบ ยกหู-วางหู ซึ่งการทำงานของ KEY 0 จะทำงานในลักษณะ TOGGLE SW

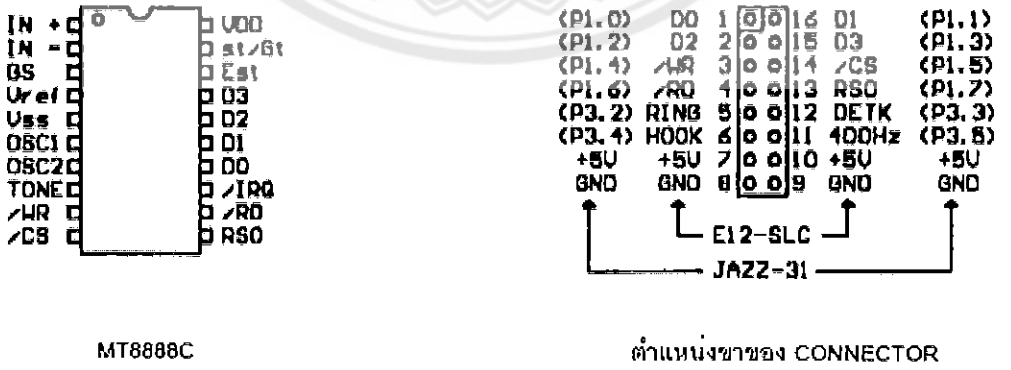
- Menu 1 สำหรับการทดสอบการรับสัญญาณเรียกเข้า หรือ Ringing ควบคุมด้วยการนับจำนวนครั้งเรียกเข้าโดยจะแสดงบน 7- SEGMENT ของ JAZZ-31 ออกจาก Menu ด้วยการกด 1 อีกครั้ง

- Menu 2 สำหรับการทดสอบการรับสัญญาณรหัส DTMF ซึ่งที่ JAZZ-31 จะแสดงข้อความ WAIT ที่ 7- SEGMENT โดยจะรอรับสัญญาณเรียกเข้า 4 ครั้ง จึงจะรับสาย ผู้ใช้สามารถทดสอบได้โดยการกดโทรศัพท์เรียกเข้ามาที่เครื่อง หรือ KEY 2 อีกครั้ง จะข้ามขั้นตอนการรอรับสัญญาณกระดิ่งไปรับสัญญาณ DTMF โดยตรง ขั้นตอนนี้ผู้ใช้สามารถกด KEY ที่เครื่องรับโทรศัพท์ได้เลยโดย JAZZ-31 จะแสดงหมายเลขที่กดให้ทราบ

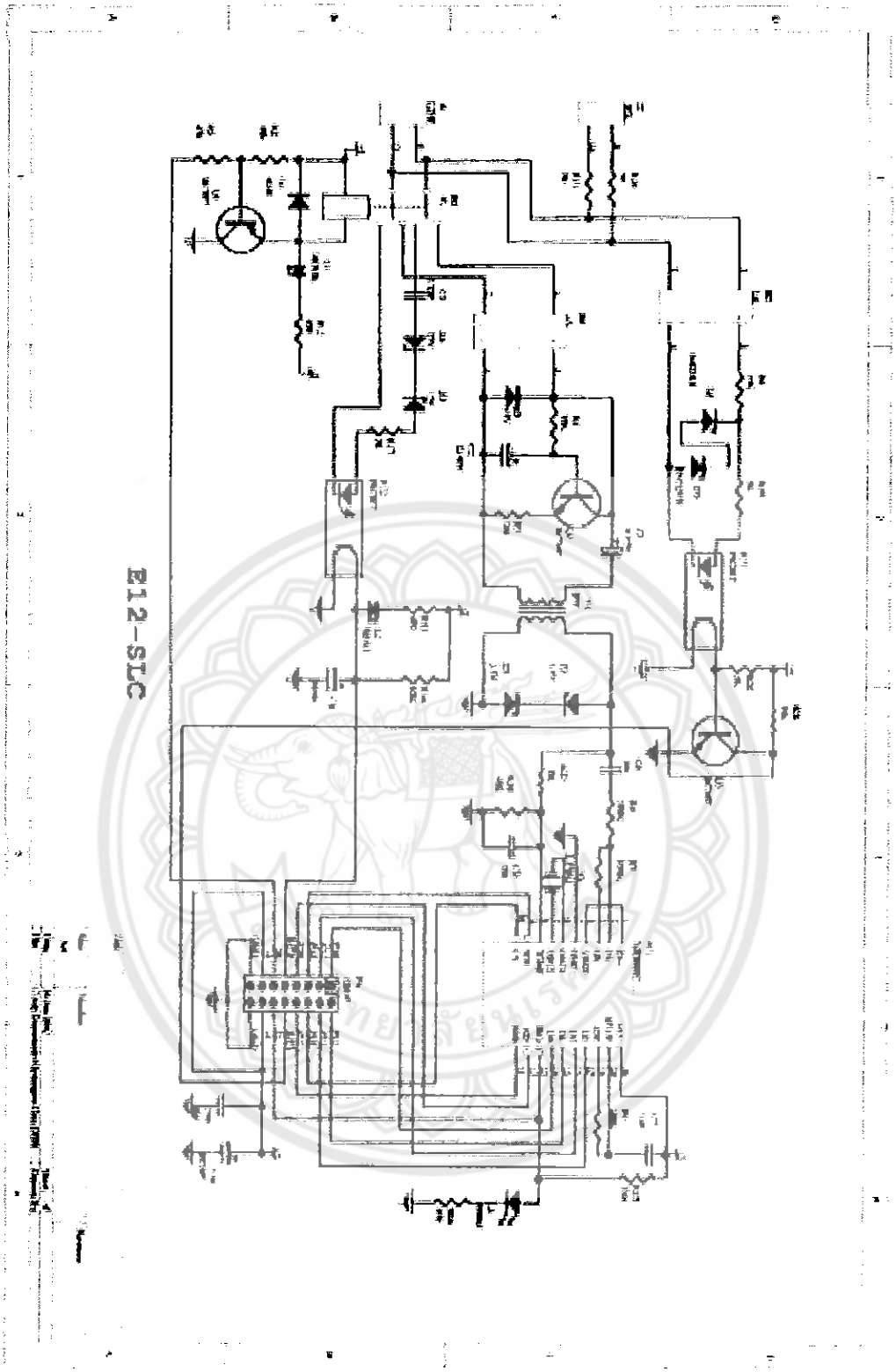
- Menu 3 สำหรับการทดสอบการรับสัญญาณรหัส DTMF เมื่อเลือก Menu นี้ Display จะแสดง "Send" ค้างไว้ ซึ่งขณะนี้จะจำลอง KEY BOARD ของ JAZZ-31ให้เป็น KEY BOARD ของโทรศัพท์ ผู้ใช้สามารถทดสอบโดยกด KEYS หมายเลขเพื่อโทรออกได้ทันที ตรวจสอบได้โดยให้ยกหูโทรศัพท์ที่ต่อพ่วงเพื่อฟังเสียงสัญญาณ ออกจาก Menu นี้โดยการกด KEY "F"

- Menu 4 สำหรับการทดสอบการรับสัญญาณ ยกหู-วางหู ของเครื่องโทรศัพท์ที่นำมาต่อพ่วง ซึ่งสังเกตผลบน Display ของ JAZZ-31 ได้ คือ จะมีการเปลี่ยนแปลงเมื่อมีการ ยกหู หรือ วางหู และออกจาก Menu นี้ ได้โดยวางหูทิ้งไว้ซักครู่ จะกลับสู่ Menu ปกติ

การนำไปประยุกต์ใช้งาน ผู้ใช้สามารถนำโปรแกรมตัวอย่างไปดัดต่อเพื่อให้เข้ากับงานประยุกต์ที่ต้องการได้ การจัดวางของ CHIPS และ CONNECTOR แสดงดังรูปที่ 2.34 และวงจรของ E12-SLC แสดงดังรูปที่ 2.35



รูปที่ 2.34 แสดงการจัดวางของ CHIPS และ CONNECTOR [7]



รูปที่ 2.35 วงจรของ E12-SLC [7]

2.3 บอร์ดบันทึกเสียง ISD 4003 [7]

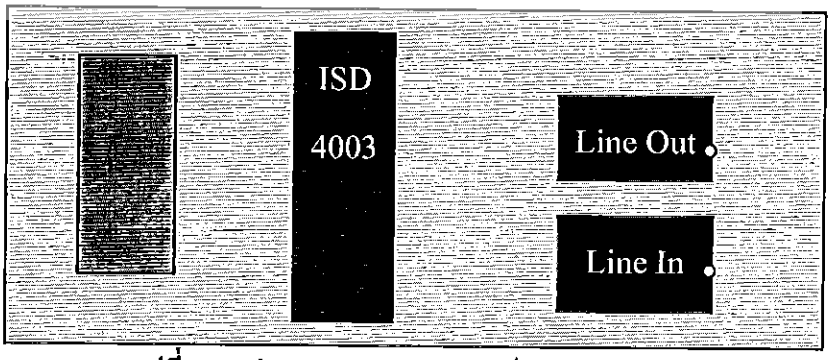
บอร์ดบันทึกเสียงเป็นบอร์ดที่ทำหน้าที่บันทึกและเล่นเสียง (Record/play) ซึ่งใช้ไอซีบันทึกเสียงตระกูล ISD4003-05M ซึ่งมีคุณสมบัติดังนี้

2.3.1 คุณสมบัติ

- สามารถเล่นและบันทึกเสียงได้ในตัวเดียว
- ทำงานที่แรงดัน 3 V
- กินกระแส 15 mA ขณะเล่น และกินกระแส 25 mA ขณะบันทึก
- บันทึกได้นาน 4,5,6,8 นาที ตามขนาดของไอซี
- บันทึกซ้ำได้มากกว่า 100,000 ครั้ง
- จดจำได้นานถึง 100 ปี
- ติดต่อสื่อสารแบบ SPI (Serial Peripheral Interface)

2.3.2 หน้าที่ของขาต่าง ๆ

SS,SCLK,MISO,MOSI	เป็นขาสัญญาณที่ใช้ในการควบคุม ISD4003
XCLK	เป็นขาที่รับสัญญาณนาฬิกาเพื่อการ Sampling สัญญาณเสียง แต่โดยปกติแล้วเราจะใช้สัญญาณนาฬิกาภายในดังนี้ ถ้าขานี้ไม่ใช่จะต่อลง Ground
INT	ขานี้จะเป็นลอจิก 0 เมื่อเล่นจนหมดหน่วยความจำ หรือ เล่นจนหมดในแต่ละข้อความนั้น
ANA IN+, ANA IN-	เป็นขาอินพุตเพื่อรับสัญญาณเสียงจากภายนอกเพื่อการบันทึกเสียง
AOUT	สัญญาณเสียงจะออกจากขานี้เมื่อ อยู่ในโหมดของการ Play
AM CAP	เป็นขาที่ใช้ในการลดสัญญาณรบกวนขณะเล่นเสียง
VccD,VccA	เป็นขา Ground ของไอซี
NC	ไม่ต้องต่อ



รูปที่2.36 ส่วนการทำงานของบอร์ดอัดเสียง ISD 4003

2.4 สวิตซ์แถบแม่เหล็ก [8]

สวิตซ์ สำหรับกลอนประตู TDK SW Catch(ตัวจับสัญญาณ) ยึดติดกับประตู , แผงที่ติดเครื่องวัดไฟฟ้า และฝาครอบบนอุปกรณ์ทุกรูปแบบ กลอนประตูชนิดนี้ยังบรรจุสวิตซ์reedอีกด้วย ซึ่งสามารถใช้รับความรู้สึกถ้าประตูถูกเปิดหรือปิด สวิตซ์ถูกกระตุ้นโดยวงจรแม่เหล็กที่มีการออกแบบแบบTDKมีลักษณะเฉพาะพิเศษ

2.4.1 คุณสมบัติ

1. (ตัวจับสัญญาณ)SW สามารถยึดติดกับประตู , ฝาครอบบนอุปกรณ์ทุกรูปแบบ เมื่อประตูถูกเปิดหรือปิดสัญญาณการยืนยันจะถูกรับได้โดยง่าย

2. การจัดเตรียมเครื่องลงกลอนประตูที่มีความแข็งแรงร่วมกับภาวะสวิตซ์ระบบอิเล็กทรอนิกส์ที่เชื่อมต่อกันที่เชื่อถือได้

3. การทำงานของอิเล็กทรอนิกส์มีความน่าเชื่อถือสูงเนื่องจากสวิตซ์reed มีการสัมผัส(ทำให้วงจรไฟฟ้าทำงานได้เต็มวง)

4. การทำงานเกี่ยวกับเครื่องจักรกลมีความน่าเชื่อถือสูง (วงจรในการเปิดหรือปิดมากกว่า 100,000 cycle) เนื่องมาจากยางหุ้มพิเศษ

5. มีรูปร่างกะทัดรัดและติดตั้งได้ง่าย

2.4.2 การประยุกต์

ในเรื่องของการลงกลอนประตูและการรับรู้ถึงการถูกสัมผัสของประตู , แผงที่ติดเครื่องวัดไฟฟ้า และฝาครอบบนเครื่องถ่ายสำเนา , เครื่องพิมพ์คอมพิวเตอร์ , อุปกรณ์สำนักงานอัตโนมัติ และอุปกรณ์โรงงานอัตโนมัติ และเครื่องจักรกล

SWITCH

ระดับการสัมผัสสูงสุด	10 วัตต์
แรงดันไฟฟ้าEdcสัมผัสสูงสุด	100 โวลต์
กระแสไฟฟ้าIdcสัมผัสสูงสุด	0.5 แอมแปร์
ความต้านทานสัมผัสแรกเริ่มสูงสุด	0.15 โอห์ม (แยกสายไฟตัวนำ)

TEMPERATURE

Operating	0 – 60	องศาเซลเซียส
Storage	-25-80	องศาเซลเซียส

RATINGS

แรงดึงแม่เหล็ก มากกว่าหรือเท่ากับ 29.4 หรือ 9.8 นิวตันเมื่อวัดโดยน้ำหนักมาตรฐาน TDK

2.5 อินฟราเรดเซนเซอร์ [9]

2.5.1 คุณสมบัติ

วงจรส่งไฟ 12 โวลต์ดีซีใช้กระแสประมาณ 20 มิลลิแอม วงจรรับใช้ไฟ 12 โวลต์ดีซี เมื่อรับสัญญาณได้จะใช้กระแสประมาณ 15 มิลลิแอม จุด OUT จะเป็น HIGHT เมื่อรับสัญญาณได้และเป็น LOW เมื่อ รับสัญญาณไม่ได้ สามารถปรับการหน่วงเวลาได้ตั้งแต่ 1 – 60 วินาที ระยะห่างในการตรวจจับประมาณ 11 เซนติเมตร สามารถปรับระดับความเร็วในการตรวจจับวัตถุได้โดยการปรับ VR1 สามารถปรับการหน่วงเวลาได้โดย VR2

2.5.2 การใช้งาน

ภาคส่ง ต่อไฟ 12 โวลต์เข้าให้ถูกขั้ว กินกระแสประมาณ 90 mA

ภาครับ - + - ต่อไฟ 12 โวลต์เข้า กินกระแสประมาณ 70 mA

- OUT เป็นจุดต่อที่ออกมาจากขาคอเล็กเตอร์เพื่อไปขับวงจรต่าง ๆ ที่ไม่ผ่านการหน่วงเวลา
- COM , NO , NC เป็นตัวคอนแทคของรีเลย์สามารถปรับการหน่วงเวลาให้ทำงานค้างได้อีก โดยปรับเวลาตั้งแต่ 1 – 10 วินาที ปรับที่ VR2 ซึ่งจุดนี้สามารถนำไปต่อกับชุดกันขโมย 4 โซน MAX09 นำไปต่อกับออดหรือหลอดไฟที่ได้เมื่อรีเลย์ดูดหน้าสัมผัส COM กับ NO ต่อกับ LED2 จะติดด้วย

2.5.3 การทำงานของวงจรตรวจจับด้วยอินฟราเรด

ภาคส่ง

มีส่วนประกอบสำคัญอยู่ 4 ส่วน คือ

1. ไอซีเร็คกูเรเตอร์เบอร์ LM7805 ทำหน้าที่ลดแรงดันจาก 12 โวลต์มาเป็น 5 โวลต์
2. วงจรผลิตความถี่ 500Hz
3. วงจรผลิตความถี่ 40 Hz จะทำงานโดยส่งความถี่ทั้ง2 สลับกันมายัง
4. ทรานซิสเตอร์ C9031 ทำหน้าที่ขยายความถี่ที่ส่งมาและส่งไปยัง
5. LED INF ทำหน้าที่ส่งอินฟราเรดไปยังภาครับ

ภาครับ

มีส่วนประกอบสำคัญอยู่ 3 ส่วนคือ

1. อุปกรณ์รับสัญญาณอินฟราเรด โดยจะให้เอาท์พุทออกมาเมื่อรับสัญญาณอินฟราเรดได้
2. ไอซีออปแอมป์เบอร์ LM324 ทำหน้าที่ขยายสัญญาณให้แรงขึ้นเพื่อควบคุมการสวิตช์ของ
3. ทรานซิสเตอร์ C458 ซึ่งจะทำหน้าที่ตัดต่อแรงดันให้กับรีเลย์ 12 โวลต์

จากทฤษฎีหลัก ๆ ของไมโครคอนโทรลเลอร์ ทำให้สามารถเขียนโปรแกรมควบคุมการทำงานของอุปกรณ์ต่าง ๆ และยังได้ศึกษาคุณสมบัติของอุปกรณ์แต่ละตัวไม่ว่าจะเป็น บอร์ดโทรศัพท์ บอร์ดบันทึกเสียง สวิตซ์แถบแม่เหล็ก และ อินฟราเรดเซนเซอร์ ว่ามีหลักการทำงานหรือมีหน้าที่อะไรในระบบ เพื่อที่จะนำข้อมูลต่าง ๆ เหล่านี้ไปใช้ในการออกแบบระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์ต่อไป

บทที่ 3
วิธีการออกแบบ

จากที่มาและหลักการต่างๆของการริเริ่มที่จะทำระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์ที่ได้มีการเชื่อมต่อกับไมโครคอนโทรลเลอร์ บอร์ดโทรศัพท์ บอร์ดบันทึกเสียง สวิตซ์แถบแม่เหล็ก และ อินฟราเรดเซนเซอร์แล้ว ในส่วนของบทนี้จะกล่าวถึงวิธีการออกแบบและ การจัดวางของอุปกรณ์ต่างๆโดยแบ่งออกเป็นภาคตรวจจับ ส่วนของเครื่องควบคุม และส่วนของภาคส่งสัญญาณเตือน ซึ่งรายละเอียดต่าง ๆจะแสดงไว้ดังนี้

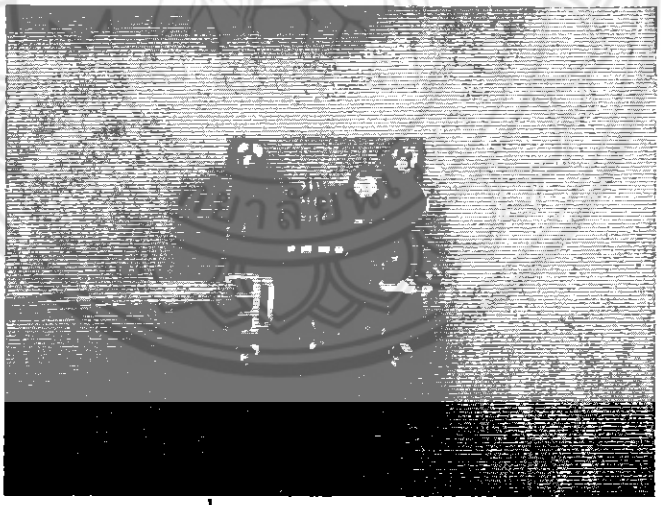
การออกแบบและจัดวางอุปกรณ์

อุปกรณ์ที่ใช้ในการทดลอง ประกอบไปด้วยส่วนต่าง ๆ 3 ส่วนด้วยกัน คือ

3.1 ส่วนของภาคตรวจจับสัญญาณ

ประกอบไปด้วย

- อินฟราเรดเซนเซอร์ มีระยะห่างในการตรวจจับประมาณ 11เซนติเมตร โดยมีหลักการทำงานคือเมื่อมีวัตถุหรือสิ่งมีชีวิตเคลื่อนที่ผ่านแนวอินฟราเรดก็จะเกิดสัญญาณอินพุท เพื่อส่งสัญญาณไปยังบอร์ดหลัก โดยตัวอย่างของอุปกรณ์จะแสดงดังรูปที่ 3.1



รูปที่ 3.1 อินฟราเรดเซนเซอร์

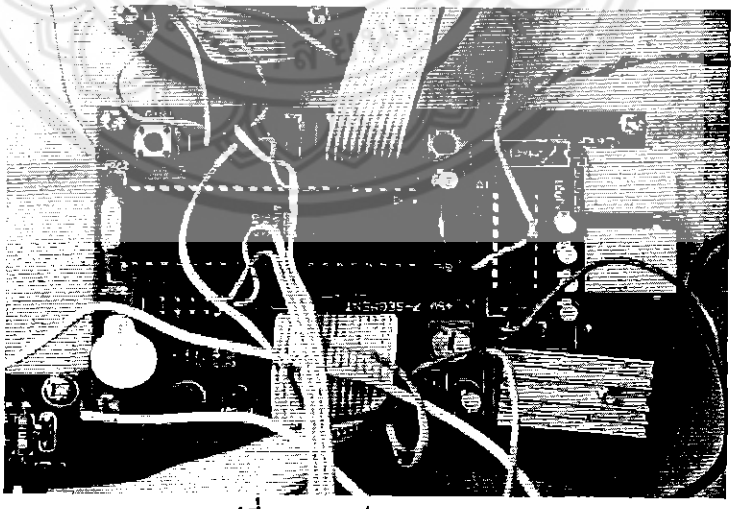
- สวิตซ์แถบแม่เหล็ก มีขนาดความกว้าง 2 เซนติเมตร และยาว 6 เซนติเมตร หลักการทำงานประกอบด้วย 2 ส่วนคือ ส่วนที่อยู่กับที่และส่วนที่เคลื่อนที่ เมื่อเกิดการเคลื่อนที่ออกจากกันก็จะเกิดสัญญาณอินพุต แล้วก็จะส่งสัญญาณที่ได้นั้นไปยังส่วนควบคุมให้ทำงานต่อ โดยตัวอย่างของอุปกรณ์จะแสดงดังรูปที่ 3.2

รูปที่ 3.2 สวิตซ์แถบแม่เหล็ก

3.2 ส่วนของเครื่องควบคุม

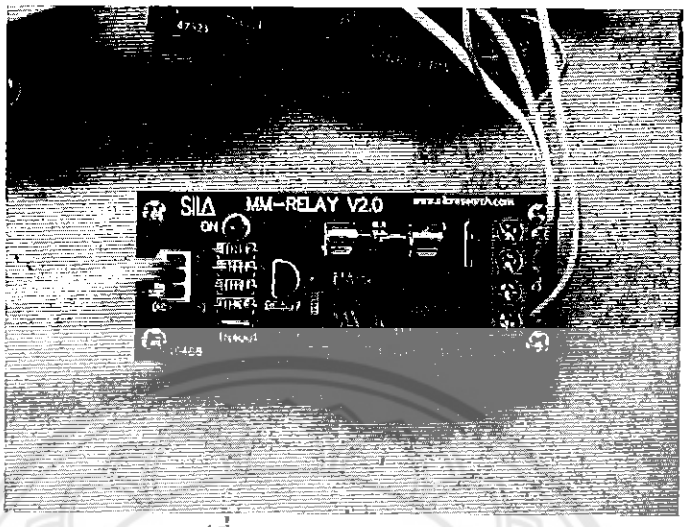
ประกอบด้วย

- ไมโครคอนโทรลเลอร์ตระกูล MCS-51 มีหลักการทำงานคือ เมื่อได้รับสัญญาณอินพุทที่มาจาก อินฟราเรดและ สวิตซ์แถบแม่เหล็ก MCS-51 จะทำการสั่งให้ไซเรนส่งเสียงดัง พร้อมกันนั้น ก็จะส่งสัญญาณเอาท์พุทออกไปยัง บอร์ดโทรศัพท์และบอร์ดอัดเสียง โดยที่จะมีปุ่ม Reset และ Interrupt ช่วยในการ set อุปกรณ์โดยตัวอย่างของอุปกรณ์จะแสดงดังรูปที่ 3.3



รูปที่ 3.3 บอร์ดควบคุม

- MM-RELAY ทำหน้าที่ในการต่อขยายบอร์ด ทำงานเป็นเอาท์พุทรีเลย์ ทำหน้าที่เป็นสวิทช์ติดต่อแรงดันได้ทั้งกระแสตรงและกระแสสลับ ควบคุมการทำงานเป็นแบบ ACTICE LOW โดยจะมีฟิวส์ 5 A ทำงานเป็นตัวช่วยป้องกันกระแสเกิน ตัวอย่างของอุปกรณ์จะแสดงดังรูปที่ 3.4

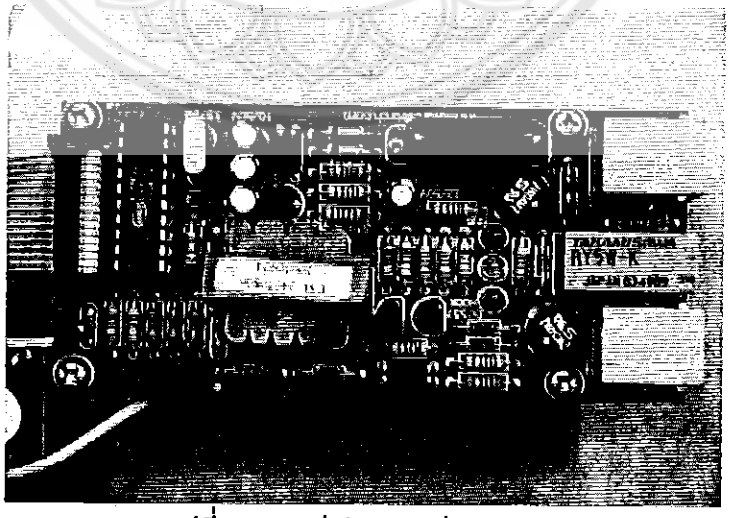


รูปที่ 3.4 MM-RELAY V2.0

3.3 ส่วนของภาคส่งสัญญาณเตือน

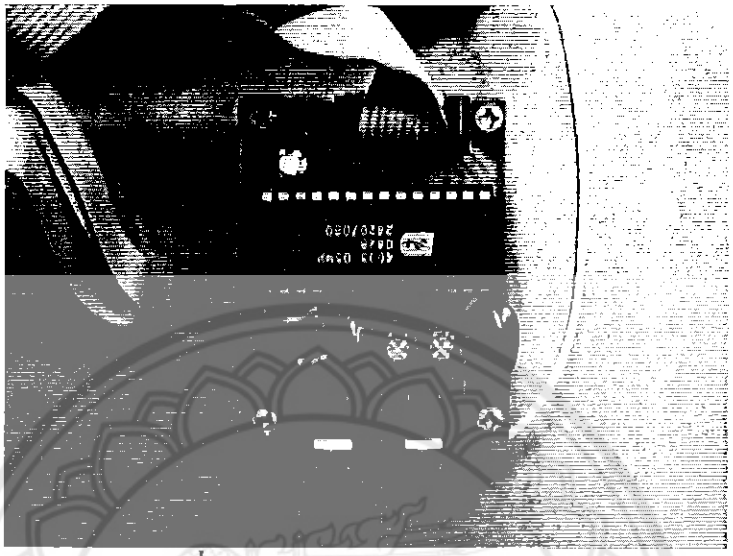
ประกอบด้วย

- บอร์ดโทรศัพท์ E12-SLC (Telephone line interface) ออกแบบให้ใช้งานร่วมกับ Controller Board สำหรับงานประยุกต์ที่ต้องควบคุมระบบโดยผ่านคู่สายโทรศัพท์ โดยมี hub switch ที่ใช้เป็นช่องเสียบสาย Line และ Tel ตัวอย่างของอุปกรณ์จะแสดงดังรูปที่ 3.5



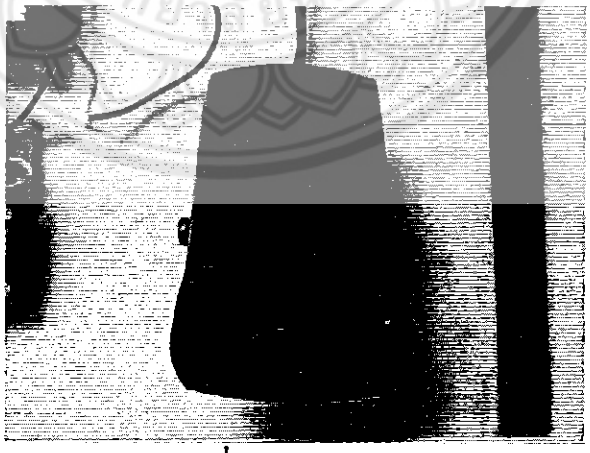
รูปที่ 3.5 บอร์ดโทรศัพท์ E12-SLC

- บอร์ดบันทึกเสียง ISD-4003 เป็นบอร์ดที่ทำหน้าที่บันทึกและเล่นเสียง (Record/play)โดยขั้นตอนการบันทึกเสียง สามารถทำได้โดยการ เสียบต่อเข้ากับลำโพงทางช่อง Line out และเสียบกับคอมพิวเตอร์ทางช่อง Line in สามารถบันทึกซ้ำได้มากกว่า 100,000 ครั้ง และจดจำได้นานถึง 100 ปี โดยตัวอย่างของอุปกรณ์จะแสดงดังรูปที่ 3.6



รูปที่ 3.6 บอร์ดบันทึกเสียง ISD-4003

- ไซเรน จะส่งเสียงดังเป็นเวลานานกี่วินาที ขึ้นอยู่กับการกำหนดค่าในการโปรแกรม และการส่งเสียงดังจะเริ่มต้นเมื่อได้รับกระแสอินพุตเข้ามา ตัวไซเรนเองจะขนาดความกว้างประมาณ 5 เซนติเมตร และยาวประมาณ 8 เซนติเมตร การติดตั้งจะมีขาตั้ง และตัวยึดทำให้สามารถนำไปติดกับฝาได้ โดยตัวอย่างของอุปกรณ์จะแสดงดังรูปที่ 3.7



รูปที่ 3.7 ไซเรน

ขั้นตอนออกแบบและการจัดวางอุปกรณ์ถือว่ามีความสำคัญไม่น้อยเพราะว่า อุปกรณ์แต่ละตัวก็มีหน้าที่และความสำคัญที่แตกต่างกันไป และทุกตัวก็ต้องทำงานให้สัมพันธ์กันไม่ว่าจะเป็นอุปกรณ์ใน

ส่วนของภาคตรวจจับสัญญาณ ได้แก่ อินฟราเรดเซนเซอร์ และ สวิตซ์แถบแม่เหล็ก ส่วนของเครื่องควบคุม ได้แก่ ไมโครคอนโทรลเลอร์และรีเลย์ ส่วนของภาคส่งสัญญาณเตือน ได้แก่ บอร์ดโทรศัพท์ บอร์ดบันทึกเสียง และไซเรน

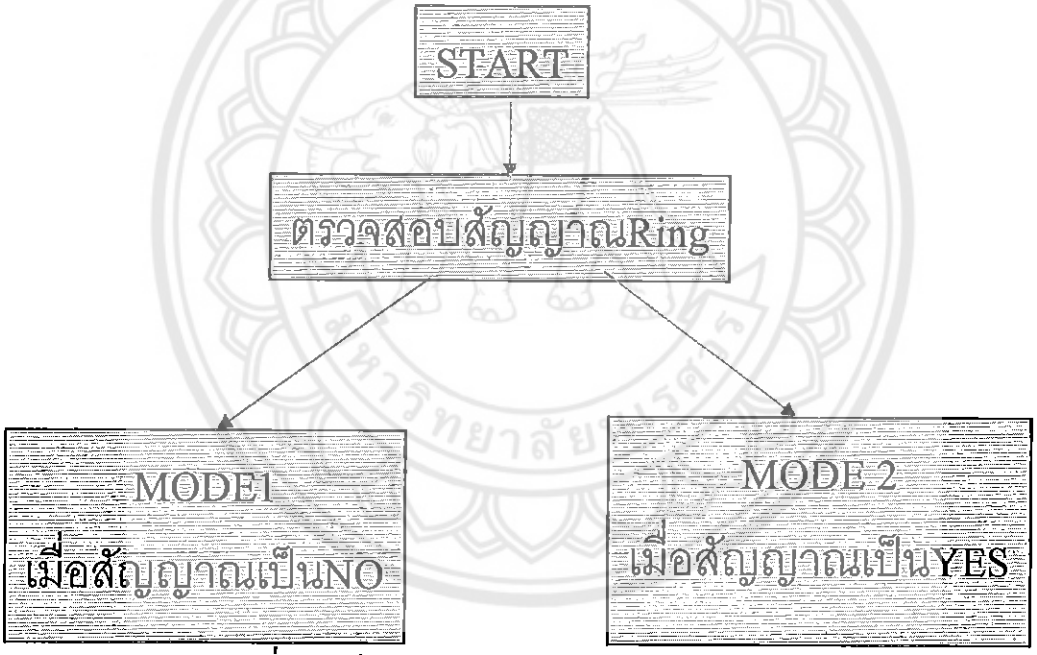
บทที่ 4

ผลการทดลอง

จากขั้นตอนการออกแบบและจัดวางอุปกรณ์แต่ละตัว ของระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์ ทำให้เราทราบถึงหลักการ และสามารถดำเนินการทดลองเพื่อให้ทราบผลและประสิทธิภาพของระบบ โดยการทำงานของระบบได้แบ่งออกเป็นโหมด ๆ ซึ่งจะแสดงรายละเอียดให้ทราบ ดังนี้

4.1 ระบบการทำงานโดยรวม

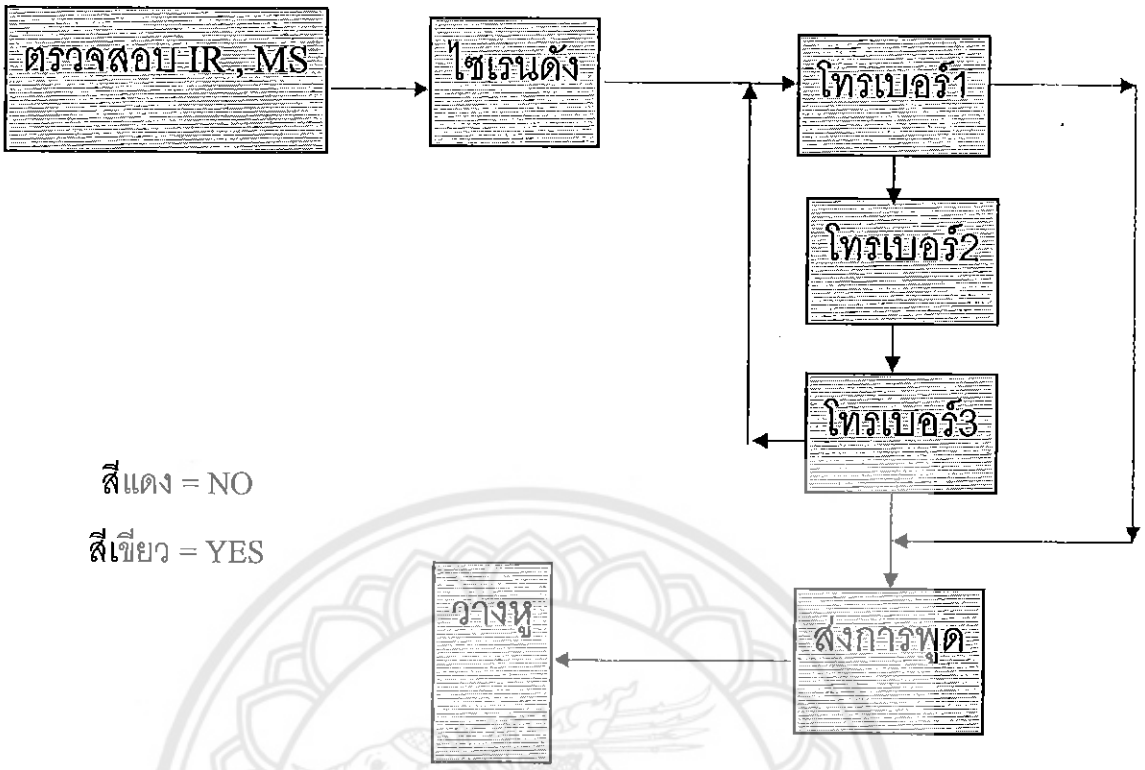
ระบบการทำงานโดยรวมได้แบ่งออกเป็น 2 โหมดโดยเมื่อมีการเริ่มทำงานระบบจะทำการตรวจสอบสัญญาณ RING ว่ามีสัญญาณ Ring เข้ามาหรือไม่ หากมีระบบจะแยกเป็นโหมดที่ 2 และหากไม่มีสัญญาณระบบจะทำงานตามปกติคือโหมดที่1 ดังรูปที่ 4.1



รูปที่ 4.1 บล็อกไดอะแกรมระบบการทำงานแบ่งเป็นโหมด

4.1.1 การทำงานในโหมดที่1

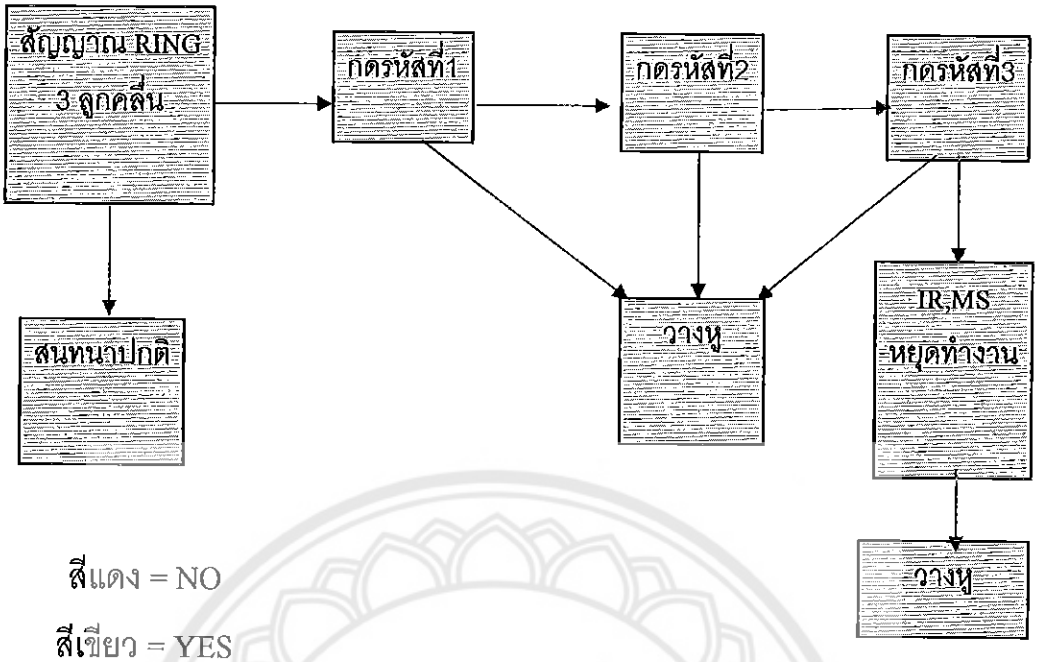
เมื่อทำการตรวจสอบสัญญาณจาก สวิตซ์แถบแม่เหล็ก และ อินฟราเรดแล้ว ระบบส่งสัญญาณทำให้ไซเรนดัง และสั่งให้บอร์ดโทรศัพท์โทรไปยังหมายเลขโทรศัพท์ที่ได้ทำการ set ไว้ทั้ง 3 เบอร์ หากเบอร์ที่ 1 ไม่มีการตอบรับก็จะส่งไปยังเบอร์ที่2 และ 3 หากยังไม่มีการรับสายจะวนกลับมายังเลขหมายที่ 1ใหม่ หรือหากมีการตอบรับจากเบอร์ใดเบอร์หนึ่งแล้วระบบก็จะส่งสัญญาณเสียงตอบรับแล้วก็จะทำการวางหูโทรศัพท์ ดังรูปที่ 4.2



รูปที่ 4.2 บล็อกไดอะแกรมการทำงาน MODE 1

4.1.2 การทำงานในโหมดที่ 2

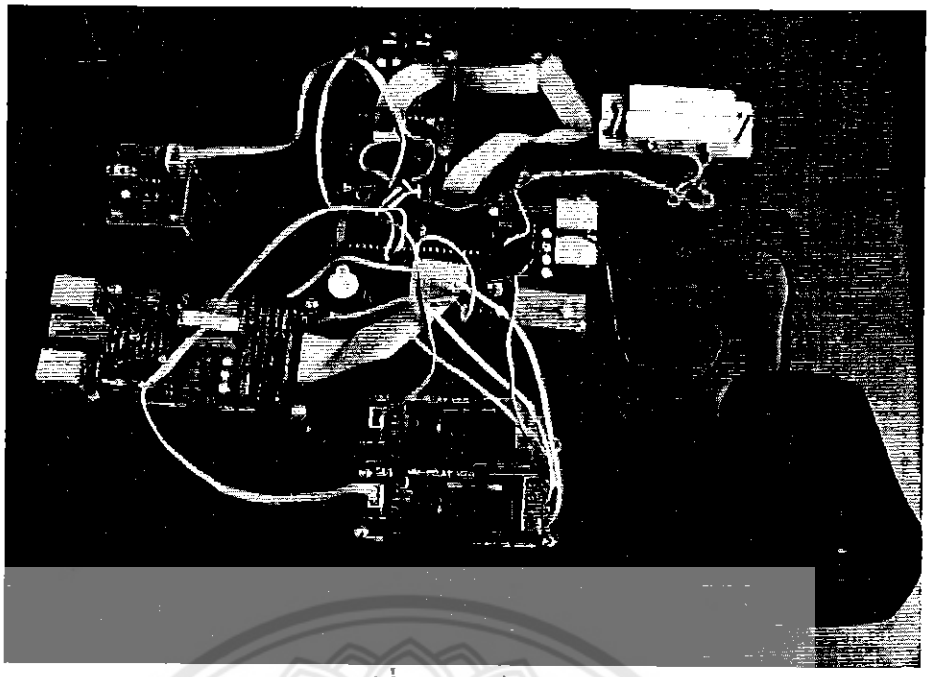
เมื่อสัญญาณ Ring ดังขึ้น เราได้ setไว้ให้เป็น 3 ลูกคลื่นคือมีเสียงสัญญาณ 3 ครั้ง หากมีการรับสายก่อนหน้านั้น ก็จะเป็นการสนทนาตามปกติ แต่หากรับสายหลังสัญญาณครั้งที่3 ระบบก็สั่งให้กดรหัส 3 ตัว ตามที่ได้ set ไว้ หากรหัสถูกต้องก็จะมีการสั่งงานให้อินฟราเรดและสวิตซ์แถบแม่เหล็กหยุดทำงาน แล้วก็ให้วางหู แต่หากมีการกดรหัสตัวใดตัวหนึ่งผิด ระบบก็จะสั่งให้วางหูทันที ดังรูปที่ 4.3



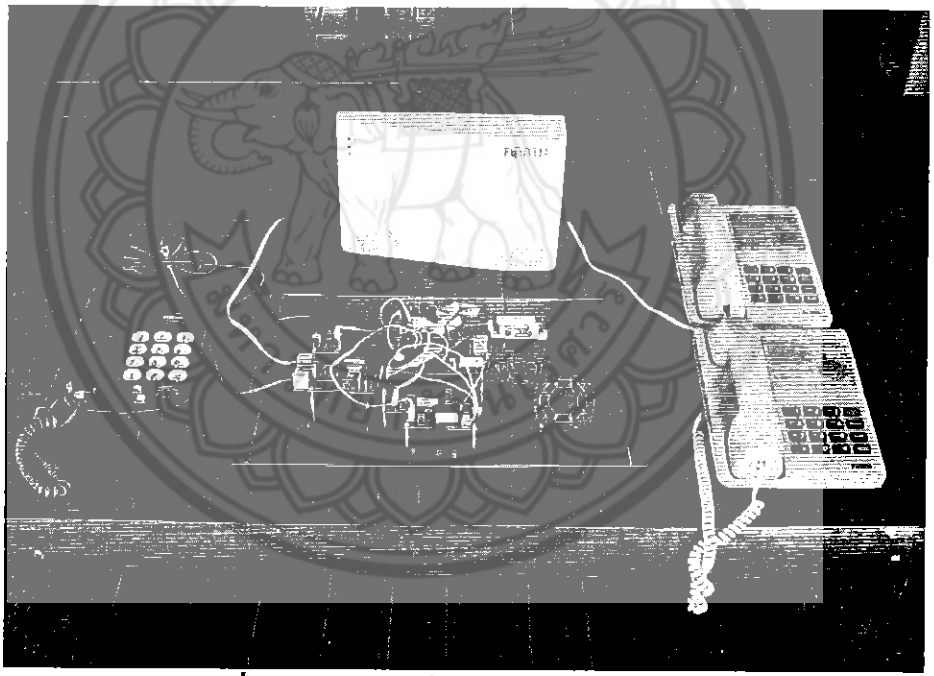
รูปที่ 4.3 บล็อกไดอะแกรมการทำงาน MODE 2

4.2 ผลการทดลอง

จากขั้นตอนต่าง ๆ ที่เราได้ทำการเชื่อมต่ออุปกรณ์หลักแต่ละส่วนเข้าด้วยกันจนได้เป็นส่วนของบอร์ดหลัก และในนำเอาไปเชื่อมต่อกับอุปกรณ์ภายนอกคือ (PABX) และ โทรศัพท์ โดยที่ส่วนของบอร์ดหลักจะแสดงดังรูปที่ 4.4 และส่วนการเชื่อมต่อกับอุปกรณ์ภายนอกจะแสดงดังรูปที่ 4.5



รูปที่ 4.4 บอร์ดหลัก



รูปที่ 4.5 แสดงการเชื่อมต่อกับอุปกรณ์ภายนอก

ผลการทดลอง

เริ่มแรกเมื่อมีการเปิดระบบ ระบบก็จะทำงานเป็นแบบวนลูป และมีการทำงานแบ่งเป็นโหมดใหญ่ๆ 2 โหมด คือกรณีที่มีสัญญาณ Ring เข้ามา ระบบก็จะทำการตรวจเช็คว่ามีการรับสายก่อนสัญญาณ Ring ครั้งที่ 3 หรือไม่ หากใช่ก็จะสั่งให้เป็นการสนทนาตามปกติ หรือหากรอหลังสัญญาณครั้ง

ที่ 3 แล้ว ระบบจะสั่งกดรหัส 3 ตัวเพื่อยืนยันการปิดระบบ แต่หากมีการกดรหัสตัวใดตัวหนึ่งผิดระบบก็จะทำการวางสาย ทำให้ต้องเริ่มทำตามขั้นตอนแรกอีกครั้ง ส่วนการทำงานอีกส่วนคือ เมื่อไม่มีสัญญาณ Ring เข้ามา ระบบก็จะทำงานเป็นภาคตรวจจับตามปกติโดยใช้มีสวิตซ์แถบแม่เหล็กและเครื่องตรวจจับอินฟราเรด หากมีสัญญาณอินพุตเข้าอุปกรณ์ตัวใดตัวหนึ่ง ระบบก็จะส่งเอาต์พุตไปยังส่วนควบคุมแล้วไซเรนจะดัง และบอร์ดโทรศัพท์จะทำการโทรไปยังเลขหมาย 3 เลขหมาย เรียงจาก 1 ไป 3 หากไม่มีการรับสายจากเลขหมายที่ 3ระบบจะโทรวนกลับไปยังเบอร์แรก แต่หากมีการรับสายจากเลขหมายใดๆแล้วระบบก็จะส่งสัญญาณเสียงตอบรับเพื่อเตือนว่ามีผู้บุกรุกทันที

บทที่ 5
การสรุปผลและการวิเคราะห์ปัญหาในการทดลอง

5.1 สรุปผลการทดลอง

จากผลการทดลองระบบสัญญาณเตือนภัยผ่านเครือข่ายโทรศัพท์ ทำให้ทราบผลการทำงานของเครื่องมือและอุปกรณ์ต่าง ๆ ที่ทำงานอย่างสัมพันธ์กัน และผลการทำงานก็เป็นที่น่าพอใจเป็นไปตามที่ตั้งค่าไว้ ซึ่งผลของการทดลองก็คือ ระบบได้มีการแบ่งการทำงานออกเป็นโหมด ๆ ซึ่งโหมดแรกเป็นกรณีที่มีผู้บุกรุกเมื่อไม่มีสัญญาณ Ring เข้ามา ระบบจะทำงานเป็นภาคตรวจจับโดยรอรับสัญญาณอินพุทจากอุปกรณ์ตัวใดตัวหนึ่ง หมายถึง เมื่อมีผู้บุกรุกไซเรนก็จะส่งเสียงดังเป็นเวลานาน 5 วินาที (ขึ้นอยู่กับค่าที่เราได้ตั้งไว้ในโปรแกรม) พร้อมกันนั้น ระบบจะทำการโทรศัพท์ไปยังเลขหมายที่ได้ทำการตั้งไว้ 3 เลขหมาย โดยเรียงจากเลขหมายที่ 1 ไปยัง 2 และ 3 ตามลำดับ หากยังไม่มีการรับสายก็จะทำการวนกลับไปยังเลขหมายที่ 1 หรือหากมีการรับสายจากเลขหมายใด ๆ แล้ว ระบบก็จะส่งสัญญาณเสียงตอบรับ เพื่อเตือนให้ทราบว่ามีผู้บุกรุก ส่วนโหมดที่ 2 เป็นกรณีที่เจ้าของบ้านต้องการจะปิดระบบเพื่อที่จะเข้าบ้าน สามารถทำได้โดยการโทรเข้าและรอให้สัญญาณ Ring ดัง 3 ครั้ง หลังจากนั้นจะได้ยินเสียงสัญญาณให้กดรหัส 3 ตัว เพื่อที่จะยืนยัน ถ้าหากมีการกดรหัสตัวใดตัวหนึ่งผิด ระบบก็จะทำการวางสาย เพื่อให้ทำการโทรเข้าใหม่อีกครั้ง

5.2 ปัญหาที่พบในการทดลอง

5.2.1 ปัญหาที่เกิดจากตัวอุปกรณ์

- อุปกรณ์บางตัวไม่มีจำหน่ายที่จังหวัดพิษณุโลกต้องสั่งซื้อที่กรุงเทพฯ หากเกิดความเสียหายจากการขนส่ง ทำให้อุปกรณ์ในส่วนนั้น ๆ ขัดข้อง จึงทำให้การดำเนินงานล่าช้า

วิธีแก้ไข สั่งซื้อทางไปรษณีย์ หรือ ไปซื้อที่กรุงเทพ ฯ

- ความเสียหายที่เกิดจากการบัดกรีให้ความร้อนมากเกินไป

วิธีแก้ไข อาจใช้ซ็อกเก็ตเข้ามารองขา แทนการบัดกรีโดยตรง

- อุปกรณ์ขาดความแข็งแรง เมื่อโดนแรงกระแทกอาจทำให้การทำงานไม่เป็นปกติ

วิธีแก้ไข นำอุปกรณ์ทั้งหมดมาติดตั้งไว้ในกล่องเหล็ก

- เมื่อต้องการจะทดสอบระบบ เกิดความลำบากในการเชื่อมต่ออุปกรณ์ต่างๆ เข้าด้วยกัน

วิธีแก้ไข ทำการขอยืมตู้สาขามา เพื่อความสะดวกในการทดสอบ

5.2.2 ปัญหาจากการเขียนโปรแกรม

- การเปลี่ยนแปลงค่าต่างในโปรแกรม ๆ เช่น Password หมายเลขโทรศัพท์ ยังไม่สามารถเปลี่ยนแปลงเองได้ ต้องเปลี่ยนแปลงจากตัวโปรแกรมผ่านการ Burn เท่านั้น

- อุปกรณ์ในแต่ละประเทศใช้ความถี่ที่ไม่ตรงกัน เมื่อนำมาใช้งานร่วมกัน อาจทำให้ระบบทำงานผิดปกติได้ จึงจำเป็นที่จะต้องใช้อุปกรณ์ที่มีความถี่ที่เหมือนกันเท่านั้น

5.3 แนวคิดในการพัฒนาต่อ

- เพื่อเพิ่มความสะดวกและง่ายต่อผู้ใช้งาน ผู้จัดทำคิดที่จะเชื่อมต่อจอ LCD และปุ่มกดหมายเลข เพื่อใช้ในการป้อนและแก้ไข Password หมายเลขโทรศัพท์ โดยไม่ต้องมาทำการโปรแกรมซ้ำใหม่อีกครั้ง

เอกสารอ้างอิง

[1] รศ. ธีรวัฒน์ ประกอบผล . **การประยุกต์ใช้งาน ไมโครคอนโทรลเลอร์ .** ครั้งที่ 8. กรุงเทพ ฯ : บริษัท ดวงกมลสมัย จำกัด: 2547.

[2]Adisak Chinawong ."**Special function register.**"[Online]Available: www.202.8.85.164/~adisak51/SFR.html./2000.

[3] MRT "**Micro Research Technology.**"[Online]Available: www.micro-research.co.th/.2005.

[4] รศ. ธีรวัฒน์ ประกอบผล . **ภาษาแอสเซมบลี สำหรับ MCS-51.** ครั้งที่ 2 . กรุงเทพ ฯ:สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น). 2547.

[5] สุนทร วิทสุรพจน์ . **การโปรแกรมภาษาแอสเซมบลี ของ ไมโครคอนโทรลเลอร์ ตระกูล 8051.** กรุงเทพ ฯ: บริษัท เอช-เอนจำกัด . 2537.

[6] MRT "**Micro Research Technology.**"[Online]Available: www.micro-research.co.th/.2005.

[7] บริษัทศิลารีเสิร์ช จำกัด "**Telephone Line Interface.**"[Online]Available: www.silaresearch.com/2005.

[8] Electronix express ."**Magnetic Switch.**"[Online]Available: www.elexp.com/cmp_agsw.htm/.2004.

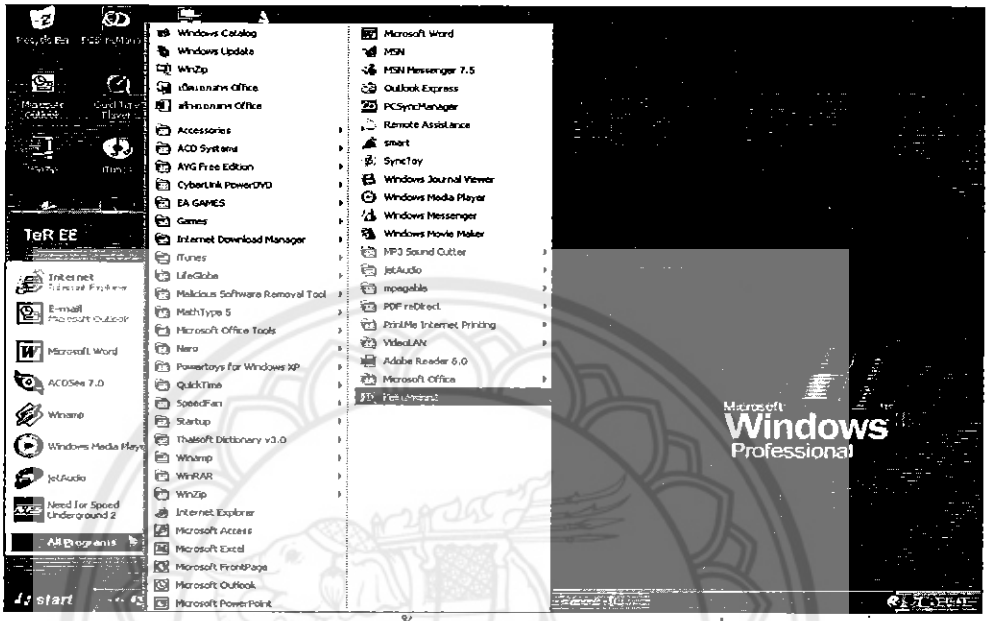
[9] Micro Activities "**เซนเซอร์ตรวจจับแสงอินฟราเรด.**"[Online]Available: www.mut.ac.th/~c_micro/workshop/acct3/acct3_ir1ss.html./2005.

ภาคผนวก ก

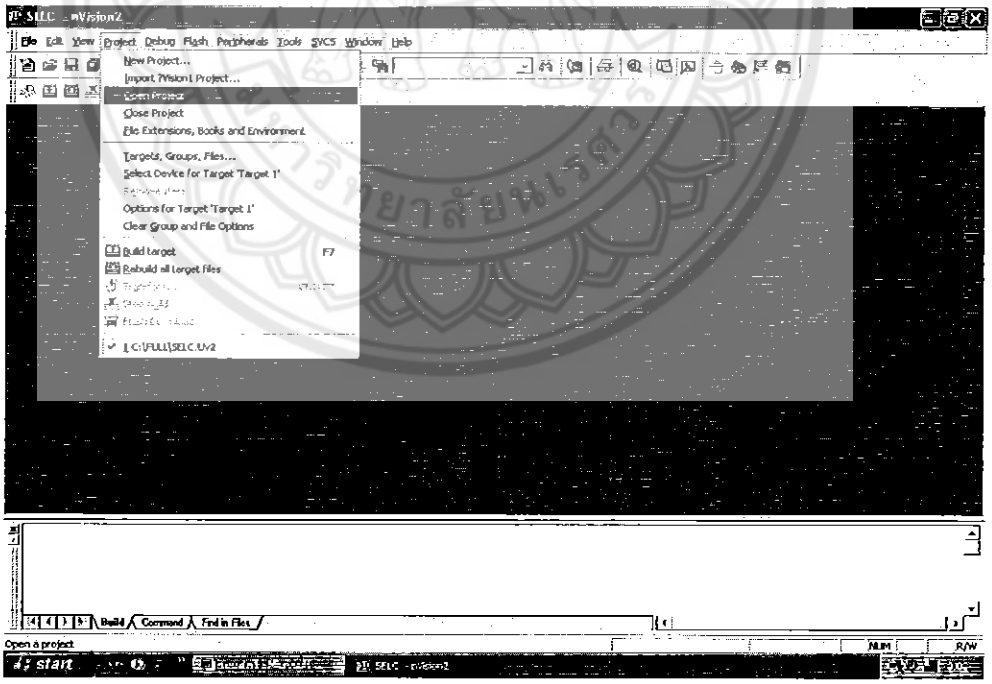
การ Burn โปรแกรม

การ Burn โปรแกรมมีขั้นตอนดังต่อไปนี้

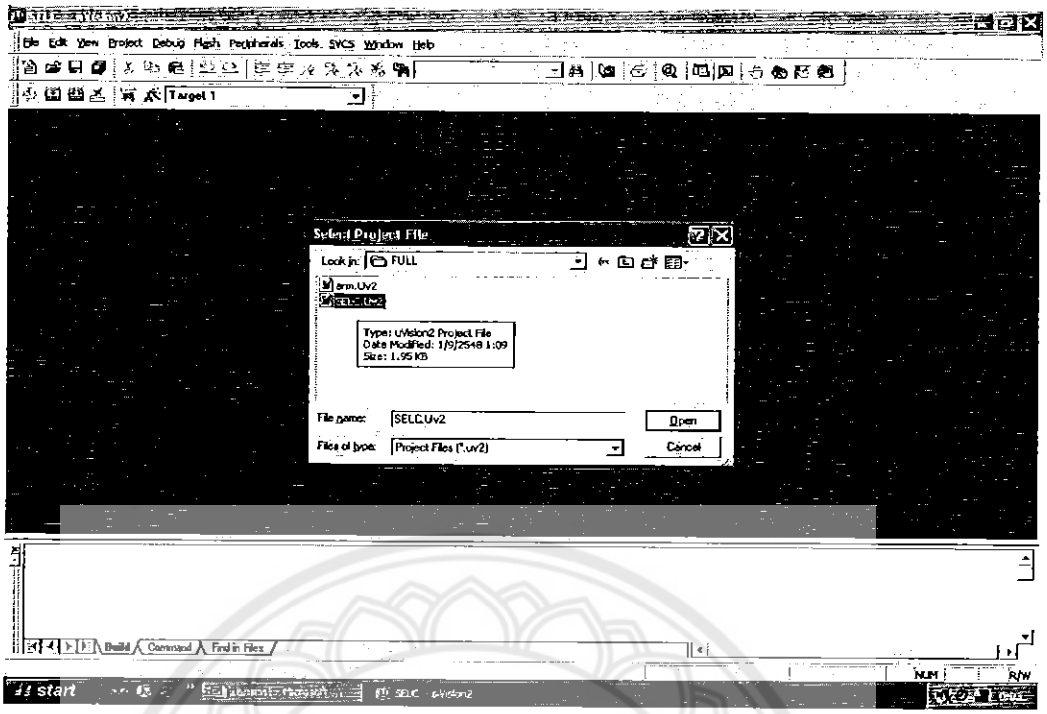
1. เปิดเครื่องจากนั้นทำการ Run โปรแกรมโดยการเลือกที่ Start / All Program / Keil uVision2



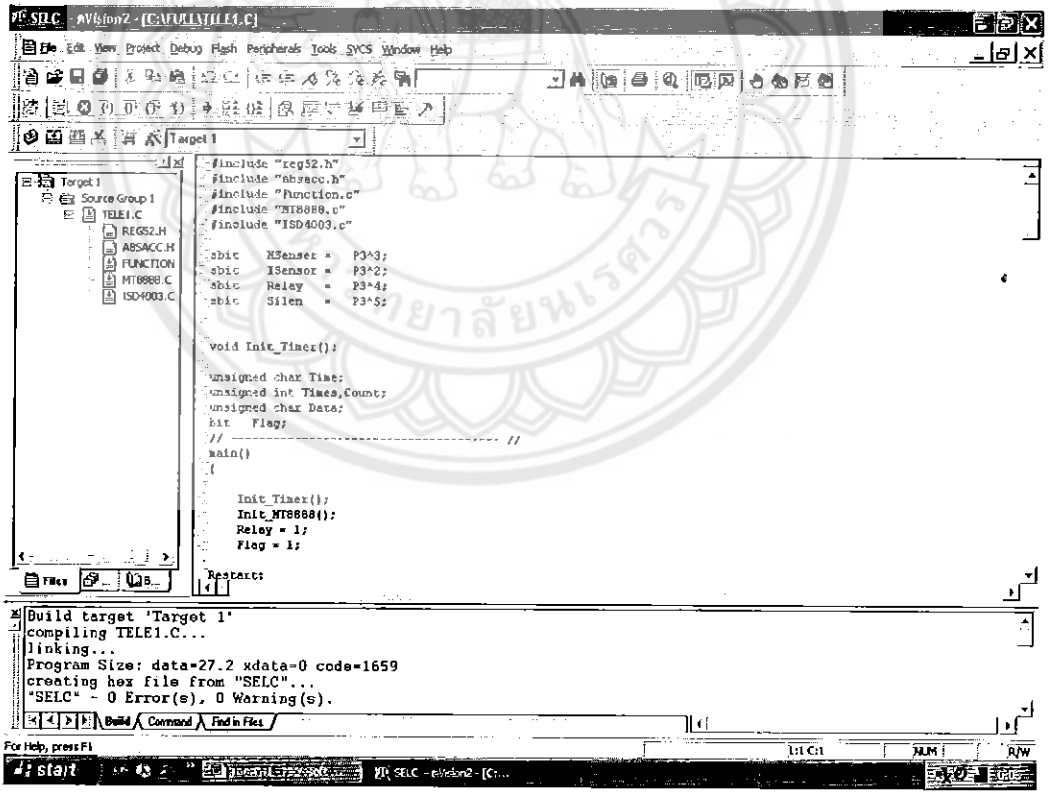
2. หน้าจอจะแสดงโปรแกรมดังรูปข้างล่างนี้ ให้เปิด Project โดยเข้าไปที่เมนู Project / Open Project



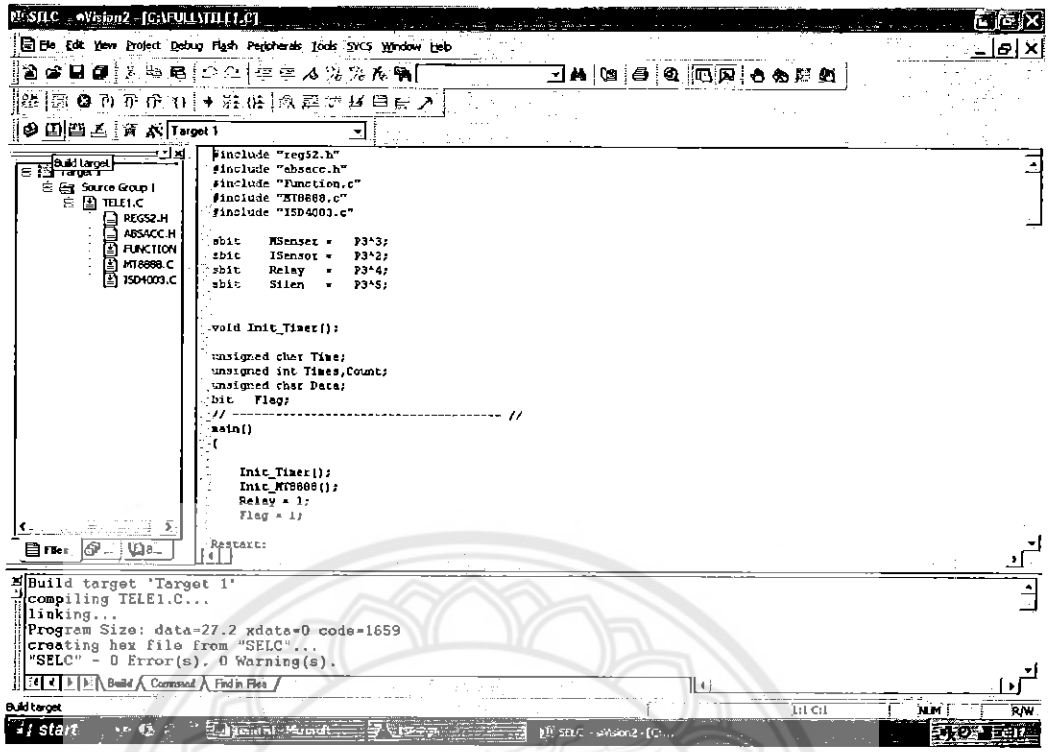
3. หน้าจอจะปรากฏหน้าต่างให้เราเลือก Project ที่เราได้สร้างไว้ แล้วคลิ๊กปุ่ม Open



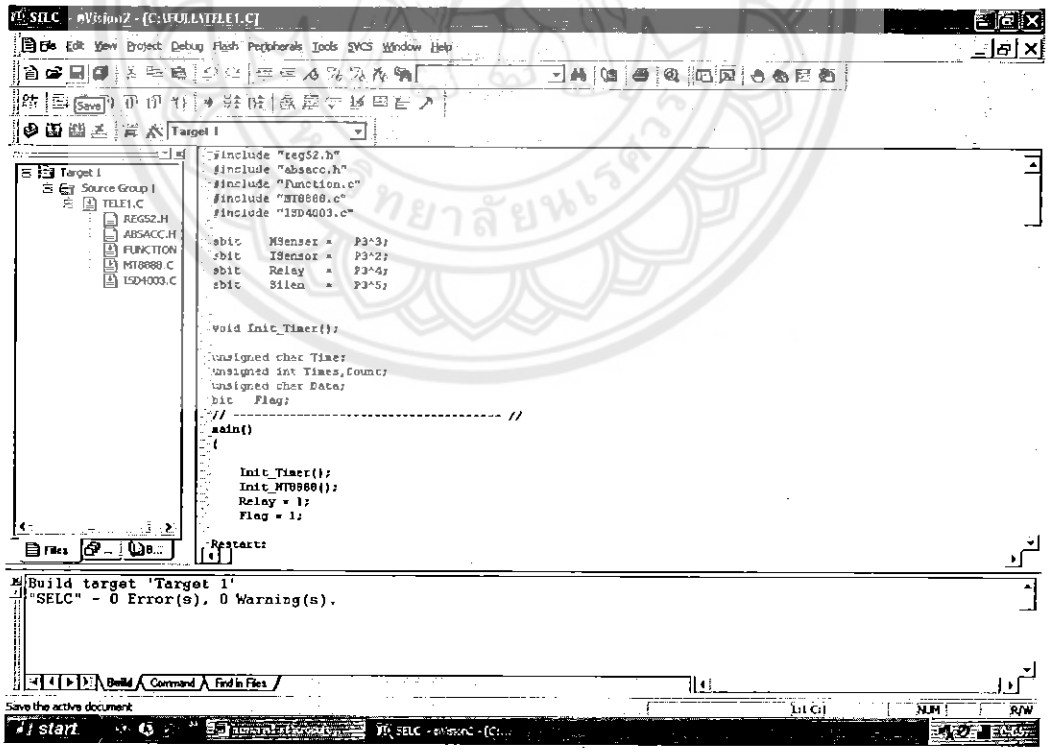
4. จากนั้นหน้าจอจะแสดงดังรูปข้างล่าง



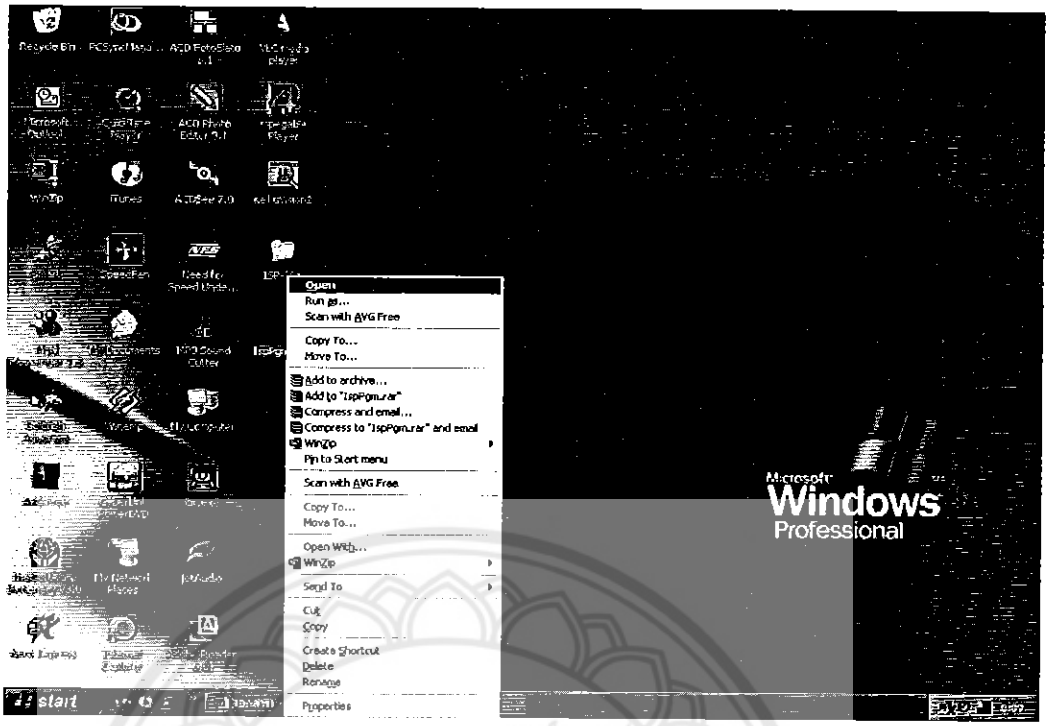
5. ตรวจสอบโปรแกรมว่ามีข้อผิดพลาดหรือไม่โดยเลือกเมนูที่ Project / Build target



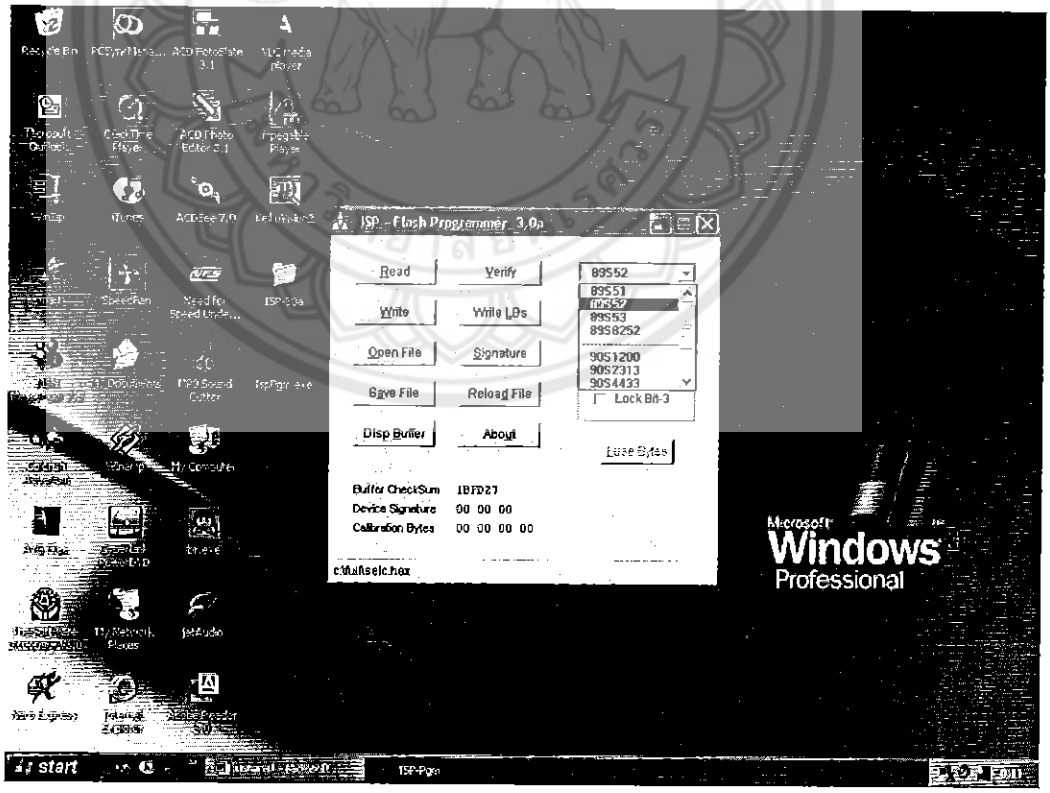
6. เมื่อทำการ Build target แล้วไม่มีข้อผิดพลาดเกิดขึ้น ให้เราทำการบันทึกไฟล์โดยไปที่เมนู File / New(การตั้งชื่อต้องใส่นามสกุล .C ด้วย)



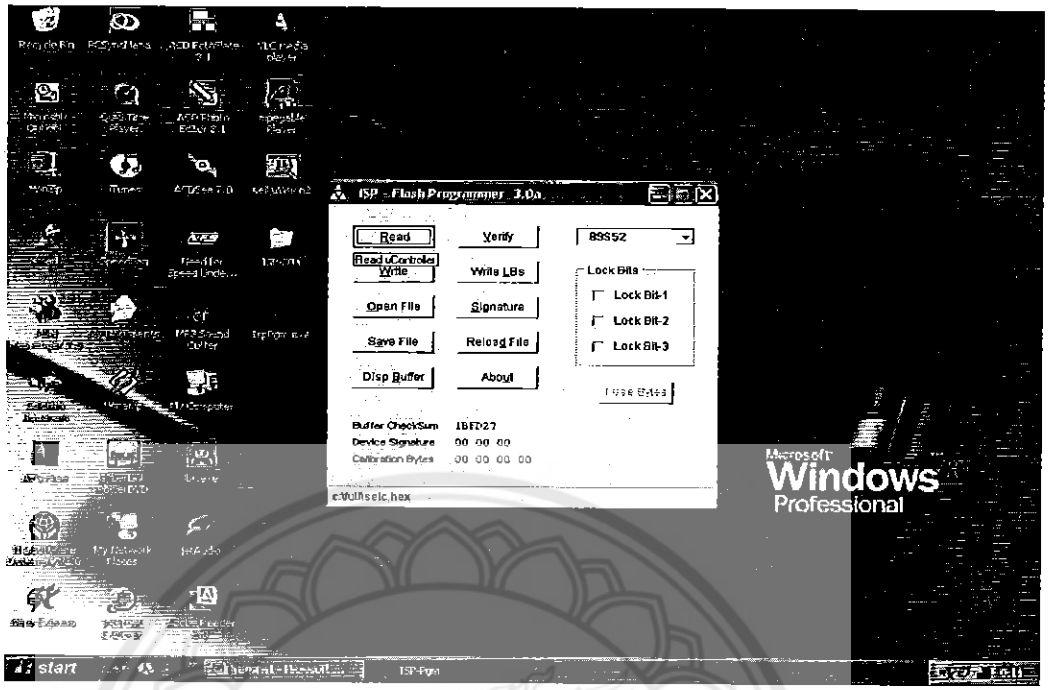
7.จากนั้นที่หน้าจอ Destop ให้เปิดโปรแกรม Isp Program โดยคลิ๊กขวาที่เมาท์แล้วคลิ๊กที่ Open



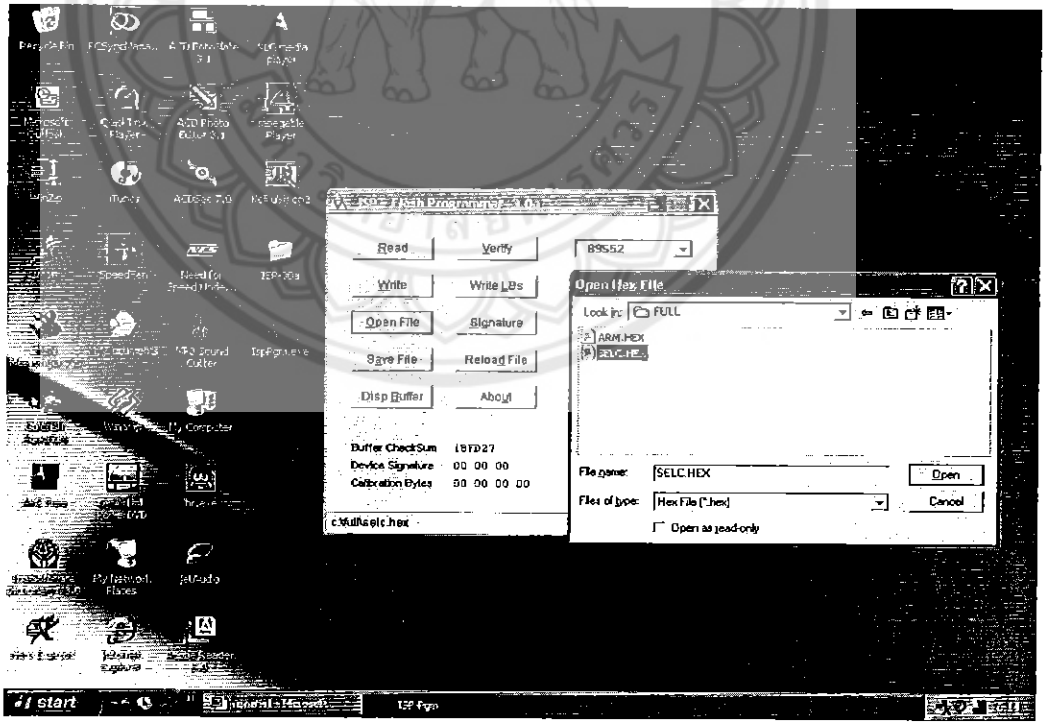
8. จากนั้นให้เราทำการเลือกบริษัทและเบอร์ไมโครคอนโทรลเลอร์ ที่เราใช้ตามรูปข้างล่างคือรุ่น 89S52



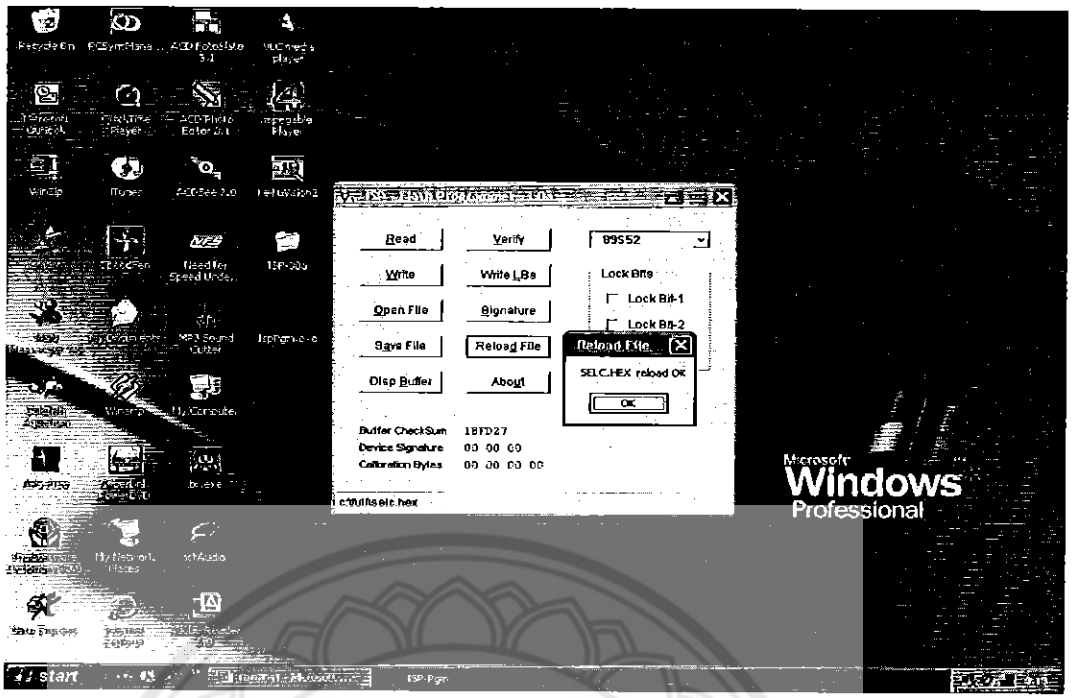
9. เมื่อเชื่อมต่อไมโครคอนโทรเลอร์เข้ากับพอร์ทขนานของคอมพิวเตอร์แล้ว ให้เข้าไปที่ Read แล้วคลิ๊ก 1 ครั้ง โปรแกรมจะทำการอ่านค่าพื้นฐานของไมโครคอนโทรเลอร์



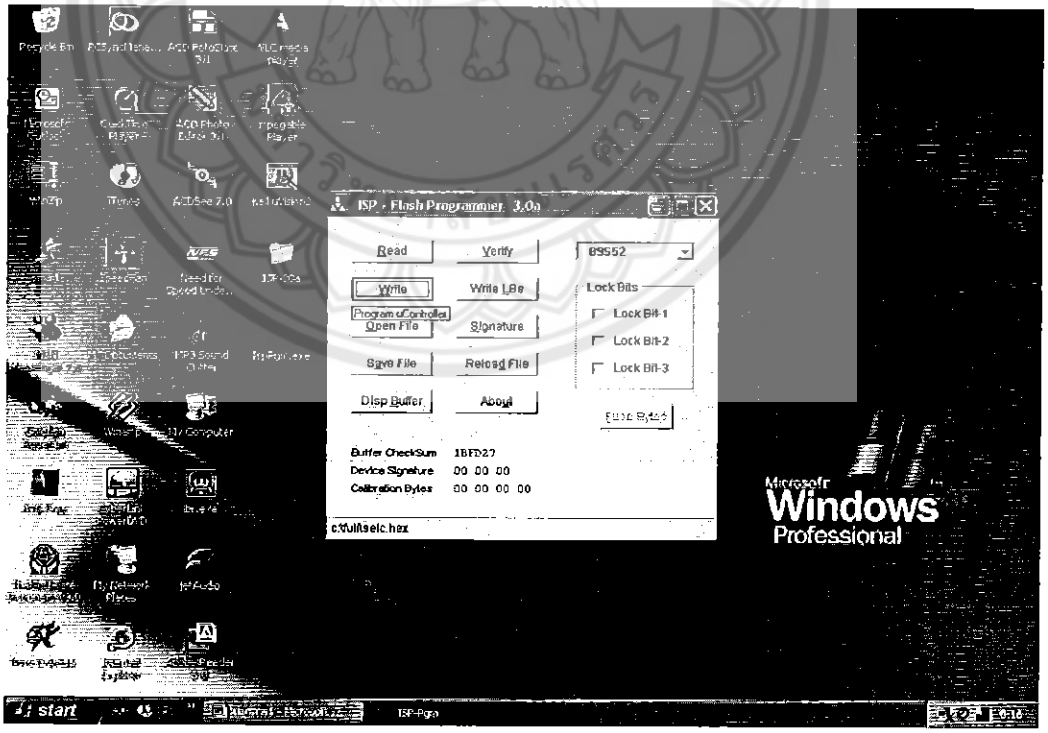
10. จากนั้นให้คลิ๊กที่ Open File จะได้หน้าต่างดังรูป แล้วให้เราเลือกไฟล์ที่เราได้ทำการบันทึกไว้แล้วทำการเปิดไฟล์คลิ๊ก Open



11.คลิ๊กที่ Reload File 1 ครั้ง โปรแกรมจะขึ้นหน้าต่างขึ้นมาเพื่อต้องการยืนยันให้เรากดปุ่ม OK



12.ขั้นตอนสุดท้าย คลิ๊กที่ Write 1 ครั้ง รอจนกว่าการเบิร์นจะเสร็จสมบูรณ์ (แถบสีน้ำเงินโหลดจนเต็ม)แล้วก็กดปุ่ม OK



ภาคผนวก ข

Source code ของโปรแกรม

```
stmt level    source
    1         #include "reg52.h"
    2         #include "absacc.h"
    3         #include "Function.c"
    4         #include "MT8888.c"
    5         #include "ISD4003.c"
    6
    7         sbit   MSenser =     P3^3;
    8         sbit   ISensor =  P3^2;
    9         sbit   Relay   =     P3^4;
   10         sbit   Silen   =  P3^5;
   11
   12
   13         void Init_Timer();
   14
   15
   16         unsigned int Times,Count;
   17         unsigned char Data;
   18         bit   Flag;
   19         // ------------------------------------ //
   20         main()
   21         {unsigned char Time;
   22    1
   23    1        Init_Timer();
   24    1           Init_MT8888();
   25    1        Relay = 1;
   26    1        Flag = 1;
```

```
27  1
28  1   Restart:
29  1     while(1)
30  1     {
31  2      if(Ring==0)
32  2        {
33  3         Rx_Mode();
34  3         CheckRing(2);
35  3         Hook_On();
36  3         Tx_Mode();          /* Tx_Mode for sent DTMF */
37  3              Delay(200);
38  3              WR_DTMF(0);   /*if active send DTMF 0 */
39  3              Delay(200);
40  3              WR_DTMF(0);
41  3              Delay(200);
42  3              Rx_Mode();                  /* Rx_Mode for received DTMF */
43  3
44  3        if(Pickup == 0)
45  3        {
46  4
47  4        while((RD_Status()&0x04)!=0x04){}
48  4          Data     =     RD_DTMF();
49  4        if(Data == 7)
50  4          {
51  5           while((RD_Status()&0x04)!=0x04){}
52  5              Data     =     RD_DTMF();
53  5           if(Data == 5)
54  5           {
55  6            while((RD_Status()&0x04)!=0x04){}
56  6               Data    =     RD_DTMF();
57  6            if(Data == 3)
58  6             {
```

```
59  7                   Flag = 0;
60  7                 }
61  6
62  6


63  6               }
64  5
65  5             }
66  4
67  4             Hook_Off();
68  4           }
69  3
70  3         }else
71  2       if(((MSenser == 1)||(ISensor ==0))&&(Flag==1))
72  2         {
73  3           goto  Alarm;
74  3
75  3         }
76  2
77  2       }
78  1   Alarm:
79  1
80  1         Silen  =  0;
81  1       Times  =  0;
82  1         Delay(200);
83  1
84  1
85  1
86  1   Start:
87  1
88  1         Hook_On();
89  1         Delay(200);
```

```
 90  1          Tx_Mode();              /*----Tx Mode-----*/
 91  1          Delay(200);
 92  1          WR_DTMF(4);             /*----Call No.13 Via PABX-------*/
 93  1          Delay(200);
 94  1          WR_DTMF(3);
 95  1          Delay(200);
 96  1          WR_DTMF(6);
 97  1          Delay(200);
 98  1          WR_DTMF(8);
 99  1          /*Check Destination PickUp*/
100  1          /*0 = PickUp*/
101  1          /* 1-2 = Dial Tone */
102  1          /* 4 = Busy Tone */
103  1          Time = 0;
104  1          while((Check_Tone() != 0)&& (Time < 2))
105  1           {
106  2            Time++;
107  2           }
108  1
109  1          if(Time >= 2)
110  1           {
111  2          Hook_Off();
112  2               Delay(1000);
113  2          Hook_On();
114  2            Delay(200);
115  2            Tx_Mode();
116  2            Delay(200);
117  2            WR_DTMF(4);
118  2            Delay(200);
119  2            WR_DTMF(3);
120  2            Delay(200);
121  2            WR_DTMF(7);
```

```
122  2            Delay(200);
123  2            WR_DTMF(1);
124  2          Time = 0;
125  2          while ((Check_Tone() != 0)&&(Time < 2))
126  2           {
127  3           Time++;
128  3           }
129  2         if(Time >= 2)
130  2           {
131  3         Hook_Off();Delay(1000);
132  3         Hook_On();
133  3           Delay(200);
134  3           Tx_Mode();
135  3           Delay(200);
136  3           WR_DTMF(4);
137  3           Delay(200);
138  3           WR_DTMF(3);
139  3           Delay(200);
140  3            WR_DTMF(6);
141  3               Delay(200);
142  3               WR_DTMF(8);
143  3         Time = 0;
144  3          while ((Check_Tone() != 0)&&(Time < 2))
145  3           {
146  4            Time++;
147  4           }
148  3           if(Time >= 2)
149  3             { Hook_Off();Delay(1000);
150  4              goto Start;
151  4             }
152  3           }
153  2
```

```
154  2            }
155  1
156  1
157  1          /* Send Sound When Distination Pick-up Phone */
158  1          Delay(200);
159  1        Relay = 0;
160  1          Player(3,40);
161  1        Player(3,40);
162  1        Relay = 1;
163  1        Hook_Off();
164  1
165  1          goto Restart;
166  1
167  1    }
168       //----------------------//
169       void Init_Timer()
170       { TMOD = 0x01;
171  1      ET0  = 1;
172  1      EA   = 1;
173  1      TR0  = 1;
174  1      TH0  = 0x10;
175  1      TL0  = 0x00;
176  1    }
177
178       void Timer0_Int(void) interrupt 1
179       {
180  1      TH0 = 0x10;
181  1      TL0 = 0x00;
182  1      Count++;
183  1      if(Count == 15)
184  1      {
185  2       Count = 0;
```

```
186  2          Times++;
187  2          if(Times == 5){ Silen = 1;}
188  2       }
189  1    }
190
```

ประวัติผู้เขียนโครงงาน

ชื่อ นายชวาล ตาคำ

ภูมิลำเนา 89 หมู่ 6 ต.วรณคร อ.ปัว จ.น่าน 55120

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนปัว
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail : **nayarm1199@hotmail.com**

ชื่อ นายเอกพล ผดุง

ภูมิลำเนา 201 หมู่ 1 ต.ฝายแก้ว กิ่งอ.ภูเพียง จ.น่าน 55000

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนสตรีศรีน่าน
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail : **aekapol_padung@hotmail.com**