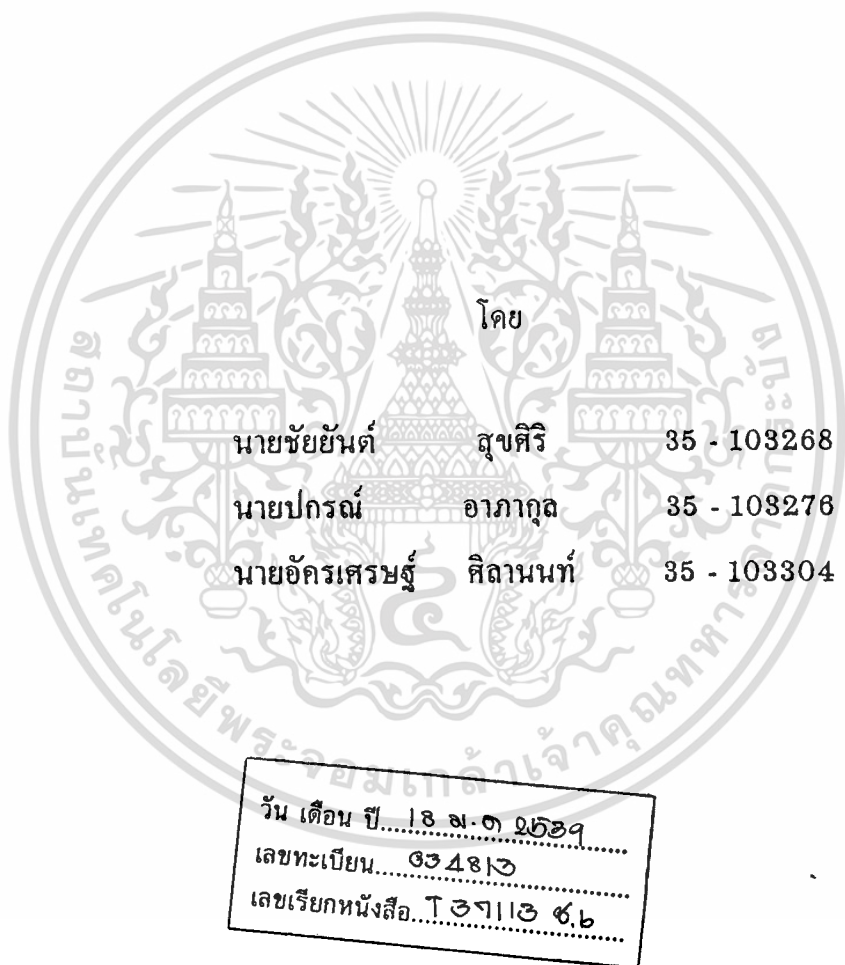




เครื่องหยอดของเหลวโปรแกรมได้

PROGRAMMABLE LIQUID DISPENSER



โดย

นายชัยยันต์

สุขศิริ

35 - 103268

นายปรกรณ์

อากาศ

35 - 103276

นายจักรเศรษฐ์

ศิลาพนธ์

35 - 103304

วัน เดือน ปี 18 พ.ค ๒๕๖๑

เลขทะเบียน ๐๖481๖

เลขเรียกหนังสือ T ๖๖1.๖ ๕.๖

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมการวัดคุม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษา 2537 อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ปริญญาโทปีการศึกษา 2537

ภาควิชา เทคโนโลยีการวัดคุมทางอุตสาหกรรม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง เครื่องหยอดของเหลวโปรแกรมได้ (PROGRAMABLE LIQUID DISPENSER)

ผู้จัดทำ

1. นาย ชัยยันต์ สุขศิริ
2. นาย ปกรณ์ อากาศ
3. นาย อัครเศรษฐ์ ศิลานนท์


(อ.ศ.สุวิทย์ งามคง)

อาจารย์ที่ปรึกษา

เครื่องหยอดของเหลวโปรแกรมได้

ชัยยันต์ สุขศิริ

ปกรณ์ อากาศ

อัศวเศรษฐ์ ศิลานนท์

ผศ.สุพรรณ กุลพานิชย์ อาจารย์ที่ปรึกษา

ปีการศึกษา 2537

บทคัดย่อ

โครงการนี้นำเสนอการใช้ไมโครคอนโทรลเลอร์ MCS-51 เบอร์ 8031 ในการควบคุมระบบนิวแมติกส์ให้มีความละเอียดเพื่อนำไปใช้ในการควบคุมการหยอดของเหลวในงานอิเล็กทรอนิกส์ หรืองานอุตสาหกรรมต่าง ๆ โดยมีส่วนประกอบของโครงงานหลักแบ่งได้ดังนี้

โครงงานนี้ ประกอบด้วย 2 ส่วนใหญ่ ๆ คือ

1. ระบบนิวแมติกส์ (Neumatic system)
2. ระบบควบคุม (Controller system)

ระบบนิวแมติกส์

ส่วนของระบบนิวแมติกส์ประกอบด้วยอุปกรณ์ต่าง ๆ เช่น Regulator Gage Solinoid Valve ซึ่งอุปกรณ์หลักของส่วนนี้ ก็คือ Solinoid Valve 3/2 ทำหน้าที่เปิดปิดลมที่จะเข้าไปดันของเหลวในหลอดฉีดของเหลว

ระบบควบคุม

ส่วนของระบบควบคุมแบ่งออกเป็น 2 ส่วน คือ

1. ส่วน Hard ware ซึ่งจาก Block Diagram จะเห็นได้ว่ามีส่วนของ บอร์ดควบคุม ซึ่งเป็นบอร์ดสำเร็จรูป ชื่อ CP-SB 31 โดยใช้ไมโครคอนโทรลเลอร์ MCS-51เบอร์ 8031 เป็นตัวหลักในการควบคุม ซึ่งรายละเอียดของบอร์ดได้อธิบายไว้ในหัวข้อที่ 2.3 บอร์ดควบคุมอินเตอร์เฟซกับคีย์บอร์ดเพื่อใช้ในการป้อนโปรแกรมค่าเวลาต่าง ๆ และกำหนดฟังก์ชันการทำงานของเครื่องอีกส่วนหนึ่งที่อินเตอร์เฟซกับบอร์ดควบคุมก็คือ จอ LCD เพื่อสำหรับใช้ในการแสดงผลการทำงานในสถานะต่าง ๆ ของเครื่อง

2. ส่วน Soft Ware เป็นโปรแกรมที่เขียนด้วยภาษาแอสเซมบลี โดยเขียนโปรแกรมให้ Hard ware ทำหน้าที่เป็น timer และมีรายละเอียดของฟังก์ชันต่าง ๆ ดังที่จะแสดงไว้ในส่วนของวิธีการใช้เครื่อง ฯ

PROGRAMABLE LIQUID DISPENSER

CHAIYAN SUKSIRI

PAKORN APAKUN

AKARASAD SILANON

SUPUN KULPANIT ADVISOR

1994

Abstract

This thesis is presentation how to use microcontroller MCS-51 No 8031 to control pneumatic system which control liquid dropping in other industrials.

The contains of thesis is follow:

This thesis contains of 2 parts

1. pneumatic system
2. controller system

Pneumatic system

pneumatic system contains of many equipment such as Regulator, Guage, Solinoid Valve. The main equipment is Solinoid Valve 3/2. It used to close or open air to push liquid in tube.

Controller system

Controller system can be devided into 2 parts

1. Hardware part, from block diagram, a part of finished control board CP-SB 31 is controlled by microcontroller MCS-51 No. 8031 . In explain at 2.3 items, interphase control board and keyboard used to program times. LCD moniter is one to inter phase to control board. Itused to show running in other conditions.

2. Software parts is programed by assembly language. It order to hard-ware to be timer and other function that shown .

คำนำ

ในปัจจุบันงานทางด้านวิศวกรรมการควบคุมโดยทั่วไปจะใช้ ไมโครคอนโทรลเลอร์ในการควบคุมการทำงานของเครื่องจักรต่างๆ ในโรงงานอุตสาหกรรม ซึ่งสามารถทำให้เครื่องจักรทำงานได้อย่างถูกต้องและมีประสิทธิภาพ ดังนั้นโครงงานนี้จึงถูกออกแบบโดยใช้ไมโครคอนโทรลเลอร์ตระกูล MCS-51 เบอร์ 8031 มาทำการควบคุมระบบนิวแมติกส์ให้สามารถทำงานได้ละเอียด และ แม่นยำ ซึ่งโครงงานนี้มีชื่อว่า เครื่องหยอดของเหลวโปรแกรมได้ มีหลักการคือ การควบคุมปริมาณลมที่จะเข้าไปคั่นของเหลวในหลอดหยอดของเหลว เพื่อให้ได้ขนาดหยดของเหลวตามต้องการ โดยการควบคุมเวลาการเปิดปิดของวาล์วลม ซึ่งใช้ไมโครคอนโทรลเลอร์เป็นตัวควบคุมเวลา, แสดงสถานะการทำงานออกทางจอ LCD และรับการโปรแกรมจากคีย์บอร์ด

สารบัญ

เรื่อง	หน้า
บทที่ 1	
- ความสำคัญและที่มาของโครงการ	1-1
- วัตถุประสงค์ของโครงการ	1-2
- ประโยชน์ที่คาดว่าจะได้รับ	1-3
บทที่ 2 ทฤษฎีและหลักการ	2-1
- 2.1 พื้นฐานทั่วไปของไมโครคอนโทรลเลอร์ เบอร์ 8031	2-1
- 2.2 IC 8255	2-29
- 2.3 รายละเอียดเกี่ยวกับบอร์ดที่ใช้ในโครงการ	2-43
บทที่ 3 รายละเอียดเกี่ยวกับโครงการ	3-1
- วิธีการใช้เครื่อง ฯ	3-5
บทที่ 4	
- การทดลองการหยอดของเหลว	4-1
- สรุปผลการทดลอง	4-3
บทสรุป	
ภาคผนวก	
กิตติกรรมประกาศ	
หนังสืออ้างอิง	

บทที่ 1

1.1 ความสำคัญและที่มาของโครงการ

ในปัจจุบัน บริษัทอิเล็กทรอนิกส์และโรงงานอุตสาหกรรมต่างๆ โดยทั่วไปในสายงานการผลิตมีการหยอดของเหลวบนงานในลักษณะต่างๆ กัน ซึ่งของเหลวที่หยอดก็มีหลายประเภท เช่น กาว, น้ำมันซิลิโคน, สารหล่อลื่น, อีพอกซี, ครีมนักกรี จึงมีผู้ผลิตเครื่องหยอดของเหลวเพื่ออำนวยความสะดวก ในสายงานการผลิต ทำให้งานการผลิตรวดเร็วขึ้น คุณภาพดีขึ้น ลดของเสียให้น้อยลง

แต่เครื่องหยอดของเหลวที่มีขายอยู่ในท้องตลาดโดยทั่วไป ก็มีข้อจำกัดขอบเขตการทำงาน คือ เครื่องหยอดของเหลวที่มีขายในท้องตลาด สามารถกำหนดค่าเวลาของการเปิดปิดโซลินอยด์วาล์วหรือกล่าวอีกนัยหนึ่งก็คือ การกำหนดขนาดของของเหลวที่หยอดได้เพียงขนาดเดียวต่อการตั้งค่าเวลาแต่ละครั้ง แต่งานในสายงานการผลิตจริงนั้นบางครั้งมีขนาดของจุดที่จะหยอดของเหลวแตกต่างกันมากมายหลายขนาดและหลายจุด ซึ่งถ้าเป็นเครื่องหยอดของเหลวที่มีขายอยู่มีท้องตลาดโดยทั่วไปจะมีความยุ่งยากในการทำงานมาก ก็ต้องคอยเปลี่ยนค่าเวลาของการเปิดปิดของโซลินอยด์ วาล์วบ่อยครั้งในการผลิตชิ้นงานแต่ละชิ้นซึ่งทำให้เป็นการเสียเวลาด้วยและที่สำคัญอีกประการหนึ่ง ก็คือ ขนาดของของเหลวที่หยอดอาจไม่ได้มาตรฐานที่กำหนดเนื่องจากการที่พนักงานเป็นผู้ปรับแต่ง ค่าเวลาเอง

ดังนั้น คณะผู้จัดทำโครงการจึงได้เสนอโครงการที่ชื่อว่า เครื่องหยอดของเหลวโปรแกรมได้หรือในชื่อภาษาอังกฤษว่า PROGRAMMABLE LIQUID DISPENSER ขึ้นเพื่อตอบสนองความต้องการของบริษัทอิเล็กทรอนิกส์และโรงงานอุตสาหกรรมทั่วไปเช่น บริษัท ซีเกท เทคโนโลยี ประเทศไทย จำกัด ฯลฯ

2. วัตถุประสงค์ของโครงการ

1.2.1 เพื่อสามารถออกแบบและสร้าง LIQUID DISPENSER ที่โปรแกรมได้เพื่อตอบสนองความต้องการของบริษัทอิเล็กทรอนิกส์และโรงงานอุตสาหกรรมโดยทั่วไป

1.2.2 เพื่อผสมผสานเทคโนโลยีทางด้านวิศวกรรมอิเล็กทรอนิกส์ และเทคโนโลยีทางด้านวิศวกรรมเครื่องกลโดยการนำเอาระบบอิเล็กทรอนิกส์มาใช้ในการควบคุมการทำงานของระบบนิวแมติกส์ ให้เกิดความละเอียดแม่นยำมากขึ้น

1.2.3 เพื่อลดค่าใช้จ่ายของบริษัท ฯ และลดการสูญเสียเงินตราในการนำเข้าสินค้าจากต่างประเทศ

3. ประโยชน์ที่คาดว่าจะได้รับ

- 1.3.1 สามารถนำไมโครคอนโทรลเลอร์ไปใช้ควบคุมการทำงานของอุปกรณ์ภายนอก
- 1.3.2 เป็นแนวทางของการศึกษา เพื่อนำไมโครคอนโทรลเลอร์ไปใช้งานในด้านต่างๆ
- 1.3.3 รู้เทคนิคในการเขียนโปรแกรมควบคุมการทำงานของไมโครคอนโทรลเลอร์
- 1.3.4 รู้จักการเชื่อมต่อ (Interface) ของไมโครคอนโทรลเลอร์กับ Hard ware
- 1.3.5 เป็นการพัฒนาศักยภาพในประเทศให้มีความสามารถในการวิจัยและพัฒนา

ผลด้านอื่น ๆ ได้

บทที่ 2

ทฤษฎีและหลักการ

ในการที่จะควบคุมการทำงาน ของระบบนิวแมติกส์ ให้มีความละเอียดมาก ๆ จำเป็นต้องนำเอาระบบไมโครโปรเซสเซอร์ มาใช้ในการควบคุมซึ่งในอดีตก็คือ Z-80 ที่เป็นที่ยอมรับกันโดยทั่วไปแต่ในปัจจุบันได้มีบริษัทผู้ผลิต ซิงเกิลชิพ โดยให้ชื่อว่าไมโครคอนโทรลเลอร์ซึ่งมี CPU PORT ต่าง ๆ และ Timer/Counter ใน ชิพเดียวกัน ซึ่งเป็นการสะดวกในการประยุกต์ใช้งานแต่ในโครงการที่สร้างนี้จำเป็นต้องInterfac กับอุปกรณ์ภายนอกหลายอย่างเช่นKey board จอ LCD และในขณะเดียวกันไมโครคอนโทรลเลอร์ ก็ต้องใช้พอร์ทในการติดต่อกับROM และ RAM ภายนอก เนื่องจากไมโครคอนโทรลเลอร์ที่ใช้ไม่มี ROM ในตัวและหน่วยความจำของ RAM ก็น้อยเกินไป ดังนั้นวงจรควบคุมของโครงการนี้จึงไม่สามารถใช้ไมโครคอนโทรลเลอร์เพียงชิพเดียว เพื่อให้โครงการสมบูรณ์ได้ จึงได้มีการต่อขยายพอร์ทในการควบคุมให้มากขึ้นโดยใช้ IC 8255(โปรแกรมเมเบิลเพอร์เฟอร์ม็อนเตอร์เฟสซิง)ทำเป็นพอร์ทเพิ่มขึ้นมาเพื่ออินเตอร์เฟสกับ KEY Board, จอLCD และเป็นพอร์ท I/O ซึ่งทฤษฎี และหลักการของบอร์ดที่ใช้ในการทำโครงการนี้ จะอธิบายไว้ในส่วนของเรื่องราวละเอียดของบอร์ดที่ใช้ในโครงการ และส่วนของSoft ware ของโครงการก็จะอธิบายในส่วนต่อไป

2.1 พื้นฐานทั่วไปของไมโครคอนโทรลเลอร์

เบอร์ 8031

คุณสมบัติของไมโครคอนโทรลเลอร์ เบอร์ 8031

8031 เป็นไมโครคอนโทรลเลอร์ (Microcontroller) ตัวหนึ่งในตระกูล MCS-51 ซึ่งไม่มีหน่วยความจำชนิดรอม (ROM) อยู่ในตัว จะมีคุณสมบัติทั่ว ๆ ไป ดังนี้

- ใช้ HMOS เทคโนโลยี ทำงานด้วยแหล่งจ่ายไฟขนาด +5 โวลต์
 - เป็นหน่วยประมวลผลกลาง (CPU) ขนาด 8 บิต
 - มีวงจรออสซิลเลเตอร์ (Oscillator) และวงจรนาฬิกา (clock) ในตัว
 - มีแบริจิสเตอร์ 4 ชุด โดยมีชุดละ 8 รีจิสเตอร์
 - สามารถอ้างแอดเดรสของหน่วยความจำโปรแกรมได้ถึง 64 กิโลไบต์
 - สามารถอ้างแอดเดรสของหน่วยความจำข้อมูลได้ถึง 64 กิโลไบต์
 - มีแรม (RAM) บนชิพ 128 ไบต์
 - มีพอร์ทอินพุตเอาต์พุต (I/O) เป็นแบบสองทิศทาง 4 พอร์ท พอร์ทละ 8 บิตนอกจากนี้ เราสามารถที่จะกำหนดพอร์ทของ 8031 ให้เป็นพอร์ทที่มีขนาด 1 บิตได้ ทำให้เรามีพอร์ทแบบ 1 บิตในการใช้งานได้ถึง 32 พอร์ท
 - พอร์ทอนุกรมสามารถที่จะโปรแกรมการรับ/ส่งแบบ Full Duplex ที่ความเร็วสูง
 - มีไทม์เมอร์/เคานท์เตอร์ขนาด 16 บิต จำนวน 2 ตัว
 - มีซอฟต์แวร์แฟลก (Flag software) สำหรับผู้ใช้ที่จะกำหนดเองได้ถึง 128 ตำแหน่งบิต
 - โครงสร้างการอินเตอร์รัพต์ (Interrupt) ทำได้ 5 แหล่ง พร้อมด้วยการจัดไพอริตี (Priority) ได้ 2 ระดับ
 - ตัวไมโครคอนโทรลเลอร์ สามารถใช้งานแบบบูลีนได้ สำหรับการใช้งานควบคุม
 - มีคำสั่งคูณและหารฮาร์ดแวร์ทำได้ภายใน 4 ไมโครวินาที (สำหรับคริสตอล 12 - MHz)
 - ตัวเลขทางคณิตศาสตร์ ใช้ได้ทั้งแบบเลขฐานสอง และฐานสิบ
 - การใช้หน่วยความจำพื้นที่สแต็ก (stack) สำหรับโปรแกรมย่อยต่างๆ ทำได้กว้าง
- ขึ้น
- สามารถเข้าถึงข้อมูลได้ทั้งแบบเป็นไบต์และแบบเป็นบิต
 - ชุดคำสั่งจะมีถึง 256 คำสั่ง

เบอร์	หน่วยความจำภายใน		ตั้งเวลา/นับ	อินเตอร์รัพต์ หมายเลข
	โปรแกรม	ข้อมูล		
8052AH	8k x 8 ROM	256 x 8 RAM	3 x 16 Bit	6
8051AH	4k x 8 ROM	128 x 8 RAM	2 x 16 Bit	5
8051	4k x 8 ROM	128 x 8 RAM	2 x 16 Bit	5
8032AH	ไม่มี	256 x 8 RAM	3 x 16 Bit	6
8031AH	ไม่มี	128 x 8 RAM	2 x 16 Bit	5
8031	ไม่มี	128 x 8 RAM	2 x 16 Bit	5
8751H	4k x 8 EPROM	128 x 8 RAM	2 x 16 Bit	5
8751H-12	4k x 8 EPROM	128 x 8 RAM	2 x 16 Bit	5

ตารางที่ 2.1.1 แสดงรายการของตระกูล MCS-51

8031 เป็นอุปกรณ์ที่ถูกออกแบบมาเพื่อสนองความต้องการของผู้ใช้ คือ มีสายอินพุตและเอาต์พุตในตัวมันเอง มีพอร์ทอินพุตและเอาต์พุตไบเฟอ์อินเตอร์เฟสและสายควบคุมอื่นๆ ที่ใช้ สำหรับแยกข้อมูลกับแอดเดรส และยังมีชุดคำสั่งเพิ่มขึ้นเป็นพิเศษ เพื่อจัดการข้อมูลแบบท่ายด้วยวงจรตั้งเวลาและวงจรรนับด้วย

ลักษณะโครงสร้างทางสถาปัตยกรรม

ลักษณะการจัดขาภายนอก

- VCC : ไฟเลี้ยงตัวไอซี (ขา 40) ต้องการแรงดันไฟกระแสตรงขนาด 5 โวลต์
- VSS : กราวด์ของวงจร (ขา 20) เป็นขาสำหรับต่อลงกราวด์
- PORT 0 : (P0.0 - P0.7/DA0 - DA7) (ขา 32-39) เป็นพอร์ทอินพุต/เอาต์พุตแบบโอเพนเดรนไบไดเรกชันแนล (Open Drain Bidirectional) สามารถที่จะรับโหลดที่ที่แอลได้ 8 ตัวการเขียนค่า "1" ไปที่พอร์ทนี้จะทำให้พอร์ทนี้อยู่ในสถานะลอย (Float)

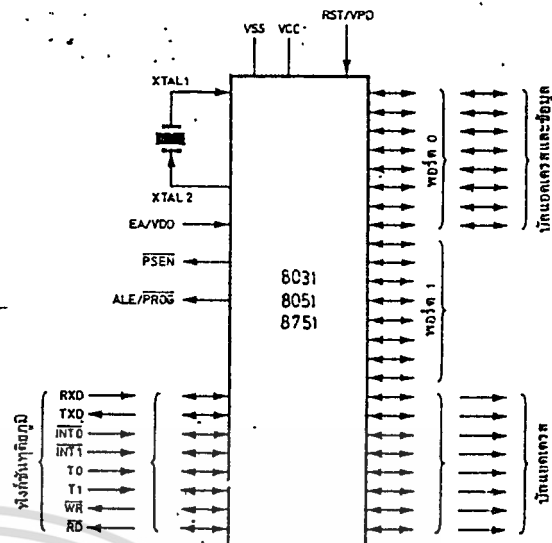
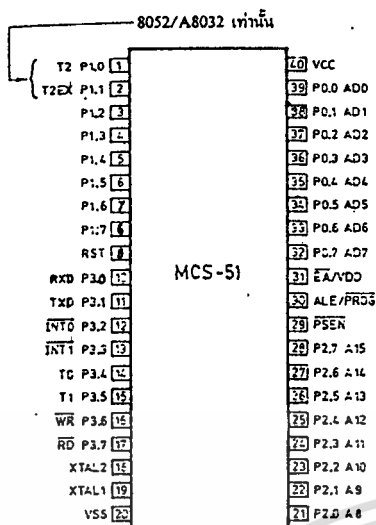
ทำให้มีลักษณะการทำงาน เป็นอินพุตมีสถานะเป็นอิมพีแดนซ์สูงในการให้พอร์ท
นี้บริการแบบอินพุต/เอาต์พุต พอร์ท 0 จะทำงานเป็นตัวมัลติเพลกซ์ด้วยสัญญา-
ณแอดเดรสไบต์ต่ำ กับบัสข้อมูล ในการใช้งานด้านหน่วยความจำภายนอกจะใช้
ลักษณะภายในเป็นตัวพูลอัพ นอกจากนี้พอร์ท 0 ยังใช้งานเป็น ตัวส่งข้อมูลออก
ทางได้อีกด้วย

- PORT 1 : (P1.0 - P1.7) เป็นพอร์ทอินพุตเอาต์พุต 8 บิต แบบโอเพนเดรนไบไดเรค
ชันแนล (Open Drain Bidirectional) พร้อมด้วยการพูลอัพภายในถ้าเป็นพอร์ท
เอาต์พุต บัฟเฟอร์สามารถขับโหลดที่ที่แอลตระกูล LS ได้ 4 ตัวพอร์ท1 เมื่อถูก
เขียนค่า "1" ด้วยโปรแกรมจะทำให้มีสถานะสูง ด้วยการพูลอัพภายในการให้สตา
นะเช่นนี้ จะเป็นการทำให้พอร์ทนี้เป็นอินพุตขณะที่พอร์ทนี้เป็นอินพุตการให้สัญญา
ณ "0" จะเป็นการจ่ายกระแสออกเนื่องจากการพูลอัพภายใน
- PORT 2 : (P2.0 - P2.7) เป็นพอร์ทอินพุตเอาต์พุต 8 บิต แบบโอเพนเดรนไบไดเรค
ชันแนล (Open Drain Bidirectional)พร้อมด้วยการพูลอัพภายใน พอร์ท 2 ทำหน้า
ที่ที่เป็นบัฟเฟอร์เอาต์พุตสามารถจ่ายโหลดที่ที่แอลตระกูล LS ได้ 4 ตัวพอร์ทนี้จะ
ถูกใช้งานเป็นตัวส่งแอดเดรสไบต์สูงด้วยเมื่อใช้งานร่วมกับหน่วยความจำภายนอก
เพื่อให้แอดเดรสสามารถอ้างได้ถึง 16 บิต ด้วยการใช้งานแบบนี้มักจะมีพูลอัพภายใน
ในที่ช่วยให้การส่งค่า "1" ได้ระดับที่แน่นอน
- PORT 3 : (P3.0 - P3.7) เป็นพอร์ทอินพุตเอาต์พุต 8 บิต แบบพูลอัพภายในนอกจาก
ทำเป็นพอร์ทอินพุตเอาต์พุตที่สามารถรับโหลดที่ที่แอลตระกูล LS ได้ 4 ตัวแล้ว
ยังใช้งานเป็นพิเศษ สำหรับตระกูล MCS-51 การทำงานตามฟังก์ชันดังกล่าวจะ
ต้องเริ่มด้วยการส่งค่า "1" ไปแลตช์ (Latch) ที่บิตรีจิสเตอร์ต่าง ๆ ก่อน
- RST : ต้องคงสภาวะ "1" เป็นเวลาอย่างน้อย 2 วัฏจักรระหว่างที่ออสซิลเลเตอร์ทำ-
งานขณะที่ต้องรีเซ็ตทั้งระบบ โดยจะต่อความต้านทานพูลดาวน์ (8.2 กิโลโอห์ม)
จากขา รีเซ็ตลงกราวด์ และเพื่อให้ตัวรีเซ็ตได้โดยอัตโนมัติ ขณะเปิดไฟจะใช้ตัว
เก็บประจุ (10 ไมโครฟารัด) ต่อคร่อมระหว่างขา รีเซ็ตกับขาไฟเลี้ยง
- ALE : ขา Address Latch Enable จะส่งเอาต์พุตออกมาเป็นพัลส์ สำหรับแลตช์แอด
เดรสต่ำระหว่างที่ เข้าถึงแอดเดรสภายนอก และยังเป็นโปรแกรมฟลัสอินพุตขณะที่
โปรแกรมอีพ롬 (ใน MCS-51 ที่มีอีพ롬) ในการใช้งานทั่ว ๆ ไปขาALE จะ
ถูกส่งออกมาทุกๆ 1/6 ของความถี่ออสซิลเลเตอร์และอาจถูกใช้เป็นตัวล๊อคภายนอก
ก็ได้

ขาพอร์ท	การทำงานตามฟังก์ชันพิเศษ
P3.0	RxD พอร์ตอนุกรมอินพุต
P3.1	TxD พอร์ตอนุกรมเอาต์พุต
P3.2	INT0 อินเตอร์รัพท์ภายนอกตัวที่ 1
P3.3	INT1 อินเตอร์รัพท์ภายนอกตัวที่ 2
P3.4	TO สัญญาณกระตุ้นเข้าที่ตัวตั้งเวลาและตัวนับ 0 (ไทม์เมอร์/เคานท์เตอร์ 0)
P3.5	T1 สัญญาณกระตุ้นเข้าที่ตัวตั้งเวลาและตัวนับ 1 (ไทม์เมอร์/เคานท์เตอร์ 1)
P3.6	WR สัญญาณควบคุมการเขียน
P3.7	RD สัญญาณควบคุมการอ่าน

ตารางที่ 2.1.2 แสดงลักษณะการทำงานของพอร์ท 3 ในฟังก์ชันพิเศษ

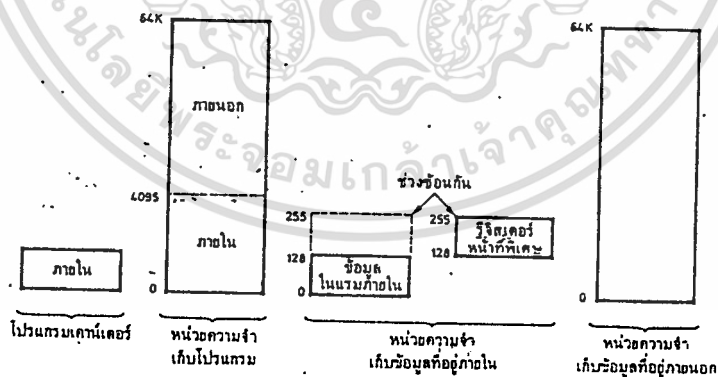
- PSEN : ขา Program Store Enable เป็นสัญญาณรีดสโตรบ (read strobe) ต่อหน่วยความจำโปรแกรมภายนอกโดยขณะที่ 8031 ทำการเอ็กซีคิว (Execute) ได้ค จากหน่วยความจำโปรแกรมภายนอก ขา PSEN จะแอคทีฟทุก ๆ 2 รอบการทำงาน
- EA/Vpp : ขา External Access Enable ขา EA จะต่อกับกราวด์ (GND) เมื่อจะอินาเบิล (enable) ให้เฟลทซ์ได้คจากหน่วยความจำโปรแกรมโดยเริ่มที่แอดเดรส 0000 H ถึง FFFFH
- XTAL1 : เป็นอินพุตที่ต่อกับ อินเวอร์ตติ้งออสซิลเลเตอร์แอมพลิไฟเออร์
- XTAL2 : เป็นเอาต์พุตจากอินเวอร์ตติ้งออสซิลเลเตอร์แอมพลิไฟเออร์ โดยขา XTAL1 และ XTAL2 จะต่อกับคริสตัลและตัวเก็บประจุอีก 2 ตัว



รูปที่ 2.1.1 แสดงการจัดวางขาของ MCS-51

หน่วยความจำ

ในไมโครคอนโทรลเลอร์ตระกูลนี้ จะมีหน่วยความจำอยู่ในตัวเอง ดังแสดงในตารางที่ 1 โดยที่ จะมีโครงสร้างตำแหน่งการใช้งานดังในรูปที่ 2.1.2



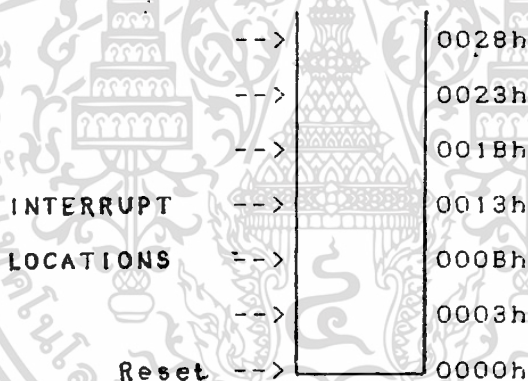
รูปที่ 2.1.2 แสดงโครงสร้างตำแหน่งของหน่วยความจำ

แต่ในโครงงานนี้ตัวไมโครคอนโทรลเลอร์ที่เราเลือกใช้จะมีแต่หน่วยความจำข้อมูลภายในเท่านั้น ส่วนหน่วยความจำโปรแกรม จะต้องต่อจากภายนอก

- หน่วยความจำโปรแกรม

หลังจากที่ตัวชิพยูนิคอร์น ตัวชิพยูนิคอร์นจะเริ่มอ่านหน่วยความจำ สำหรับเก็บโปรแกรมจากแอดเดรสที่ 0000H โดยที่เราสามารถที่จะกำหนดให้ตัวประมวลผล อ่านโปรแกรมการทำงานได้จากหน่วยความจำทั้งแบบภายนอกและภายในโดยใช้ขา EA เป็นตัวกำหนด โดยถ้าขา EA มีสถานะเป็นหนึ่ง ตัวประมวลผลจะอ่านโปรแกรมจากหน่วยความจำภายใน ถ้าขา EA มีสถานะเป็นศูนย์ ตัวประมวลผลจะอ่านโปรแกรมจากหน่วยความจำภายนอก

การอ่านโปรแกรมจากหน่วยความจำภายนอก ตัวประมวลผลจะส่งสัญญาณ PSEN (Program Store Enable) ออกไปยังหน่วยความจำโปรแกรม



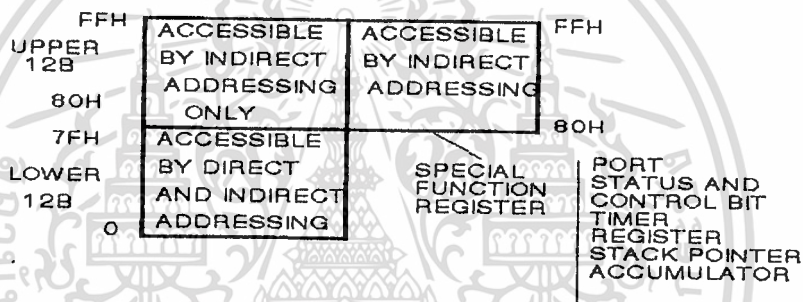
รูปที่ 2.1.3 แสดงหน่วยความจำโปรแกรม

รูปที่ 2.1.3 แสดงถึงตำแหน่งของการบริการอินเตอร์รัพต์ โดยจะอยู่ที่ตำแหน่งต่างๆ ที่คงที่ในหน่วยความจำโปรแกรม ถ้ามีการอินเตอร์รัพต์จะทำให้ชิพยูนิคอร์นจะไปทำงานตามแอดเดรสของอินเตอร์รัพต์นั้นๆ เช่นอินเตอร์รัพต์ภายนอก INTO จะไปทำงานที่แอดเดรส 0003H แต่ถ้าไม่ใช้ INT ก็ใช้เป็นหน่วยความจำโปรแกรมทั่วๆ ไปได้ ตำแหน่งอินเตอร์รัพต์ 0003H เป็น INT 0, 000BH เป็น TIMER 0, 0013H เป็น INT 1, 001BH เป็น TIMER 1

- หน่วยความจำข้อมูล

หน่วยความจำข้อมูลของ 8031 นี้จะอ้างถึงตำแหน่งของหน่วยความจำโดยใช้พอร์ท 0 และ 2 เป็นขाय่างแอดเดรสขนาด 16 บิต และจะใช้สัญญาณ RD (Read), WR (Write) ในการติดต่อกับ หน่วยความจำภายนอกโดยสามารถอ้างหน่วยความจำภายนอกได้ 64 KB หน่วยความจำข้อมูลภายนอก สามารถถูกแอดเดรสได้ทั้งแบบ 1 หรือ 2 ไบต์ก็ได้โดยถ้าอ้างโดยใช้ 1 ไบต์ จะต้องใช้พอร์ทอื่น ๆ มาใช้ในการกำหนดเพจของหน่วยความจำแรม แต่ถ้าอ้างโดยใช้ 2 ไบต์ ไบต์แอดเดรสตัวสูงจะถูกส่งออกมาที่ พอร์ท 2

การแอดเดรสหน่วยความจำข้อมูลภายในจะมีการแบ่งใช้งานออกเป็น 2 ส่วนคือ 128 ไบต์ล่าง และ SFR ดังในรูปที่ 2.1.4



รูปที่ 2.1.4 แสดงโครงสร้างหน่วยความจำข้อมูลภายใน

- หน่วยความจำข้อมูล 128 ไบต์

ในส่วนของหน่วยความจำนี้นั้น จะสามารถแบ่งออกได้เป็น 3 ส่วน คือ

1. ส่วนของรีจิสเตอร์ 32 ไบต์
2. ส่วนของบิตแอดเดรส
3. ส่วนแรมมีใช้งานทั่วไป

1. ส่วนของรีจิสเตอร์



จะเริ่มตั้งแต่แอดเดรส 0000H-001FH โดยที่จะแบ่งรีจิสเตอร์ออกเป็น 4 กลุ่มๆ ละ 8 บิต ในการใช้งานจริงจะสามารถใช้ได้ทีละกลุ่มเท่านั้น และเราจะใช้ 2 บิตในรีจิสเตอร์ PSW ในการเลือกกลุ่มของรีจิสเตอร์

8031 มีรีจิสเตอร์ที่อำนวยความสะดวกในการใช้งานตามคำสั่งต่างๆ อยู่มากมายเป็นรีจิสเตอร์ขนาด 8 บิต ประกอบด้วยรีจิสเตอร์ต่าง ๆ ดังต่อไปนี้

- แอดคิวมูเลเตอร์ (Accumulator)

เป็นรีจิสเตอร์ที่ใช้งานเกี่ยวกับการประมวลผลต่างๆ ของตัวชิพและใช้คำนวณทางตัวเลข

- รีจิสเตอร์ B (register B)

เป็นรีจิสเตอร์ที่ใช้ในการคูณและหารค่าต่าง ๆ ในการทำงานของชิพ

- สแต็กพอยน์เตอร์ (Stack Pointer)

เป็นรีจิสเตอร์ขนาด 8 บิต ซึ่งจะมีค่าเพิ่มขึ้น เมื่อมีข้อมูลถูกเก็บระหว่าง การใช้คำสั่ง PUSH และ CALL สแต็กพอยน์เตอร์จะเริ่มที่แอดเดรสตำแหน่งที่ 07H หลังจากตัวชิพถูกรีเซ็ต

- ดาต้าพอยน์เตอร์ (Data Pointer)

เป็นรีจิสเตอร์ขนาด 8 บิต 2 ตัว ซึ่งจะประกอบด้วยค่าไบต์สูงและไบต์ต่ำโดยที่มีหน้าที่ใช้เก็บแอดเดรสขนาด 16 บิต ในการติดต่อกับหน่วยความจำภายนอก

- ซีเรียลดาต้าบัฟเฟอร์ (Serial Data Buffer)

เป็นรีจิสเตอร์ขนาด 8 บิต โดยจะแยกเป็นรีจิสเตอร์บัฟเฟอร์ตัวส่งและตัวรับข้อมูล

- รีจิสเตอร์แสดงสถานะของโปรแกรม (Program Status Word (PSW))

เป็นรีจิสเตอร์ขนาด 8 บิต ประมาณด้วยแฟล็กที่ใช้บอกสถานะของโปรแกรกดังในรูปที่ 5

(MSB)

(LSB)

CY	AC	FO	RS1	RS0	OV	P
----	----	----	-----	-----	----	---

สัญลักษณ์ตำแหน่งข้อกำหนดการทำงาน

CY	PSW7	แฟลกตัวทด จะเซต/เคลียร์ด้วยฮาร์ดแวร์หรือซอฟต์แวร์ระหว่างผลลัพธ์หลังการใช้คำสั่งทางคณิตศาสตร์หรือตรรกศาสตร์ที่แน่นอน
AC	PSW6	แฟลกตัวทดลองของ Auxiliary จะเซต/เคลียร์ด้วยฮาร์ดแวร์ระหว่างการบวกและลบ ที่แสดงผลจากการทดหรือยืมจากบิตที่ 3 ของ ACC
PO	PSW5	แฟลก O จะเซต/เคลียร์ด้วยซอฟต์แวร์ที่ผู้กำหนดสถานะแฟล็กนี้เอง
RS1	PSW4	เรจิสเตอร์ควบคุมการเลือกแบงก์ ด้วยค่า RS1 และ RS0 จะเซต/เคลียร์ด้วยซอฟต์แวร์ เพื่อเลือกกลุ่มเรจิสเตอร์ทำงานในแต่ละแบงก์โดยปรับค่าใน RS1 และ RS0 ให้อินาเบิ้ลคุณสมบัติการเลือกแบงก์ต่อไปนี้
RS0	PSW3	

RS1	RS0	เลือกแบงก์	ค่าแอดเดรส
0	0	แบงก์ 0	00H - 07H
0	1	แบงก์ 1	08H - 0FH
1	0	แบงก์ 2	10H - 17H
1	1	แบงก์ 3	18H - 1FH

OV

PSW2

แฟลก (Overflow) จะเซต/เคลียร์ด้วยฮาร์ดแวร์ระหว่างการใช้คำสั่งที่แสดงผลถึงการเกิดลักษณะ Overflow ทางคณิตศาสตร์

- PSW1 บิตสำรอง จะไม่สามารถเซต/เคลียร์ด้วยผู้ใช้ เพราะสำรอง ไว้สำหรับ
โรงงานผู้สร้าง
- P PSW0 แพลกพาริตี จะเซต/เคลียร์ด้วยฮาร์ดแวร์ ในแต่ละวัฏจักรคำสั่งแสดง
ถึงตัวเลขค่า "1" ในแต่ละบิตของแอกคิวมูเลเตอร์เช่น "1" มี 6 ตัวจะ
เป็นพาริตีคู่ P บิต จะเท่ากับ 0

- ไทม์เมอร์/เคาน์เตอร์รีจิสเตอร์ (Timer/Counter Register)

เป็นรีจิสเตอร์ขนาด 8 บิตที่ใช้งานเป็นคู่ ได้ (TH0,TL0), (TH1,TL1)เป็นรีจิสเตอร์ที่
ใช้ในการนับสำหรับการทำงานแบบไทม์เมอร์/เคาน์เตอร์

- คอนโทรล รีจิสเตอร์ (Control Register)

เป็นรีจิสเตอร์ขนาด 8 บิต ที่ใช้ทำหน้าที่พิเศษ ได้แก่ IP, IE, TMOD, TCON, SCON
และ PCON จะประกอบด้วยบิตคอนโทรลและบิตแสดงสถานะสำหรับใช้งานแบบอินเตอร์รัพต์
ไทม์เมอร์/เคาน์เตอร์ และซีเรียลพอร์ทการทำงานของไทม์เมอร์/เคาน์เตอร์ จะถูกควบคุม
โดยรีจิสเตอร์ 2 ตัวคือ TMOD และ TCON

GATE	C/T	M1	M0	GATE	C/T	M1	M0
------	-----	----	----	------	-----	----	----

<-----TIMER R1-----><-----TIMER R0----->

GATE : ควบคุมเกต เมื่อเซตเป็น "1" จะเป็นอินพุต ตัวจับเวลา/ตัวนับเท่านั้น ขณะที่
ที่ขา INTx มีสถานะสูงและควบคุม TRx ใน TCON จะถูกเซตเป็น "1" เมื่อตัว
นับภายในถูกเคลียร์ให้อินพุตเมื่อไรก็ตามที่บิตควบคุม TRx ถูก เซตเป็น "1"

C/T : เลือกการทำงานแบบตัวจับเวลา หรือตัวนับ ถ้าเป็น "0" จะเลือกทำงานเป็นตัวจับเวลา (โดยใช้สัญญาณนาฬิกาภายในเป็นสัญญาณอ้างอิงถึง) ถ้าเป็น "1" จะเป็นการทำงานแบบตัวนับ และรับสัญญาณเข้าที่ขา Tx

M1	M0	การทำงาน
0	0	ทำงานแบบตัวจับเวลาของ MCS-48 ใช้ TLx เป็นตัวป้อน บิตอีก 5 บิต
0	1	การใช้ตัวจับเวลา/ตัวนับขนาด 16 บิต จะใช้ THx และ TLx เป็นตัวนับไม่มี Prescaler
1	0	การไหลขนาด 8 บิต โดยอัตโนมัติที่ตัวนับและตัวจับเวลา โดยใช้ THx เก็บค่าที่ตั้งไว้ และถ่ายเข้าไปที่ TLx ใหม่ทุกครั้งที่เกิด Over flow คือ TLx ถูกนับเป็น "0"
1	1	ตัวจับเวลา 0 ทำงาน โดยให้ TLO และ TH0 เป็นตัวนับนับแยกกัน

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
-----	-----	-----	-----	-----	-----	-----	-----

สัญลักษณ์ ตำแหน่ง

ข้อกำหนดการทำงาน

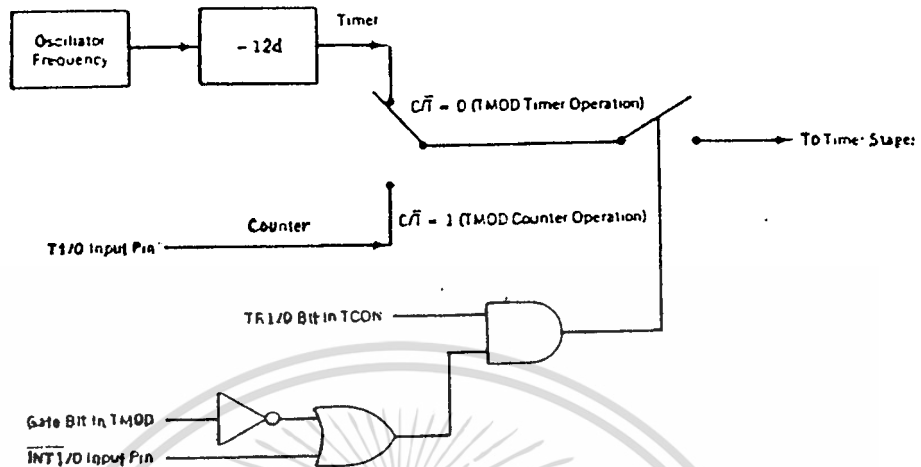
TF1	TCON.7	ตัวจับเวลา 1 แฟล็กเป็น "1" เมื่อเกิด Overflow ถูกเซตเป็นหนึ่ง ด้านฮาร์ดแวร์ทางสัญญาณ เมื่อ ตัวจับเวลา / ตัวนับ Overflow และ จะเคลียร์ตัวเองเมื่ออินเตอร์รัพต์ไปแล้ว
TR1	TCON.6	ตัวจับเวลา 1 เป็นตัวควบคุมบิตให้เริ่มทำงาน จะเซตหรือเคลียร์ด้วยซอฟต์แวร์ที่จะมาทำให้ ตัวจับเวลา/ตัวนับ 1 เริ่มหรือหยุดการทำงาน

TF0	TCON.5	ตัวจับเวลา 0 แพลกเป็น '1' เมื่อเกิด Overflow ถูกเซตเป็นหนึ่ง ด้านฮาร์ดแวร์ทางสัญญาณ เมื่อตัวจับเวลา / ตัวนับ Overflow เคลียร์ตัวเองเมื่ออินเตอร์รัพต์ไปแล้ว
TRO	TCON.4	ตัวจับเวลา 0 เป็นตัวควบคุมบิตให้เริ่มทำงานจะเซตหรือเคลียร์ ด้วยซอฟต์แวร์ที่จะมาทำให้ ตัวจับเวลา/ตัวนับ เริ่มหรือหยุดการทำงาน
IE1	TCON.3	อินเตอร์รัพต์ 1 เป็นแฟลกขอสัญญาณ เซตด้วยฮาร์ดแวร์เมื่อสัญญาณขอการอินเตอร์รัพต์ปรากฏเข้าที่ขา INT1 และเคลียร์ เมื่อการทำงานอินเตอร์รัพต์สิ้นสุด
IT1	TCON.2	อินเตอร์รัพต์ 1 รูปแบบการควบคุมบิต จะเซต หรือเคลียร์ได้ด้วยซอฟต์แวร์ที่จะเป็นตัวกำหนดให้ การกระตุ้นอินเตอร์รัพต์จากภายนอกที่ขอบขา ลงหรือระดับแรงดันต่ำ โดยถ้า IT1 = 1 จะควบคุมอินเตอร์รัพต์ด้วยขอบขาสูง และถ้า IT1= 0 จะควบคุมอินเตอร์รัพต์ ด้วยระดับแรงดันต่ำ
IE0	TCON.1	อินเตอร์รัพต์ 0 เป็นแฟลกขอสัญญาณ เซตด้วยฮาร์ดแวร์ เมื่อสัญญาณขอการอินเตอร์รัพต์ปรากฏเข้าที่ขา INTO และเคลียร์ เมื่อการทำงานอินเตอร์รัพต์สิ้นสุด
ITO	TCON.0	อินเตอร์รัพต์ 0 รูปแบบการควบคุมบิตจะเซต หรือเคลียร์ได้ด้วยซอฟต์แวร์ ที่จะเป็นตัวกำหนด ให้การกระตุ้นอินเตอร์รัพต์จากภายนอกเป็นแบบขอบขาสูงหรือเป็นแบบระดับแรงดันต่ำ

นอกจากผู้ใช้จะสามารถเลือกการทำงานของรีจิสเตอร์เป็นไทม์เมอร์หรือเคานท์เตอร์ได้แล้ว ผู้ใช้ยังสามารถกำหนดให้ไทม์เมอร์หรือเคานท์เตอร์ มีการทำงานที่พิเศษแยกย่อยลงไปได้อีกถึง 4 โหมด ตามความเหมาะสมของการใช้งาน (TO,T1 มีให้เลือก 4 โหมด แต่ T2 มีให้เลือก 3 โหมด) ดังจะได้กล่าวต่อไป

- ไทม์เมอร์ 0 และไทม์เมอร์ 1

สามารถจะเลือกการทำงานให้เป็นไทม์เมอร์ หรือเคานท์เตอร์ได้ ด้วยการเปลี่ยนค่าบิต C/T ซึ่งอยู่ในรีจิสเตอร์ TMOD ถ้าบิตนี้เป็น 0 หมายถึงให้เป็นไทม์เมอร์ แต่ถ้าบิตนี้เป็น 1 หมายถึงให้เป็นเคานท์เตอร์ ลักษณะการทำงานของไทม์เมอร์/เคานท์เตอร์ แสดงได้ดังรูปที่ 2.1.5

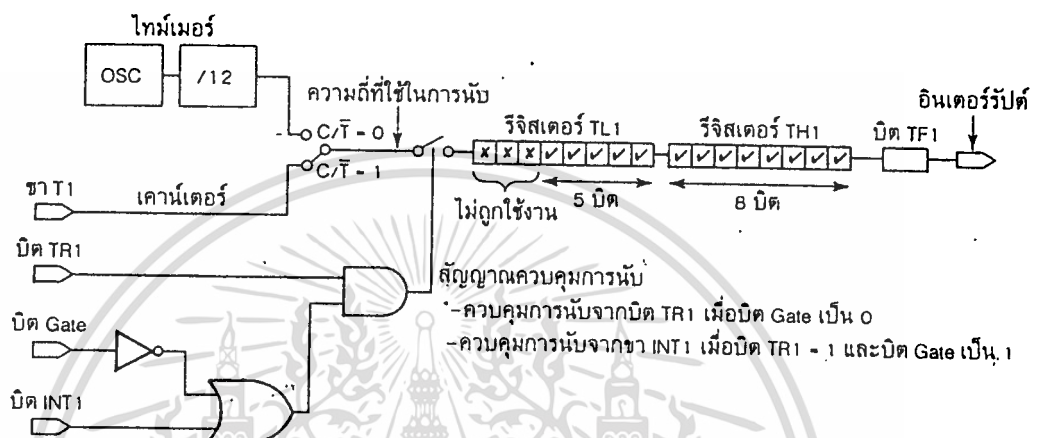


รูปที่ 2.1.5 การควบคุมการทำงานของไทม์เมอร์/เคาน์เตอร์

ไทม์เมอร์ / เคาน์เตอร์ทั้ง 2 ตัวนี้มีโหมดการทำงานอยู่ 4 โหมดโดยการทำงานจะเป็นโหมดใดขึ้นอยู่กับค่าของบิต M0, M1 ในรีจิสเตอร์ TMOD เช่นเดียวกับบิต C/T การทำงานในโหมด 0,1,2 จะคล้าย ๆ กันสำหรับไทม์เมอร์/เคาน์เตอร์ทั้ง 2 แต่ในโหมด 3 จะมีการทำงานที่ต่างกันออกไปจากนั้น 3 โหมดดังนี้

- การทำงานของไทม์เมอร์/เคาน์เตอร์โหมด 0

ไทม์เมอร์/เคาน์เตอร์ในโหมดนี้ จะถูกกำหนดการทำงานให้เป็นเคาน์เตอร์ขนาด 8 บิตซึ่งจะถูกเพิ่มค่าต่อเมื่อนับสัญญาณจากแมชชีนไซเคิลหรือการเปลี่ยนสถานะลอจิกที่ขา T0, T1 ครบ 32 แล้ว (เป็นเคาน์เตอร์ขนาด 8 บิตที่ถูกหารด้วย 32) กล่าวอีกอย่างหนึ่งคือไทม์เมอร์แต่ละตัว จะทำงานเป็นเคาน์เตอร์เพื่อนับแมชชีนไซเคิล หรือการเปลี่ยนสถานะลอจิกที่ขา T0, T1 ขนาด 13 บิต นั้นเองโดยการนับสัญญาณ 32 ครั้งแรก จะใช้รีจิสเตอร์ TLx จำนวน 5 บิต ($2^5 = 32$) และนำมาประกอบกับรีจิสเตอร์ THx จำนวน 8 บิตเพื่อให้เป็นเคาน์เตอร์ขนาด 13 บิต



รูปที่ 2.1.6 โหมดการทำงานของไทม์เมอร์/เคาน์เตอร์

จากรูปที่ 2.1.6 จะเห็นว่ารีจิสเตอร์ TL1 และ TH1 จะมาประกอบกันเข้าเป็นเคาน์เตอร์ขนาด 13 บิต เมื่อค่าในรีจิสเตอร์ทั้ง 2 เปลี่ยนจากเป็น 1 ทั้งหมดไปเป็น 0 ทั้งหมด จะมีผลไปเซตบิตไทม์เมอร์อินเตอร์รัพต์แฟลก TF1 อินพุตที่ใช้นับ ถูกอินทิเกรตให้กับไทม์เมอร์เมื่อบิต TR1 และ Gate = 0 หรือสถานะที่ขา INT1 เป็น 1 อย่างใดอย่างหนึ่ง ดังแสดงในภาพ (การเซตให้ Gate เป็น 1 จะยอมให้ไทม์เมอร์ถูกควบคุมโดยสถานะที่ขา INT1 ทำให้การวัดความกว้างของพัลส์ทำได้ง่าย) TR1 เป็นบิตในรีจิสเตอร์ TCON ซึ่งแต่ละบิตมีความหมายดังแสดงในรูปที่ 2.1.6 ส่วน Gate เป็นบิตหนึ่งในรีจิสเตอร์ TMOD ดังได้กล่าวมาแล้ว

รีจิสเตอร์ขนาด 13 บิต ประกอบด้วย 8 บิตของรีจิสเตอร์ TH1 และ 5 บิตของรีจิสเตอร์ TL1 ส่วน 3 บิตบนของรีจิสเตอร์ TL1 ไม่ถูกกำหนดและไม่ถูกใช้การเซตค่าของบิตรีเฟลก (TR1) ก็ไม่ได้เคลียร์ค่าในรีจิสเตอร์ทั้งสอง

- การทำงานของไทม์เมอร์/เคานท์เตอร์โหมด 1

ในโหมด 1 นี้มีการทำงานเหมือนกับโหมด 0 ทุกประการ เว้นแต่ค่าในไทม์เมอร์รีจิสเตอร์จะถูกรันทั้ง 16 บิตเลยนั่นคือ ในโหมดนี้ไทม์เมอร์/เคานท์เตอร์ จะเป็นเคานท์เตอร์หรือไทม์เมอร์ขนาด 16 บิตแทน

- การทำงานของไทม์เมอร์/เคานท์เตอร์โหมด 2

ในโหมด 2 จะกำหนดไทม์เมอร์/เคานท์เตอร์รีจิสเตอร์ ทำงานเป็น 8บิตเคานท์เตอร์ (TLx) ซึ่งมีการโหลดค่าเอง โดยอัตโนมัติ ด้วยค่าในรีจิสเตอร์ THx เมื่อเกิด Overflow ในไทม์เมอร์/เคานท์เตอร์ ซึ่งค่าในรีจิสเตอร์ THx นี้สามารถกำหนดได้ล่วงหน้า โดยซอฟต์แวร์ และจะไม่เปลี่ยนแปลง เมื่อถูกโหลดไปไว้ในรีจิสเตอร์ TL1

- การทำงานของไทม์เมอร์/เคานท์เตอร์โหมด 3

ในการทำงานของไทม์เมอร์/เคานท์เตอร์โหมด 3 นี้ ไทม์เมอร์ 1 จะไม่มีการนับ ซึ่งมีผลเหมือนกับให้ค่า $TR1 = 0$

ไทม์เมอร์ 0 ในโหมด 3 จะทำให้รีจิสเตอร์ TLO และ THO ทำงานเหมือนเป็นเคานท์เตอร์ขนาด 8 บิต แยกต่างหากจากกัน 2 ตัว สำหรับการทำงานในโหมด 3 ของไทม์เมอร์ 0 แสดงในรูปที่ 9 ซึ่งจะเห็นว่ารีจิสเตอร์ TLO ใช้ Timer 0 Control bits C/T , Gate , TRO, INTO และ TFO ส่วนรีจิสเตอร์ THO จะถูกบังคับให้เป็นไทม์เมอร์ (นับจำนวน machine cycle) และรับการนับ TR1 และ TF1 จากไทม์เมอร์ 1 ดังนั้นขณะรีจิสเตอร์ THO จะควบคุมไทม์เมอร์ 1 อินเทอร์รัพต์

- การสื่อสารข้อมูลแบบอนุกรม

ซีเรียลพอร์ทที่มีใน MCS-51 สามารถทำงานได้ในแบบ Full Duplex สามารถรับและส่งข้อมูลได้พร้อม ๆ กัน โดยมีบัฟเฟอร์ในการรับข้อมูล กล่าวคือ สามารถรับข้อมูลไบต์ที่สองเข้ามาได้ก่อนที่ไบต์แรก ซึ่งได้รับเข้ามาก่อนจะถูกอ่านจากรีซีฟรีจิสเตอร์ (แต่ถ้าไบต์แรกยังไม่ถูกอ่าน เมื่อการรับไบต์ที่สองสิ้นสุดลงข้อมูลจะเกิดการสูญหาย) ซีเรียลพอร์ทจริงๆแล้ว

ประกอบไปด้วยรีจิสเตอร์ 2 ตัว คือ รีซีฟรีจิสเตอร์ และทรานสมิสรีจิสเตอร์ รีจิสเตอร์ทั้ง 2 มีตำแหน่งเดียวกันใน SFR คือ รีจิสเตอร์ SBUF การเข้าถึงรีจิสเตอร์แต่ละตัว ซีพียูจะรู้ว่าต้องติดต่อกับรีจิสเตอร์ตัวใด การเขียนข้อมูลไปที่รีจิสเตอร์ SBUF จะหมายถึง โหลดค่าไปยังทรานสมิสรีจิสเตอร์ ส่วนการอ่านข้อมูลในรีจิสเตอร์ SBUF หมายถึง การรับข้อมูลจากรีซีฟรีจิสเตอร์

โหมดการทำงานของ MCS-51 มีการทำงานแบ่งออกเป็น 4 โหมด ดังนี้

MODE 0 : ข้อมูลแบบอนุกรมถูกรับเข้าและส่งออกผ่านขา TXD ส่วนขา RXD จะให้ Shift Clock ในโหมดนี้จะทำการรับส่งข้อมูลแบบ 8 บิต โดยรับและส่งบิตต่าก่อน (LSB First) ส่วนอัตราการส่งข้อมูล (Baud Rate) ถูกกำหนดไว้แน่นอนที่ 1/12 ของความถี่ออสซิลเลเตอร์ที่ใช้ในระบบ

MODE 1 : ข้อมูลจำนวน 10 บิตถูกส่ง (ผ่านทางขา TXD) หรือถูกรับ (ผ่านทางขา RXD) โดยมีบิตเริ่มต้น 1 บิต (มีค่าเป็น 0) รับส่งข้อมูลแบบ 8 บิต (รับและส่งบิตต่าก่อน) และบิตสิ้นสุด 1 บิต (มีค่าเป็น 1) ขณะที่ทำการรับ บิตสิ้นสุดจะไปอยู่ในบิต RB8 ของ SFR SCON ส่วนค่า Baud Rate สามารถเปลี่ยนแปลงได้ ดังจะได้กล่าวในรายละเอียดต่อไป

MODE 2 : ข้อมูลจำนวน 11 บิตถูกส่ง (ผ่านทางขา TXD) หรือถูกรับ (ผ่านทางขา RXD) โดยมีบิตเริ่มต้น 1 บิต (มีค่าเป็น 0) รับส่งข้อมูลแบบ 8 บิต (รับหรือส่งบิตต่าก่อน) บิตที่ 9 เป็นบิตที่สามารถโปรแกรมได้ (Programmable 9th Data bit) และบิตสิ้นสุด 1 บิต (มีค่าเป็น 1) โดยในขณะที่ส่งข้อมูล บิตข้อมูลบิตที่ 9 (the 9th data bit) ซึ่งเป็นบิต TB8 ใน SFR SCON สามารถถูกกำหนดให้เป็น 0 หรือ 1 ตัวอย่าง เช่น พาริตีบิต (bit P ในรีจิสเตอร์ PSW) สามารถนำไปไว้ที่บิต TB8 ของรีจิสเตอร์ SCON โดยไม่สนใจบิตสิ้นสุดค่า Baud Rate สามารถตั้งให้เป็น 1/32 หรือ 1/64 ของความถี่ออสซิลเลเตอร์ที่ใช้

MODE 3 : ข้อมูลจำนวน 11 บิตถูกส่ง (ผ่านทางขา TX) หรือถูกรับ (ผ่านทางขา RXD) โดยมีบิตเริ่มต้น 1 บิต (มีค่าเป็น 0) รับส่งข้อมูลแบบ 8 บิต (LSB First) 1 Programmable 9th bit และบิตสิ้นสุด 1 บิต (มีค่าเป็น 1) ซึ่งจะเห็นได้ว่าในโหมด 3 นี้จะเหมือนกับโหมด 2 ทุกอย่าง เว้นเสียแต่ในโหมด 3 นี้สามารถเปลี่ยนแปลงค่า Baud Rate ได้ ดังจะกล่าวถึงรายละเอียดต่อไป

การทำงานของซีเรียลพอร์ททั้ง 4 โหมดที่กล่าวมานี้ การส่งข้อมูล ถูกทำให้เกิดขึ้นโดยคำสั่งใด ๆ ที่ใช้ SBUF เป็นดิสตินเนชันรีจิสเตอร์ ส่วนการรับข้อมูลถูกทำให้เกิดขึ้น ในโหมด 0 โดยเงื่อนไข $R1 = 0$ และ $REN = 1$ การรับข้อมูลถูกทำให้เกิดขึ้นในโหมดอื่น ๆ โดยรับบิตเริ่มต้นเข้ามา ถ้า $REN = 1$

- การติดต่อระหว่างส่วนประมวลผลหลาย ๆ ตัว
(COMMUNICATIONS)

(MULTIPROCESSOR

การทำงานของซีเรียลพอร์ทในโหมด 2 และ โหมด 3 จะมีรูปแบบการใช้งานพิเศษ นอกเหนือจากการรับ-ส่งข้อมูลธรรมดาตามที่ได้กล่าวมาแล้ว การใช้งานพิเศษที่ว่านี้คือ การติดต่อสื่อสารระหว่างซีพียูด้วยกันเอง (Multiprocessor communications) โดยซีพียูที่ใช้ในระบบมีจำนวนหลายตัวและสามารถเลือกการติดต่อกับตัวใดก็ได้ โดยการทำงานในโหมดนี้บิตที่ 9 ที่รับเข้ามาจะถูกนำไปไว้ใน บิต RB8 จากนั้นบิตสิ้นสุดก็จะถูกรับตามเข้ามา ซึ่งซีเรียลพอร์ทสามารถโปรแกรม เพื่อที่เมื่อบิตสิ้นสุดถูกรับเข้ามาแล้ว จะมีผลไปกระตุ้นให้ส่วนซีเรียลพอร์ทอินเตอร์รัพท์ทำงานเพื่ออินเทอร์รัพท์ซีพียูก็ต่อเมื่อ บิต RB8 = 1 ซึ่งการทำเช่นนี้ได้ก็โดยการเซตบิต SM2 ในรีจิสเตอร์ SCON โดยรายละเอียดและแนวทางในการใช้คุณสมบัติพิเศษที่เรียกว่า Multiprocessor นั้นมีดังนี้

เมื่อหน่วยประมวลผลหลัก (Master Processor) ต้องการส่งข้อมูลจำนวนหนึ่งไปยังหน่วยประมวลผลย่อย (Slave) ตัวหนึ่งจากที่มีหลายตัวในระบบ ในขั้นแรกหน่วยประมวลผลหลักจะต้องส่งค่าแอดเดรสขนาด 1 ไบต์ซึ่งจะเป็นค่าที่บอกตำแหน่งของหน่วยประมวลผลเป้าหมายที่ต้องการติดต่อกับแอดเดรสไบต์ จะมีข้อแตกต่างจากไบต์ข้อมูลทั่วไปตรงที่บิตที่ 9 จะเป็น 1 เมื่อข้อมูลนั้นเป็นแอดเดรสและจะเป็น 0 เมื่อข้อมูลนั้นเป็นข้อมูล หากในหน่วยประมวลผลย่อยมีการเซตบิต SM2 = 1 แล้วจะไม่มีหน่วยประมวลผลย่อยตัวใดถูกอินเตอร์รัพท์ หากข้อมูลที่เข้ามาเป็นข้อมูล แต่ถ้าข้อมูลที่ได้รับเข้ามาเป็นแอดเดรส(บิตที่ 9 มีค่า 1) หน่วยประมวลผลย่อยทุกตัวจะถูกอินเตอร์รัพท์ เพื่อว่าแต่ละตัวจะสามารถตรวจสอบได้ว่า ข้อมูลขนาด 1 ไบต์ ที่ได้รับเข้ามามีค่าตรงกับแอดเดรสของตัวเองหรือไม่ หน่วยประมวลผลย่อยที่มีค่าแอดเดรสตรงกับข้อมูลที่ได้รับเข้ามา จะทำการเคลียร์บิต SM2 และเตรียมรับข้อมูลที่เป็นข้อมูล ซึ่งจะตามเข้ามาหลังจากรับแอดเดรสแล้วส่วนหน่วยประมวลผลย่อยตัวอื่นที่ตรวจสอบข้อมูลดูแล้ว ปรากฏว่าไม่ตรงกับแอดเดรสของตัวเอง จะยังคงปล่อยให้บิต SM2 ของมันถูกเซตต่อไปและกลับไปทำงานที่ค้างอยู่ก่อนได้รับการอินเตอร์รัพท์ต่อ โดยไม่สนใจข้อมูลที่เป็นข้อมูล ซึ่งตามเข้ามาหลังแอดเดรส เช่นกัน

บิต SM2 จะไม่มีผลในการทำงานของซีเรียลพอร์ทโหมด 0 แต่การทำงานในโหมด 1 บิต SM2 สามารถถูกใช้เพื่อที่จะตรวจสอบบิตสิ้นสุด (Validity of the stop bit) ในการรับข้อมูลของโหมด 1 ถ้าบิต SM2 = 1 การเกิดรีซีฟอินเตอร์รัพท์ จะยังไม่ทำงานจนกว่าบิตสิ้นสุดที่รับข้อมูลเข้ามามีค่าเป็น 1

-Serial Port Control Register

ซีเรียลพอร์ทคอนโทรลเรจิสเตอร์ คือหน่วยความจำ SFR ที่เรียกละย่อว่า SCON โดยรีจิสเตอร์ตัวนี้ไม่เพียงแต่เก็บข้อมูลเกี่ยวกับ โหมดการทำงานของซีเรียลพอร์ท เท่านั้นแต่ยังใช้เป็นทีเก็บข้อมูลซึ่งเป็นบิตที่ 9 สำหรับการส่งและการรับ(บิตTB8 และRB8) รวมทั้งยังมีบิตที่ทำหน้าที่อินเตอร์รัพต์ซีพียู (Serial Port Interrupt) คือบิต T1 และ R1 อีกด้วย

SM0	SM1	SM2	REN	TB8	RB8	T1	R1
-----	-----	-----	-----	-----	-----	----	----

โดย SM0, SM1 เป็นตัวกำหนดใช้โหมดต่าง ๆ ของพอร์ทอนุกรม ดังนี้

SM0	SM1	โหมด	ลักษณะการทำงาน	ตัวบ่งชี้
0	0	0	เลื่อนรีจิสเตอร์	IOBC/12
0	1	1	8 บิต UART	แปรผันได้ตามการเลือกตัวจับเวลา
1	0	2	9 บิต UART	IOBC/64 หรือ IOBC/32
1	1	3	9 บิต UART	แปรผัน

UART : Universal Asynchronous Receiver / Transceiver

SM2 ควบคุมอินพุตเปิด การให้โพรมิตเซอร์หลายตัวในการสื่อสารซึ่งกันและกันในโหมด 2 และ 3 ถ้า SM2 เซตเป็น 1 ดังนั้น RI จะต้องไม่แอกทีฟ ถ้าไม่มีการรับบิตที่เกาทำให้บิต RB8 นี้ เป็น 0 ในโหมด 1 ถ้า SM2 เซตเป็น 1 ดังนั้น RI จะไม่แอกทีฟถ้า STOP บิตไม่ถูกรับในโหมด 0 SM2 ควรมีค่าเท่ากับ 0

REN ตัวอินพุตเปิดอนุกรมการรับเซตเป็น "1" ด้วยโปรแกรมในการเลือกอินพุตการรับและ เป็น "0" ด้วยโปรแกรม เมื่อให้เป็นคัสเอเบิล

- TB8 เป็นข้อมูลบิตที่ 9 ซึ่งจะถูกส่งในโหมด 2 และ 3 ซึ่งจะให้ เป็น "1" หรือ "0" ได้ด้วยการโปรแกรม
- RB8 ในโหมด 2 และ 3 ข้อมูลบิตที่ 9 จะถูกรับไปในโหมด 1 ถ้า SM2 = 0 RB8 จะกลายเป็น STOP บิตที่ถูกรับไปในโหมด 0 RB8 ไม่ใช่
- TI เป็นแฟล็กอินเตอร์รัพต์การส่ง เซตด้วยฮาร์ดแวร์ คือสัญญาณที่ปลายช่วงเวลาของบิตที่ 8 ใน โหมด 0 หรือที่จุดเริ่มต้นของบิต STOP ในโหมดอื่น ในการส่งแบบอนุกรมของทุกโหมดจะต้องเคลียร์บิตนี้ด้วยโปรแกรมหลังการส่ง
- RI เป็นแฟล็กอินเตอร์รัพต์การรับ เซตด้วยฮาร์ดแวร์คือสัญญาณที่ปลายช่วงเวลาของบิตที่ 8 ในโหมด 0 หรือที่จุดครึ่งทางของช่วงบิต STOP ในโหมดอื่นในการรับแบบอนุกรมยกเว้นกรณีใช้ SM2 จะต้องเคลียร์บิตด้วยโปรแกรมหลังการรับ

-Baud Rate

Baud Rate หมายถึง อัตราเร็วในการรับส่งข้อมูลซึ่งจะมีค่าเท่าใดก็ขึ้นอยู่กับการทำงานในแต่ละโหมดของรีจิสเตอร์พอร์ทดังนี้

- ในโหมด 0 Baud Rate = (ความถี่ออสซิลเลเตอร์ที่ใช้)/12
- ในโหมด 2 Baud Rate ขึ้นอยู่กับค่าของบิต SMOD ซึ่งอยู่ในรีจิสเตอร์ PCON โดยถ้าบิต SMOD = 0 (ซึ่งเป็นค่านี้เมื่อเกิดการรีเซต) Baud Rate จะเป็น 1/64 ของความถี่ออสซิลเลเตอร์ที่ใช้ แต่ถ้าบิต SMOD = 1 Baud Rate จะเป็น 1/32 ของความถี่ออสซิลเลเตอร์ หรืออาจเขียนเป็นสูตรได้ว่า

$$\text{Baud Rate ในโหมด 2} = [2(\text{SMOD}) \times (\text{Oscillator Frequency})]/64$$

ส่วน Baud Rate ในโหมด 1 และ 3 จะถูกกำหนดโดยอัตราการเกิด Overflow ในไทม์เมอร์ 1 (Timer 1 Overflow Rate)

-การใช้ไทม์เมอร์ 1 เป็นตัวสร้าง Baud Rates

เมื่อไทม์เมอร์ 1 ถูกใช้เป็นตัวกำเนิดสัญญาณ Baud Rate (Baud Rate Generator)

สำหรับการทำงานของซีเรียลพอร์ทในโหมด 1 และ 3 โดย Baud Rate จะถูกกำหนดด้วยอัตราการเกิด Overflow ของไทม์เมอร์ 1 และยังขึ้นอยู่กับบิต SMOD ในรีจิสเตอร์ PCON อีกด้วย ซึ่งเราอาจเขียนเป็นสมการที่ใช้คำนวณหา Baud Rate ได้ดังนี้

$$\text{Baud Rate ในโหมด 1,3} = [2^{\text{SMOD}} \times (\text{อัตรา Overflow ของไทม์เมอร์ 1})] / 32$$

เนื่องจากเมื่อเกิด Overflow ในไทม์เมอร์ตัวใด จะทำให้ซีพียูถูกอินเตอร์รัพต์ ดังนั้นเมื่อเรานำไทม์เมอร์ 1 มาเป็นตัวสร้างสัญญาณ Baud Rate จึงควรที่จะห้ามไม่ให้เกิดอินเตอร์รัพต์ขึ้นในระหว่างการรับส่งข้อมูล และตัวไทม์เมอร์เองยังสามารถถูกกำหนดให้ทำงานเป็นไทม์เมอร์ หรือเคานท์เตอร์อย่างใดอย่างหนึ่งได้ ซึ่งมีโหมดการทำงานย่อยลงไปอีก 3 โหมดในการใช้งานที่พบบ่อยที่สุดนั้น ไทม์เมอร์ 1 ถูกกำหนดให้อยู่ในโหมด Auto-Reload (ค่า 4 บิตบนของ TMOD คือ 0010B) ในกรณีนี้ Baud Rate จะถูกกำหนดโดยสมการดังนี้

$$\text{Baud Rate Mode 1,3} = \frac{2^{\text{SMOD}}}{32} \times \frac{\text{Oscillator Frequency}}{12 \times [256 - (\text{TH1})]}$$

เราสามารถสร้าง Baud Rate ค่าต่ำ ๆ ด้วยไทม์เมอร์ 1 โดยปล่อยให้ไทม์เมอร์ 1 ทำการอินเตอร์รัพต์ซีพียูได้ และกำหนดให้ไทม์เมอร์ทำงานในแบบ 16 บิต (โหมด 1 ค่าบิตบนของ TMOD คือ 0010B) และใช้ไทม์เมอร์ 1 อินเตอร์รัพต์ซีพียู เพื่อทำการโหลดค่าเองด้วยซอฟต์แวร์และเกิด Overflow เนื่องจากในโหมด 16 บิต ไม่มีการทำงานแบบ Auto Reload นั้นเอง

- การอินเตอร์รัพต์

8051 กำหนดให้มีแหล่งอินเตอร์รัพต์ได้ 5 แหล่ง ดังมีรายละเอียดดังนี้

- การอินเตอร์รัพต์จากภายนอก (INT0 ,INT1)

เป็นอินเตอร์รัพต์ที่เกิดขึ้นภายนอก แต่ละตัวสามารถเลือกให้เป็น Level Activated หรือ Transition Activated อย่างใดอย่างหนึ่ง ขึ้นอยู่กับบิต IT0 และ IT1 ในรีจิสเตอร์ TCON

แฟลกที่ทำให้เกิดการอินเตอร์รัพต์ในซีพียูจริง ๆ แล้ว คือค่าในบิต IE0 และ IE1 ของ รีจิสเตอร์ TCON โดยเมื่อเกิดอินเตอร์รัพต์จากภายนอกขึ้น แฟลกซึ่งทำให้เกิดอินเตอร์รัพต์ จะถูกเคลียร์โดยฮาร์ดแวร์ เมื่อซีพียูถูกย้ายไปทำงาน ที่ตำแหน่งของแอดเดรสที่กำหนดได้ (interrupt service routine) ก็ต่อเมื่ออินเตอร์รัพต์ตัวนั้นเป็นชนิด transsition-activated แต่ถ้าอินเตอร์รัพต์ตัวนั้นเป็นชนิด level-activated แล้วเมื่อซีพียูถูกย้ายไปทำงานที่ตำแหน่งของแอดเดรสที่กำหนดไว้ (interrupt service routine) จะไม่เคลียร์บิต IEx ให้วงจรภายนอก ที่ขออินเตอร์รัพต์ซีพียูจะต้องทำหน้าที่ควบคุมสถานะของสัญญาณที่ขา INTx ให้กลับสู่สภาพ เดิมเอง

- ไทม์เมอร์ 0 และไทม์เมอร์ 1 อินเตอร์รัพต์

จะถูกทำให้เกิดอินเตอร์รัพต์ได้โดยบิต TFO และ TF1 ซึ่งถูกเซตโดยการเปลี่ยนค่า จากเป็น 1 ทั้งหมดเป็น 0 ทั้งหมดในไทม์เมอร์/เคานท์เตอร์ รีจิสเตอร์ของแต่ละตัว (ยกเว้น ไทม์เมอร์ 0 ในโหมด 3) เมื่อไทม์เมอร์อินเตอร์รัพต์เกิดขึ้นค่าของแฟลกที่ทำให้เกิดอินเตอร์รัพต์จะถูกเคลียร์โดยฮาร์ดแวร์ทันทีเมื่อซีพียูถูกย้ายไปทำงานในส่วน ของ Service Routine

- ซีเรียลพอร์ทอินเตอร์รัพต์

เราสามารถที่จะสร้างสัญญาณอินเตอร์รัพต์ได้โดยสัญญาณที่จะทำให้เกิดอินเตอร์รัพต์ ได้มาจากบิต TI หรือ RI โดยนำมาผ่านเกตแบบ OR และค่าในแฟลกทั้งสองตัวนี้จะไม่ถูก เคลียร์โดยฮาร์ดแวร์ เมื่อซีพียูไปทำงานในส่วน Service Routine เพราะจริง ๆ แล้ว ใน Service Routine จะต้องหาว่า RI หรือ TI เป็นตัวทำให้เกิดอินเตอร์รัพต์ขึ้น และบิตนั้นจะถูก เคลียร์ได้โดยซอฟต์แวร์เท่านั้น

บิตทั้งหมดที่ทำให้เกิดสัญญาณอินเตอร์รัพต์สามารถถูกเซตหรือเคลียร์โดยฮาร์ดแวร์ นั่นคือ อินเตอร์รัพต์ทั้งหมดที่กล่าวมา สามารถอินาเบิลให้เกิดสัญญาณอินเตอร์รัพต์หรืออินเตอร์รัพต์ที่กำลังรอการบริการจากซีพียูอยู่ถูกยกเลิกได้ด้วยซอฟต์แวร์

อินเตอร์รัพต์ซอร์สแต่ละชนิดเหล่านี้ สามารถถูกอินาเบิล (enable) เป็นตัว ๆ หรือ ถูกดิสเอเบิล (disale) เป็นตัว ๆ ได้โดยการเปลี่ยนแปลงบิตใน SFR IE ซึ่งจะมีบิตที่เป็น Global Disable Bit (EA) อยู่ด้วย ซึ่งสามารถดิสเอเบิล (disable) อินเตอร์รัพต์ทั้งหมดได้ในครั้ง เดียว

EA	X	ET2	ES	ET1	EX1	EX0	EX0
----	---	-----	----	-----	-----	-----	-----

2-23

- โครงสร้างระดับของไพรอริตี้ (Priority Level Structure)

ในแต่ละอินเตอร์รัพต์ซอร์ส สามารถที่จะถูกเลือกกระับความสำคัญได้ 2 ระดับ โดยการเซตหรือการเคลียร์บิตในรีจิสเตอร์ IP ดังในรูปที่ 12

ระดับความสำคัญของการขออินเตอร์รัพต์ที่มีได้ 2 ระดับ ได้แก่

- Low Priority Interrupt : สามารถถูกอินเตอร์รัพต์ได้โดย High Priority Interrupt แต่ไม่สามารถถูกอินเตอร์รัพต์โดย Low Priority Interrupt ตัวอื่น ๆ ได้
- High Priority Interrupt : ไม่สามารถถูกอินเตอร์รัพต์จากอินเตอร์รัพต์ซอร์สตัวอื่นได้เลย

PT2	PS	PT1	PX1	PT0	PX0
-----	----	-----	-----	-----	-----

สัญลักษณ์	ตำแหน่ง	ฟังก์ชัน
-	IP.7	สำรอง
-	IP.6	สำรอง
PT2	IP.5	กำหนดระดับความสำคัญการอินเตอร์รัพต์ตัวจับเวลา 2 ถ้า PT2 = 1 เป็นการโปรแกรมให้มีระดับการอินเตอร์รัพต์ความสำคัญสูงกว่า
PS	IP.4	การโปรแกรมให้มีระดับการอินเตอร์รัพต์ฟอร์ทอนุกรมถ้า PS = 1 เป็นการโปรแกรมให้มีระดับความสำคัญสูงกว่า
PT1	IP.3	กำหนดระดับความสำคัญอินเตอร์รัพต์ตัวจับเวลา 1
PX1	IP.2	กำหนดระดับความสำคัญอินเตอร์รัพต์ภายนอก 1 (INT1)
PT0	IP.1	กำหนดระดับความสำคัญอินเตอร์รัพต์ตัวจับเวลา 0
PX0	IP.0	กำหนดระดับความสำคัญอินเตอร์รัพต์ภายนอก 0 (INT0)

ถ้ามีการอินเทอร์รัพต์พร้อมกัน 2 ชนิด ซึ่งมีระดับความสำคัญในการขออินเทอร์รัพต์ไม่เท่ากัน อินเทอร์รัพต์ที่มีระดับสูงกว่า (High Priority Interrupt) จะได้รับการบริการก่อนแต่ถ้ามีการขออินเทอร์รัพต์พร้อมกัน 2 ชนิด ซึ่งมีระดับความสำคัญเท่ากันลำดับการรับสัญญาณอินเทอร์รัพต์ภายในจะกำหนดเองว่าอินเทอร์รัพต์ชนิดใดจะถูกบริการก่อน ดังนั้นภายในระดับความสำคัญของการขออินเทอร์รัพต์หนึ่ง ๆ จะมี Second Priority Structure (ระดับความสำคัญในการขออินเทอร์รัพต์ย่อย) ดังนี้

แหล่งอินเทอร์รัพต์	ระดับของไพริอริตี้
1. IEO	สูงสุด
2. TFO	:
3. IE1	:
4. TF1	:
5. RI+TI	:
6. TF2+EXF2	ต่ำสุด

รีจิสเตอร์ IP มีบิตที่ไม่ถูกใช้งานอยู่จำนวนหนึ่ง คือ IP.7, IP.6, และ IP.5 ตำแหน่งการบริการอินเทอร์รัพต์ (interrupt service routine) ที่เกี่ยวข้องกับสัญญาณอินเทอร์รัพต์ขณะนั้น ดังแสดงข้างล่าง

แหล่งอินเทอร์รัพต์	ตำแหน่งของแอดเดรส
IEO	0003H
TFO	000BH
IE1	0013H
TF1	001BH
RI+TI	0023H
TF2+EXF2	002BH

2. ส่วนของบิตแอดเดรส

จะเริ่มที่ตำแหน่งแอดเดรสที่ 0020H-002FH การใช้งานของหน่วยความจำส่วนนี้ก็คือ เราสามารถที่จะกำหนดข้อมูลเป็นบิตได้ โดยที่จะมีการเก็บข้อมูลแบบเป็นบิต

3. ส่วนแรมที่ใช้งานทั่วไป

จะเริ่มตั้งแต่ตำแหน่งที่ 0030H-007FH โดยจะใช้เป็นส่วนที่ใช้ในการเก็บข้อมูลทั่วๆ ไป

- ส่วน SFR

เป็นส่วนของแรมภายในที่ทำหน้าที่พิเศษ สามารถที่จะเข้าถึงได้โดยการอ้างแอดเดรสตั้งแต่ 0080H-00FFH เหมือนแรมทั่วๆ ไป แต่ละตัวจะมีแอดเดรสที่แตกต่างกันออกไป แต่แอดเดรสตั้งแต่ 0080H-00FFH จำไม่ถูกใช้งานเป็น SFR ทั้งหมด แต่อย่างไรก็ตามส่วนที่เหลือก็ไม่สามารถที่จะนำไปใช้งานได้ ส่วน SFR จะประกอบด้วยรีจิสเตอร์ต่าง ๆ ดังตารางที่ 2.1.3

Symbol	Name	Address
*ACC	Accumulator	0E0H
*B	Register	0F0H
*PSW	Program Status Word	0D0H
SP	Stack Pointer	81H
DPTR	Data Pointer 2 Bytes	
DPL	Low Byte	82H
DPH	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	0CCH
*TL2	Timer/Counter 2 Low 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

ตารางที่ 2.2.3

อินพุทเอาต์พุทของซีพียู

โครงสร้างของอินพุทเอาต์พุทพอร์ทของ 8031 ถูกออกแบบมาให้มีการติดต่อกับอุปกรณ์ต่าง ๆ ภายนอกได้ทั้งแบบขนานและอนุกรม โดยที่การติดต่อแบบขนาน จะมีขาให้ใช้งานถึง 32 ขา แบ่งออกเป็นพอร์ท 4 พอร์ท พอร์ทละ 8 ขา ส่วนการติดต่อแบบอนุกรมนั้นจะมีขาที่ใช้ในการติดต่ออยู่ 4 ขา

การทำงานของตัวซีพียู เมื่อเริ่มต้นการทำงานขาอินพุทเอาต์พุทต่าง ๆ จะมีค่าสถานะของสัญญาณเป็นหนึ่งทุกขา และการทำงานของขาอินพุทเอาต์พุทนั้น จะเป็นการรับส่งสัญญาณแบบขนาน มีพอร์ทที่จะติดต่อใช้งานแบบขนาน 4 พอร์ท ดังนี้

- พอร์ท 0 จะทำหน้าที่ในการส่งค่าแอดเดรสไบต์ต่ำ ออกไปติดต่อกับหน่วยความจำภายนอกทางพอร์ทนี้ และพร้อมกันนี้จะมีการส่งข้อมูล ที่ใช้ในการติดต่อออกมาทางพอร์ทนี้ด้วย โดยใช้วิธีการส่งแบบมัลติเพลกซ์ (Multiplex)

- พอร์ท 1 จะทำหน้าที่ในการติดต่อเป็นพอร์ทแบบขนานขนาด 8 บิต โดยที่สามารถที่จะกำหนดให้ทำงานเป็นได้ทั้งพอร์ทอินพุทและเป็นพอร์ทเอาต์พุท

- พอร์ท 2 จะทำหน้าที่ในการติดต่อเป็นพอร์ทแบบขนานขนาด 8 บิต โดยที่สามารถที่จะกำหนดให้ทำงาน เป็นได้ทั้งพอร์ทอินพุทและเป็นได้ทั้งพอร์ทเอาต์พุท แต่เมื่อตัวซีพียูมีการติดต่อกับหน่วยความจำภายนอกโดยที่จะต้องมีการอ้างค่าแอดเดรสของหน่วยความจำ ซึ่งมีขนาด 16 บิต พอร์ทนี้จะทำหน้าที่ในการส่งค่าของแอดเดรสไบต์สูง ออกมาติดต่อกับหน่วยความจำ

- พอร์ท 3 จะทำหน้าที่ในการติดต่อเป็นพอร์ทแบบขนาน 8 บิต โดยที่สามารถที่จะกำหนดให้ทำงานเป็นได้ทั้งพอร์ทอินพุทและเป็นได้ทั้งพอร์ทเอาต์พุท แต่พอร์ท 3 นี้เราสามารถที่จะกำหนดการทำงานแต่ละขาของพอร์ทนี้ ซึ่งมีอยู่ 8 ขา ให้เป็นขาที่ใช้ในการติดต่อแบบอนุกรมได้ ดังจะกล่าวดังต่อไปนี้

1. ขา RXD เป็นขาที่ใช้ในการรับสัญญาณแบบอนุกรม จากวงจรภายนอกเข้ามาในตัวซีพียู
2. ขา TXD เป็นขาที่ใช้ในการส่งสัญญาณแบบอนุกรม จากตัวซีพียูออกไปยังวงจรภายนอก
3. ขา INTO เป็นขาที่ใช้ในการรับสัญญาณขัดจังหวะการทำงานของตัวซีพียูโดยสัญญาณจะถูกส่งออกมาจากอุปกรณ์ภายนอกที่ตัวซีพียูติดต่อกอยู่ หลังจากทีซีพียูได้รับสัญญาณนี้แล้วตัวซีพียู จะกระโดดไปยังตำแหน่งหน่วยความจำ ที่ถูกกำหนดไว้ให้เป็นส่วนที่เก็บโปรแกรมการบริการอินเทอร์รัพต์ของขานั้น ๆ

4. ขา INT1 เป็นขาที่มีลักษณะการทำงานเหมือนกับขา INTO จะแตกต่างกันตรงที่ค่าของ

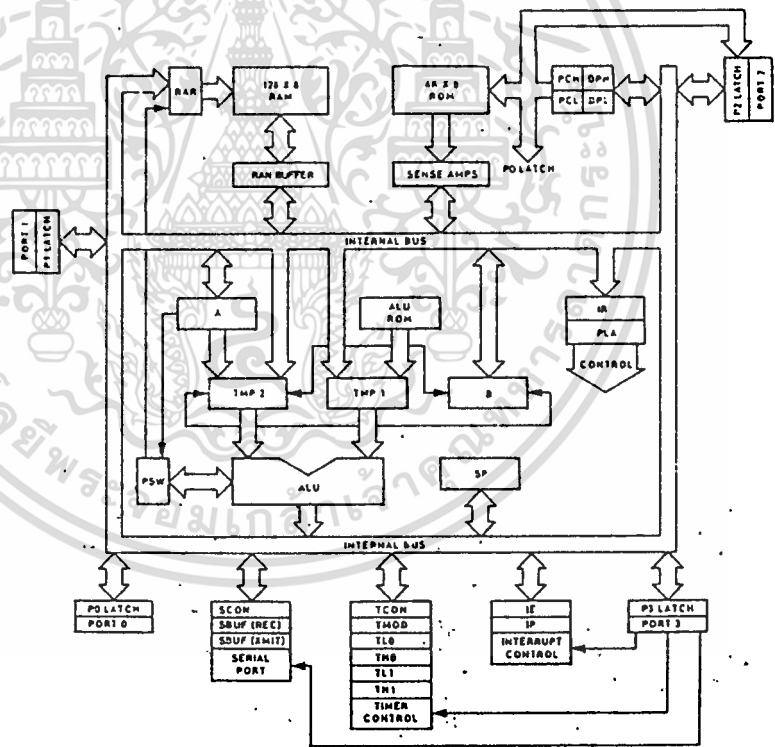
แอดเดรสที่ใช้ในการกำหนดหน่วยความจำ ที่ใช้ในการเก็บโปรแกรมการบริการอินเทอร์พรัต มีค่าต่างกัน

5. ขา T0 เป็นขาที่ใช้ในการรับสัญญาณแบบอนุกรม (Serial) โดยที่ใช้ฟังก์ชันการทำงานของเคาน์เตอร์-ไทม์เมอร์ นอกจากนี้ขาเนี่ยังอาจจะเป็นขาที่ใช้ป้อนสัญญาณนาฬิกาของระบบออกไปให้วงจรภายนอก ที่ต้องการใช้สัญญาณนาฬิกาอีกด้วย

6. ขา T1 เป็นขาที่มีลักษณะการใช้งานคล้ายกับขา T0 แต่ขาเนี่ยไม่สามารถที่จะจ่ายสัญญาณออกมาได้

7. ขา RD เป็นขาที่ใช้ส่งสัญญาณการอ่านหน่วยความจำข้อมูลออกไปให้แก่ หน่วยความจำภายนอก เพื่อบอกให้หน่วยความจำภายนอกส่งข้อมูลมายังซีพียู

8. ขา WR เป็นขาที่ใช้ส่งสัญญาณ การเขียนหน่วยความจำข้อมูลออกไปให้แก่ หน่วยความจำภายนอก เพื่อบอกว่าตัวซีพียูต้องการที่จะส่งข้อมูลออกไปเก็บ ได้ที่หน่วยความจำภายนอก



รูปที่ 2.1.7 โครงสร้างของ MCS-51

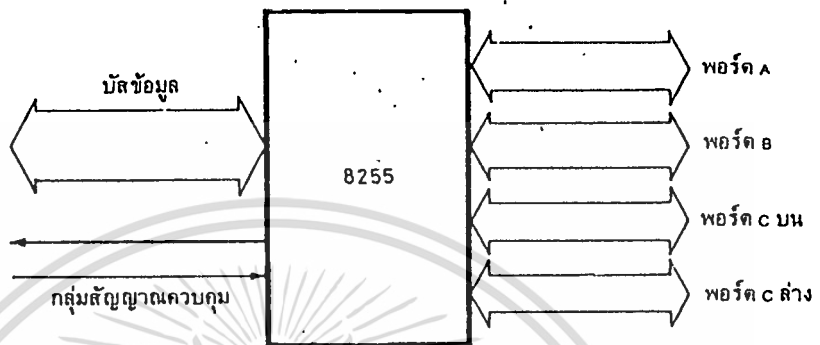
2.2 ไอซี 8255 โปรแกรมเมเบิล เพอริเฟอร์ล

อินเตอร์เฟซซิง(Programmable Peripherals Interfacing)

เป็นไอซี พอร์ทที่สามารถโปรแกรมการทำงานได้ ในการนำเอาไมโครโปรเซสเซอร์ไปใช้งานนั้น จำเป็นต้องให้ไมโครโปรเซสเซอร์สามารถติดต่อกับโลกภายนอกได้ซึ่งก็คือให้มันสามารถส่งสัญญาณมาควบคุมอุปกรณ์ต่าง ๆ ได้ เช่น สเตปมอเตอร์ ความคุมอุปกรณ์ต่าง ๆ ส่วนที่ทำให้ไมโครโปรเซสเซอร์ สามารถติดต่อกับโลกภายนอกได้ ที่รู้จักกันดีก็คือ พอร์ท (Port)ซึ่งมีอยู่หลายลักษณะด้วยกันเช่นเป็นตัวไอซีแบบไตรสเตท(Tri-state)เบอร์74LS244 หรือพวกแลตช์ (Latch) เช่น 74LS374 เหล่านี้สามารถนำมาต่อใช้งานกับ CPU ได้ง่ายที่สุด โดยตัว CPU จะเป็นตัวควบคุมการอ่านเขียนพอร์ท หากเป็นการอ่านข้อมูลจากพอร์ท มักจะใช้ไอซีแบบ ไตรสเตทเป็นอินพุต โดย CPU จะส่งสัญญาณไปเปิดเกตของไตรสเตทนี้ให้ข้อมูลเข้าสู่สายข้อมูล (Data Bus) และเข้าสู่ไมโครโปรเซสเซอร์หรือ CPU ต่อไป แต่สำหรับพอร์ทเอาต์พุตก็จะใช้ แลตช์ ฟลิป-ฟลอป ทำหน้าที่รับสัญญาณจากไมโครโปรเซสเซอร์ มาแลตช์ไว้ที่ตัวมัน (CPU ส่งสัญญาณมาทริก) เพื่อให้อุปกรณ์ภายนอกนั้นรับสัญญาณข้อมูลจากตัวแลตช์ไปอีกทีหนึ่ง ที่ทำเช่นนี้เพราะตัวไมโครโปรเซสเซอร์หรือ CPU นี้ทำงานเร็วมาก ซึ่งในช่วงของการส่งข้อมูลออกพอร์ทจะใช้เวลาไม่กี่ไมโครเซค(s)ซึ่งอาจจะทำให้อุปกรณ์ภายนอกรับไม่ทัน มีบริษัทต่าง ๆ เล็งเห็นความสำคัญของการติดต่อระหว่าง CPU กับอุปกรณ์ภายนอกนี้ จึงได้ทำไอซีสำเร็จรูปขึ้นหลายเบอร์ และที่จะขอนำมากล่าวในบทนี้ก็คือไอซีเบอร์ 8255 ซึ่งเป็นของบริษัทอินเทล ซึ่งได้ออกแบบมาใช้กับไอซีเบอร์ 8080 แต่เราสามารถนำมาประยุกต์ใช้กับเบอร์อื่น ๆ ได้โดยไม่ยุ่งยาก สาเหตุที่ 8255 เป็นที่นิยมมากก็เพราะว่ามันสามารถถูกโปรแกรมให้ทำงานในลักษณะต่าง ๆ ไม่ว่าจะเป็นอินพุตหรือเอาต์พุตหรือแม้แต่แบบแฮนด์เชกกิง (Handshaking)ได้ทั้งยังราคาถูกอีกต่างหาก

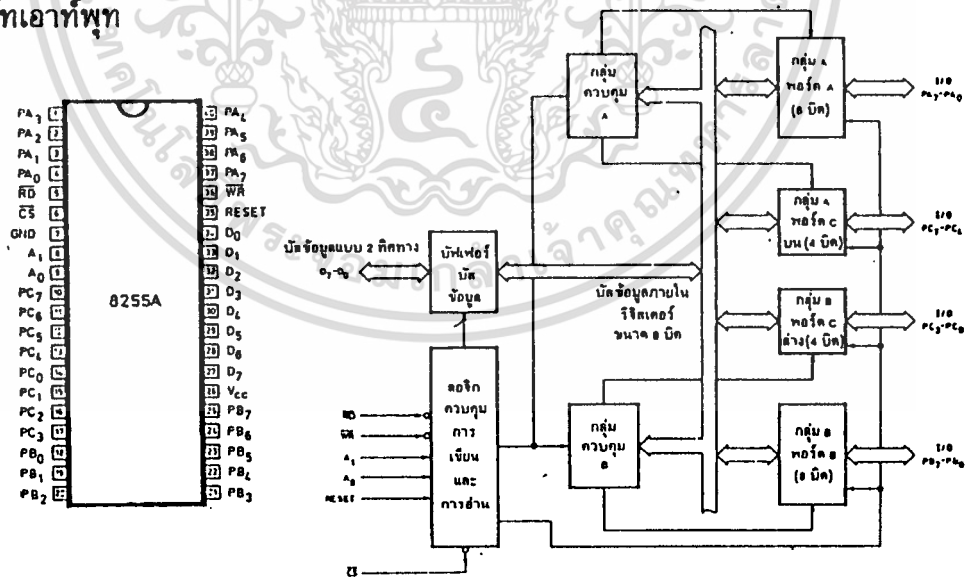
8255 พอร์ตข้อมูลแบบขนาน

8255 เป็นไอซีที่มีขา 40 ขา ได้รับการออกแบบมาเป็นพอร์ตให้ไมโครโปรเซสเซอร์ ได้ 3 พอร์ต โดยมีโครงสร้างพื้นฐานแสดงได้ดังรูปที่ 2.2.1



รูปที่ 2.2.1 แผนผังโครงสร้างของไอซี 8255

การเรียกพอร์ตของ 8255 จะเรียกพอร์ตต่าง ๆ ว่า พอร์ต A พอร์ต B และพอร์ต C โดยพอร์ต C แยกเป็น 2 ส่วนคือพอร์ต c ล่างหรือตั้งแต่ PC₀ - PC₃ มีจำนวน 4 บิต และ พอร์ต C บน หรือตั้งแต่ PC₄ - PC₇ ที่พิเศษคือ พอร์ตทุกพอร์ตเป็นได้ทั้งพอร์ตอินพุตและพอร์ตเอาต์พุต



รูปที่ 2.2.2 แผนผังวงจรภายในและการจัดวางขาของไอซี 8255

รูปที่ 2.2.2 เป็นแผนผังภายในของไอซีและการจัดวางขาของไอซี 8255 การทำงานของวงจรจะใช้สัญญาณควบคุมจากไมโครโปรเซสเซอร์ มาควบคุมการทำงานโดยไมโครโปรเซสเซอร์ จะส่งคำสั่งมาโปรแกรมการทำงาน หรือกำหนดรูปแบบของพอร์ท ให้เป็นอินพุทหรือเอาต์พุทได้

ขาต่าง ๆ ของ 82555

เพื่อให้เข้าใจวิธีการต่อใช้งาน 8255 จึงจำเป็นต้องเข้าใจความหมายและตำแหน่งของขาต่าง ๆ เสียก่อน ขาทัง 40 ขาของไอซีประกอบด้วย

D₀-D₇ เป็นขาที่ข้อมูลอินพุทเอาต์พุทจะต้องผ่านเข้าออกจากส่วนนี้ D₀-D₇ จึงต่อเข้ากับระบบบัสของไมโครโปรเซสเซอร์ เพื่อให้ไมโครโปรเซสเซอร์สามารถอ่าน หรือเขียนข้อมูลจากพอร์ทผ่านทางบัสนี้

CS (สัญญาณเลือกชิพ) ขานี้เป็นขาอินพุทที่รับสัญญาณจากภายนอก เพื่อเลือกชิพ 8255 โดยเมื่อขานี้เป็น "0" จะทำให้ 8255 ต่อเข้ากับระบบบัสของไมโครโปรเซสเซอร์ เพื่อให้ไมโครโปรเซสเซอร์เขียนหรืออ่านข้อมูลจากพอร์ทได้

RD (สัญญาณการอ่าน) เป็นสัญญาณอินพุทที่ต้องส่งมาจากชิพยูเมื่อสัญญาณที่ขานี้เป็น "0" และสัญญาณ CS เป็น "0" ด้วย ไอซี 8255 จะทำตัวให้ชิพยูอ่านข้อมูลจากบัส ในขณะที่เป็นพอร์ทอินพุท

WR เป็นสัญญาณการเขียน จะแอกทีฟเมื่อสัญญาณ WR และสัญญาณ CS เป็น "0" สัญญาณนี้จะมาจาก ชิพยูเมื่อต้องการเขียนข้อมูลลงบนพอร์ทที่กำหนด

A₀- A₁ (สัญญาณแอดเดรส) ลอจิกของสัญญาณทั้งสองจะถอดรหัสออกเป็น 4 รหัสเพื่อกำหนดครีจิสเตอร์ภายในที่เชื่อมต่อกับพอร์ทอินพุทเอาต์พุทของ 8255

RESET (สัญญาณรีเซต) เป็นสัญญาณที่ส่งจากภายนอกเข้ามาทำหทารีเซต 8255 เพื่อเคลียร์สถานะต่างๆ ของ 8255 เมื่อ 8255 ได้รับการรีเซต ก็จะกลับเข้าสู่โหมดอินพุทหรือพอร์ทที่เป็นพอร์ทอินพุท

PA₀ - PA₇ เป็นสายสัญญาณที่เป็นพอร์ทของ 8255 ถูกเลือกโดยสัญญาณแอดเดรส A₀ - A₁

PB₀ -PB₇ เป็นสายสัญญาณที่เป็นพอร์ท B ของ 8255 ถูกเลือกโดยสัญญาณแอดเดรส A₀ - A₁

PC₀ - PC₇ เป็นสายสัญญาณที่เป็นพอร์ท C ของ 8255 การกำหนดพอร์ทนี้จะได้รับการกำหนดโดยสัญญาณแอดเดรส A₀ - A₁ พอร์ท C นี้แบ่งเป็น 2 กลุ่มคือ กลุ่ม PC₀ - PC₃ และกลุ่ม PC₄ - PC₇

รีจิสเตอร์ภายในของ 8255

เมื่อต่อ 8255 เข้ากับ MCS-51 ได้แล้ว สิ่งที่ใช้จะต้องทำก็คือ การทำโปรแกรมให้ 8255 ทำงานตามที่ต้องการ จากกรณีที่ 8255 มีพอร์ทที่ MCS-51 มองเห็น 4 พอร์ท แต่ละพอร์ท จะเสมือนเป็นรีจิสเตอร์ที่สามารถเขียนและอ่านได้ รีจิสเตอร์แต่ละตัวนี้จึงถูกกำหนดด้วยแอดเดรสตามที่ตั้งไว้ เช่น ในกรณีที่แอดเดรส 10H, 11H, 12H และ 13H รีจิสเตอร์แต่ละตัวจะได้รับการกำหนดควบคุมกับสัญญาณ RD และ WR เพื่อแสดงความหมาย ตัวอย่างเช่น พอร์ท A ซึ่งเมื่อเขียนที่พอร์ทนี้ จะเป็นการส่งข้อมูลเอาต์พุตและถ้าอ่านพอร์ทนี้ ก็จะเป็นการอินพุตข้อมูลจากพอร์ท ดังนั้นสัญญาณของขาควบคุมที่ประกอบกัน จะแสดงความหมายดังตารางที่ 2.2.1

RD	WR	A1	A0	ความหมาย
1	0	0	0	เขียนพอร์ท A ซึ่งเป็นข้อมูล
0	1	0	0	อ่านพอร์ท A ซึ่งเป็นข้อมูล
1	0	0	1	เขียนพอร์ท B ซึ่งเป็นข้อมูล
0	1	0	1	อ่านพอร์ท B ซึ่งเป็นข้อมูล
1	0	1	0	เขียนพอร์ท C ซึ่งเป็นข้อมูล
0	1	1	0	อ่านพอร์ท C ซึ่งเป็นข้อมูล
1	0	1	1	เขียนข้อมูลซึ่งเป็นรหัสควบคุม
0	1	1	1	อ่านเข้ามาซึ่งไม่มีความหมายใด

ตารางที่ 2.2.1 สัญญาณควบคุมการทำงานของ 8255

การใช้งาน 8255 จะต้องส่งรหัสควบคุม (control code) เข้าไปยังพอร์ทข้อมูลควบคุม เพื่อควบคุมการทำงานของ 8255 โดยใช้สัญญาณควบคุมพอร์ทหมายเลข 13H การควบคุมการทำงานของ 8255 มีหลายโหมด แต่ละโหมดจะแตกต่างกันออกไป การโปรแกรมให้ 8255 ทำงานจะทำได้ 3 โหมดคือ โหมด 0 โหมด 1 โหมด 2

โหมด 0 หรือ อินพุทเอาต์พุทแบบพื้นฐาน

การกำหนดโหมดการทำงาน จะต้องส่งข้อมูลคำสั่งเข้าไปโปรแกรมในพอร์ทควบคุมของ 8255 ซึ่งในที่นี้ใช้พอร์ทหมายเลข 13 H แต่ละบิตของข้อมูลที่ส่งไปจะมีความหมายในตัวเอง ลักษณะความหมายของแต่ละบิตในรหัสควบคุมแสดงได้ดังรูปที่ 7.6

การโปรแกรม 8255 คือ การให้ค่ารหัสบิตต่าง ๆ เข้าไปในรหัสควบคุมแล้วส่งไปยังรีจิสเตอร์ของพอร์ทควบคุม ความหมายของบิตต่าง ๆ มีดังนี้

บิต D₇ เป็นบิตที่แสดงรหัสคำสั่งควบคุม ถ้าบิตนี้เป็น "1" หมายถึงรหัสควบคุมนี้จะมีผลต่อการเปลี่ยนแปลงการเซตโหมดต่าง ๆ ของ 8255

บิต D₆ และ D₅ เป็นการเลือกโหมดของพอร์ท A ซึ่งมี 3 โหมดคือ โหมด 0 โหมด 1 และ โหมด 2 ดังรูปที่ 7.6

บิต D₄ ถ้ามีค่าเป็น "0" หมายถึงการกำหนดพอร์ท A เป็นเอาต์พุท ถ้ามีค่าเป็น "1" จะหมายถึงการกำหนดให้พอร์ท A เป็นอินพุท

บิต D₃ เป็นบิตที่บอกถึงการเซตโหมดของพอร์ท C บน ถ้าเป็น "0" จะทำให้พอร์ท C บนเป็นเอาต์พุท

บิต D₂ เป็นบิตที่บอกถึงการเซตโหมดของพอร์ท B ถ้าเป็น "0" หมายถึง การเลือกพอร์ท B เป็นโหมด 0 และถ้าเป็น "1" หมายถึงการเลือกโหมด 1

บิต D₁ เป็นการกำหนดอินพุทเอาต์พุทของพอร์ท B ถ้าเป็น "0" หมายถึงเอาต์พุทถ้าเป็น "1" หมายถึงอินพุท

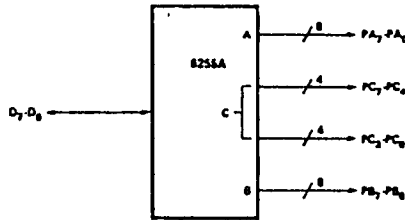
บิต D₀ เป็นการกำหนดอินพุทเอาต์พุทของพอร์ท C ล่าง ถ้าเป็น "0" หมายถึงเอาต์พุท ถ้าเป็น "1" หมายถึงอินพุท

การทำงานในโหมด 0

ในโหมด 0 เป็นโหมดที่กำหนดให้พอร์ททุกพอร์ทบนตัว 8255 เป็นอินพุทเอาต์พุทแบบพื้นฐาน รูปแบบความเป็นไปได้จึงมีทั้งสิ้น 16 รูปแบบตามลักษณะของพอร์ท A พอร์ท B พอร์ท C บนและพอร์ท C ล่าง ลักษณะของรหัสควบคุมแต่ละแบบจะเป็นดังรูปที่ 2.2.3

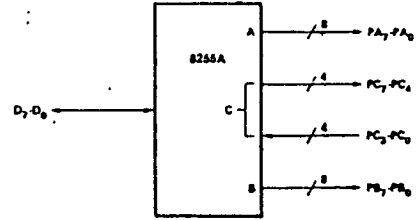
รหัสควบคุม # 0

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	0	0	0	0



รหัสควบคุม # 1

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	0	0	0	1



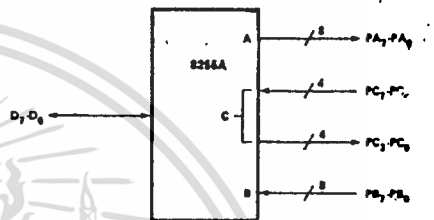
รหัสควบคุม # 2

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	0	0	1	0



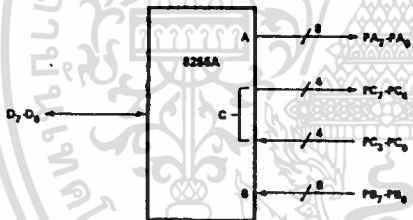
รหัสควบคุม # 6

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	0



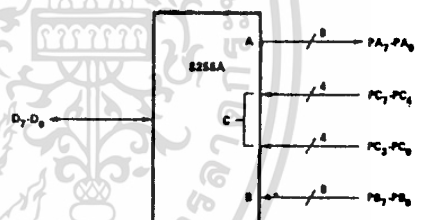
รหัสควบคุม # 3

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	0	0	1	1



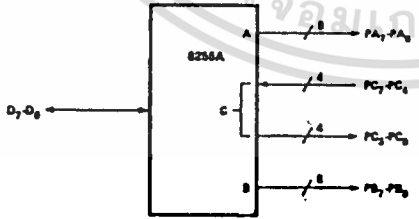
รหัสควบคุม # 7

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	1



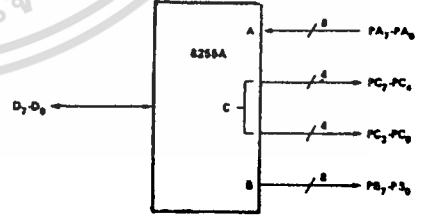
รหัสควบคุม # 4

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	0



รหัสควบคุม # 8

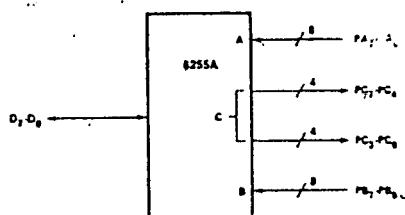
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	0



รูปที่ 2.2.3 ลักษณะของรหัสควบคุมแบบต่าง ๆ ในโหมด 0

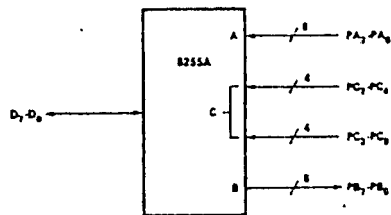
รหัสควบคุม #10

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	0



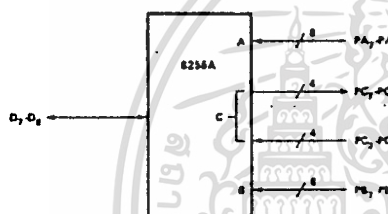
รหัสควบคุม #13

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	1	0	0	1



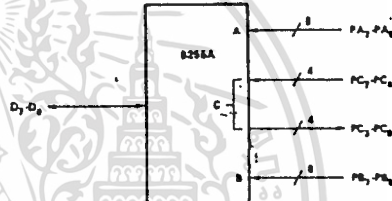
รหัสควบคุม #11

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	1



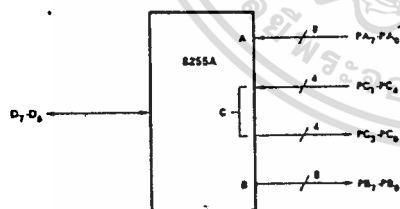
รหัสควบคุม #14

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	1	0	1	0



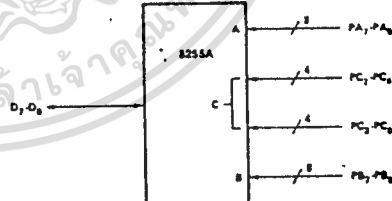
รหัสควบคุม #12

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	1	0	0	0



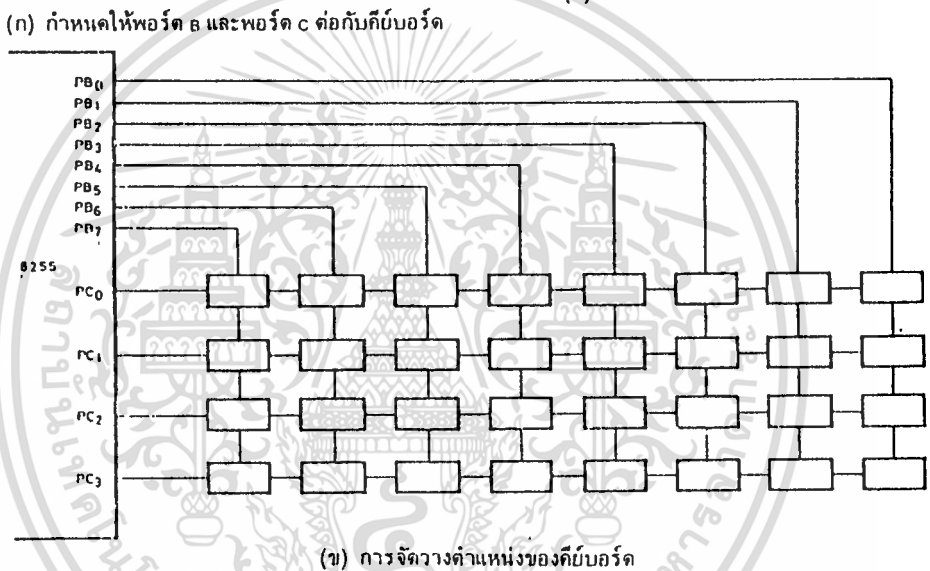
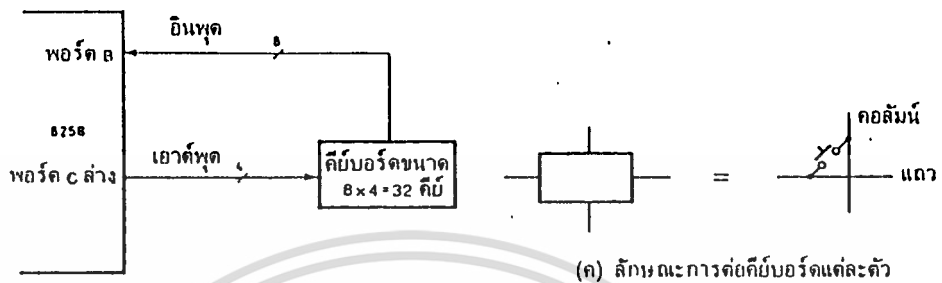
รหัสควบคุม #15

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	1	0	1	1



รูปที่ 2.2.3 (ต่อ) ลักษณะของรหัสควบคุมแบบต่าง ๆ ในโหมด 0

ตัวอย่างการใช้งานของ 8255 ในโหมด 0 นี้ เช่น เมื่อต้องการให้พอร์ท B เป็นพอร์ทอินพุตและพอร์ท C ส่วนเป็นพอร์ทเอาต์พุตเพื่อรับรู้การกดคีย์บอร์ดและหาดูว่ามีคีย์ใดกดซึ่งเราสามารถจัดการคีย์บอร์ดในรูปแบบเมตริกซ์ได้ดังรูปที่ 2.2.4



รูปที่ 2.2.4 วงจรการต่อคีย์บอร์ดขนาด 32 คีย์แบบเมตริกซ์

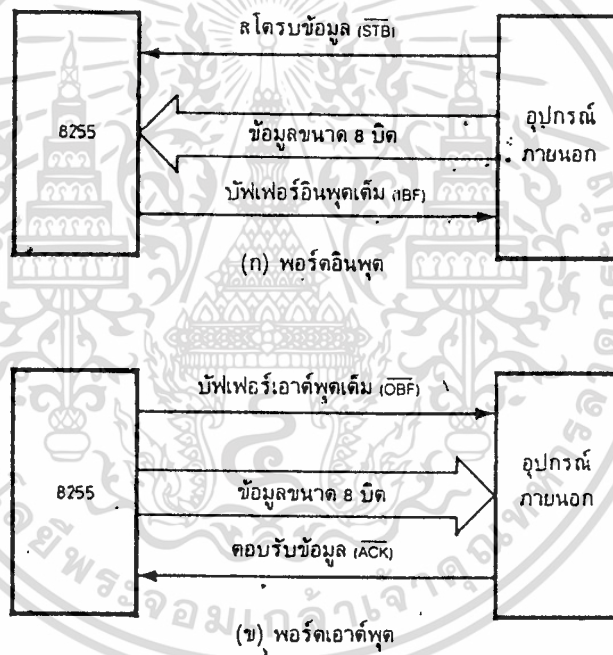
สาเหตุที่เรานิยมต่อคีย์บอร์ดจำนวนมาก แบบเมตริกซ์ เนื่องจากเป็นโครงสร้างทางฮาร์ดแวร์แบบประหยัดและช่วยลดข้อยุ่งยากต่าง ๆ ได้ แต่ก็ยังต้องใช้ซอฟต์แวร์ในการควบคุมหรือตรวจสอบว่าคีย์ใดกดในรูปที่ 2.2.4 เป็นการต่อแบบเมตริกซ์ ซึ่งมีจำนวนแถว 4 แถว และคอลัมน์ 8 คอลัมน์ทำให้ได้จำนวนคีย์ทั้งสิ้น 32 คีย์

หลักการทำงานทั่วไปเราจะทำการแสกนดู กล่าวคือกำหนดให้แต่ละแถวซึ่งเป็นพอร์ทเอาต์พุตเป็น "0" หรือ "1" ในเวลาต่างกันเช่นแถวแรกเป็น "0" แถวอื่นเป็น "1" หมด

แล้วทำการอ่านข้อมูลทีพอร์ทอินพุตว่า มีบิตใดบิตหนึ่งทางคอลัมน์เป็น "0" หรือไม่ ถ้ามีก็ทราบได้ว่า คีย์ในตำแหน่งแถวแรกและคอลัมน์ที่เท่าไรเป็นคีย์ที่รับการกด แต่ถ้าไม่มี ซีฟิยูก็จะสแกนไปยังแถวถัดไปและวนรอบไปเรื่อย ๆ ตลอดเวลาตั้งนั้นการตรวจสอบคีย์กจะทำให้ทราบว่า แถวหรือคอลัมน์ที่เท่าไรซึ่งเป็นคีย์ที่ได้รับการกด

การทำงานของ 8255 ในโหมด 1

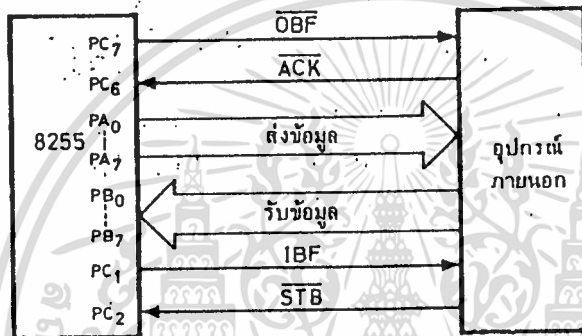
การทำงานของ 8255 ในโหมด 1 เป็นโหมดที่ทำให้อินพุตเอาต์พุต มีการตรวจสอบสัญญาณ(handshaking)โดยใช้อินพุตเอาต์พุตพอร์ท A และพอร์ท B เป็นหลัก และใช้พอร์ท C บนเป็นตัวตรวจสอบสัญญาณ(handshake) ของพอร์ท A ส่วนพอร์ท C ล่างเป็นตัวตรวจสอบสัญญาณของพอร์ท B การจัดสัญญาณต่าง ๆ เหล่านี้ แสดงได้ดังรูปที่ 2.2.5



รูปที่ 2.2.5 โครงสร้างตัวตรวจสอบสัญญาณของพอร์ทอินพุตและเอาต์พุต

แนวความคิดของการใช้พอร์ทอินพุตเอาต์พุต โดยมีการตรวจสอบสัญญาณก็เพื่อให้มีการซิงโครไนซ์ ระหว่างอุปกรณ์ภายนอกที่ทำงานได้ช้ากับการทำงานของคอมพิวเตอร์ที่ทำงานได้เร็ว เช่นเครื่องพิมพ์ที่ทำงานได้ช้า เมื่อคอมพิวเตอร์ส่งตัวอักษรตัวแรกมาพิมพ์ เครื่อง

พิมพ์รับตัวอักษรและกำลังจะพิมพ์ คอมพิวเตอร์ก็ส่งตัวอักษรตัวที่ 2 ตัวที่ 3 ตามมาทำให้การประมวลผลของอุปกรณ์เครื่องพิมพ์ทำงานไม่ทัน ซึ่งอาจทำให้ข้อมูลสูญหาย ดังนั้นเครื่องพิมพ์จึงส่งสัญญาณบอกคอมพิวเตอร์ว่า “อย่าเพิ่งส่งมาเพราะยังไม่พร้อมที่จะรับ” ลักษณะของการรับส่งข้อมูลอินพุตเอาต์พุตแบบมีตัวตรวจสอบสัญญาณ ดังรูปที่ 7.9 นั้น จะใช้ PA₀ - PA₇ เป็นเอาต์พุต และ PB₀ - PB₇ เป็นอินพุตโดยมีพอร์ท C เป็นตัวตรวจสอบสัญญาณ ดังแผนผังในรูปที่ 2.2.6



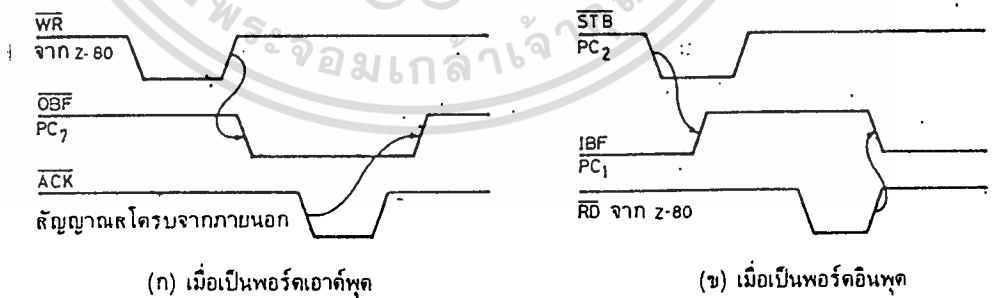
รูปที่ 2.2.6 วงจรการต่อ 8255 ในโหมด 1

เมื่อโปรแกรม 8255 เป็นโหมด 1 แล้ว ตัว 8255 จะใช้พอร์ท C เป็นสัญญาณควบคุม โดยแต่ละบิตของพอร์ท C เป็นไปตามที่กำหนดไว้ ดังตารางที่ 2.2.2

ขา	กรณีอินพุต	กรณีเอาต์พุต
PC ₀	INTR _B	INTR _B
PC ₁	IBF _B	OBF _B
PC ₂	STB _B	ACK _B
PC ₃	INTR _A	INTR _A
PC ₄	STB _A	I/O
PC ₅	IBF _A	I/O
PC ₆	I/O	ACK _A
PC ₇	I/O	OBF _A

ตารางที่.2.2.2 หน้าที่ของสัญญาณต่าง ๆ ของพอร์ท C ในการทำงานเป็นตัวตรวจสอบสัญญาณเมื่อ 8255 ทำงานในโหมด 1 โดยปกติ 8255 จะให้สัญญาณอินเตอร์รัพต์ไปบอกซีพียูด้วย สัญญาณอินเตอร์รัพต์ของ 8255 จะเกิดขึ้นที่ PC₀ และ PC₃ โดยที่เมื่อบัฟเฟอร์พร้อมแล้วและ ต้องการให้ซีพียูส่งอินพุตหรือเอาต์พุตมาที่บัฟเฟอร์ สัญญาณอินเตอร์รัพต์ก็จะเกิดขึ้น สังเกตว่าสัญญาณอินเตอร์รัพต์เป็นสัญญาณแอกทีฟ "1"

โครงสร้างการตรวจสอบสัญญาณของ 8255 แสดงด้วยสัญญาณทางไฟฟ้าดังรูปที่ 2.2.7



รูปที่ 2.2.7 แผนผังเวลาการรับและส่งข้อมูลโดยใช้ตัวตรวจจับสัญญาณ

สังเกตว่าการทำงานของ 8255 จะเกี่ยวกับสัญญาณ RD และสัญญาณ WR ซึ่งจะทำให้สัญญาณควบคุมเปลี่ยนแปลงไป การตรวจสอบสัญญาณซึ่งกันและกันนี้ เป็นวิธีการรับส่งที่มีประสิทธิภาพเช่น ในกรณีอินพุต เมื่ออุปกรณ์ภายนอกต้องการส่งข้อมูลให้ซีพียู ก็จะส่งข้อมูลแบบขนานเข้ามาพร้อมทั้งสโตรบ (STB) บอก 8255 ซึ่ง 8255 จะนำข้อมูลนั้นไปเก็บไว้ในรีจิสเตอร์ภายในก่อนแล้วส่งสัญญาณตอบบอกว่า“บัฟเฟอร์ยังเต็มอยู่นะ (IBF)อย่าเพิ่งส่งมาอีก”ครั้นเมื่อซีพียูอ่านข้อมูลจากรีจิสเตอร์ไปแล้ว ส่วนของสัญญาณบัฟเฟอร์อินพุต (IBF) ก็จะบอกว่า “ว่างแล้วส่งมาได้” อุปกรณ์ภายนอกก็จะส่งข้อมูลมาให้อีก

ทำนองเดียวกัน สำหรับพอร์ทเอาต์พุต เมื่อซีพียูส่งข้อมูลออกทางพอร์ทเอาต์พุตให้กับ 8255 ตัว 8255 ก็จะรับไว้ในรีจิสเตอร์ภายใน พร้อมทั้งส่งสัญญาณออกไปบอกอุปกรณ์ภายนอกว่า “เอาต์พุตบัฟเฟอร์ของฉันมีข้อมูลอยู่นะ (OBF) มาอ่านเอาไปซิ” อุปกรณ์ภายนอกเมื่อทราบและพร้อมจะอ่านข้อมูลก็จะส่งสัญญาณตอบรับ (ACK) พร้อมกับอ่านข้อมูลไป โดยสัญญาณ ACK จะมีความหมายว่า “ฉันอ่านข้อมูลของเธอไปแล้วนะ” ตัว 8255 ก็จะตอบกลับว่า “บัฟเฟอร์ฉันว่างแล้ว เธอรอก่อนนะจะมีข้อมูลใหม่ส่งมาให้อีก”

ในการที่จะโปรแกรมโหมด 1 นี้ เราจะใช้รหัสควบคุมเป็น 101(I/O) 01(I/O) ในส่วน I/O หมายถึง ถ้าเป็นอินพุตก็คือ “1” ถ้าเป็นเอาต์พุตก็คือ “0” โดย I/O ตัวแรกเป็นพอร์ท A ตัวที่ 2 เป็นพอร์ท B เช่นถ้าต้องการให้พอร์ท A เป็นเอาต์พุตและพอร์ท B เป็นอินพุต เราจะใช้รหัสควบคุมเป็น 10100110 หรือ A6H

จากการพิจารณาการทำงานของซีพียูจะเห็นว่า ทำอย่างใดจึงจะเขียนหรืออ่านพอร์ทได้ถูกต้อง วิธีที่ง่ายวิธีหนึ่งคือ ซีพียูจะคอยตรวจสอบสัญญาณของ 8255 เช่น กรณีเอาต์พุต ซีพียูจะคอยอ่านพอร์ท C แล้วตรวจสอบบิต 7 (OBF) หลังจากที่ส่งข้อมูลไปแล้ว ถ้าบิต 7 ยังเป็น “0” แสดงว่ายังไม่ได้รับการสโตรบ แต่ถ้าเป็น “1” แล้วแสดงว่าอุปกรณ์ภายนอกรับข้อมูลไปแล้วสำหรับกรณีอินพุตก็คอยตรวจสอบสัญญาณ (IBF) ได้เช่นกันว่า มีข้อมูลใหม่มาหรือยัง คือ ตรวจสอบบิต PC₁ ของพอร์ท C

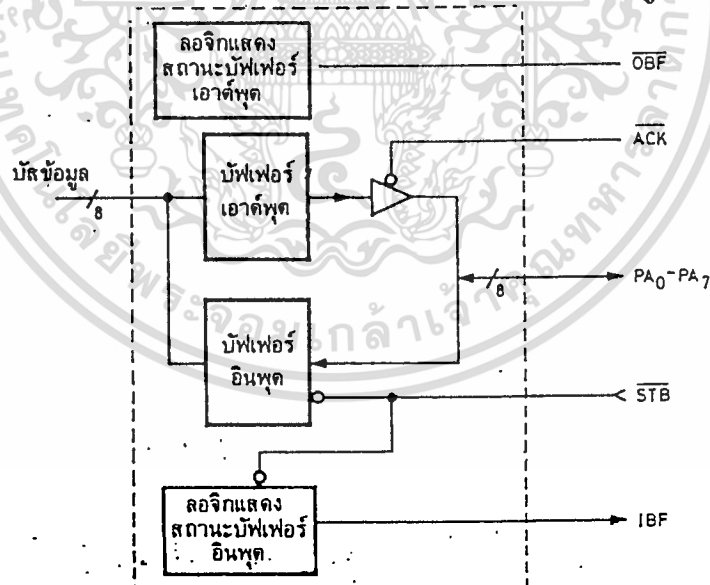
การทำงานของ 8255 ในโหมด 2

8255 ยังมีโหมดทำงานอีกหนึ่งโหมดคือ โหมด 2 ซึ่งทำได้เฉพาะพอร์ท A ในโหมดนี้ 8255 จะใช้พอร์ท A ทำหน้าที่เป็นพอร์ทแบบ 2 บิตทางคือสามารถเป็นได้ทั้งพอร์ทอินพุตและพอร์ทเอาต์พุตโดยโครงสร้างของพอร์ท A ทั้งอินพุตเอาต์พุตมีตัวตรวจสอบสัญญาณทั้งคู่ส่วนพอร์ท C จะทำหน้าที่เป็นสัญญาณตรวจสอบ โดยมีสัญญาณแต่ละขาดังตารางที่ 2.2.3

พอร์ท C	ความหมาย
PC ₀	I/O
PC ₁	I/O
PC ₂	I/O
PC ₃	INTR _A
PC ₄	STB _A
PC ₅	IBF _A
PC ₆	ACK _A
PC ₇	OBF _A

ตารางที่ 7.3 หน้าที่ของพอร์ท C ในโหมด 2

โครงสร้างของพอร์ท A ที่ทำงานแบบ 2 ทิศทาง แสดงได้ดังรูปที่ 2.2.8



รูปที่ 2.2.8 โครงสร้างของพอร์ท A ที่ทำงานแบบพอร์ทสองทิศทาง

สังเกตว่า เมื่อโปรแกรมพอร์ท A เป็นโหมด 2 แล้ว พอร์ท B จะต้องโปรแกรมเป็นโหมด 0 หรือโหมด 1 ก็ได้ ซึ่งก็ทำงานแบบแยกอิสระอีกในการใช้งานพอร์ทแบบ 2 ทิศทางนี้ใช้ได้กับการงานบางประเภท เช่น ในการรับส่งข้อมูลของพอร์ทมาตรฐานบางประเภท เช่น IEEE 488 หรือใช้เชื่อมโยงระหว่างคอมพิวเตอร์กับคอมพิวเตอร์ในการรับส่งข้อมูลสลับกันไปและกลับ



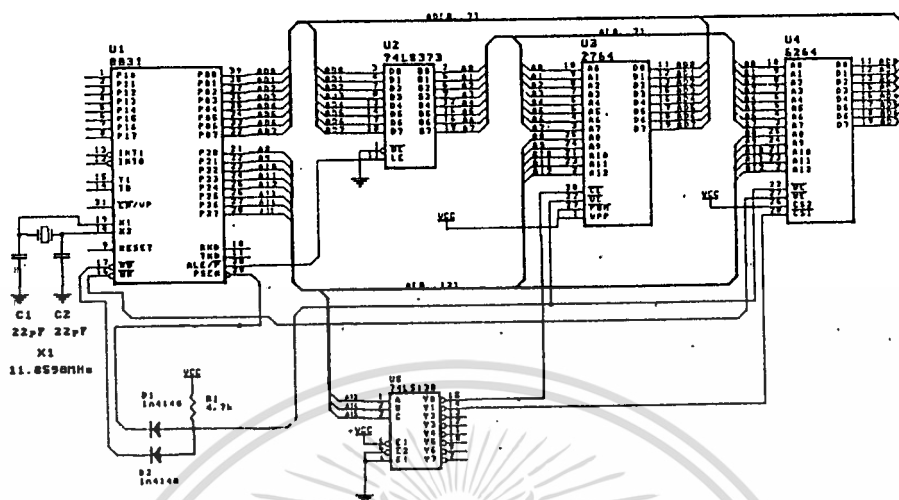
2.3 รายละเอียดเกี่ยวกับบอร์ดที่ใช้ในโครงงาน

- ชื่อบอร์ด "CP-SB31" (Controll Pack Single Board 8031)
- CPU เบอร์ 8031
- หน่วยความจำ ผู้ใช้สามารถเลือกใช้ได้อย่างอิสระตลอด 128 KB โดยการเลือกเซตจัมเปอร์ ให้ถูกต้อง (คู่มือเรื่อง)
- พอร์ต I/O 8255 1 ตัว
- อินเทอร์เฟซกับ LCD ได้ทันที
- พอร์ตสื่อสารข้อมูลอนุกรม 1 ช่อง
- อินเทอร์เฟซกับคีย์บอร์ดขนาด 4 x 4 ได้
- เพาเวอร์ซัพพลาย 5 โวลต์บนบอร์ด
- พัฒนาโปรแกรมร่วมกับ PC ผ่านพอร์ตอนุกรม RS-232
- ขนาดบอร์ด 7.6 x 14 ซม.

การต่อ CPU 8031 กับหน่วยความจำภายนอก

- ขา 31 ของ CPU (EA/VP) ถ้าต่อขานี้ลงกราวด์ จะเป็นการติดต่อกับโปรแกรมภายนอก ซึ่งจะมีผลต่อพอร์ต P0 และ P1 ดังนี้
- P0 (ขา 32-39) จะมีลิตเติ้ลเฟลิกซ์ระหว่าง DATA D0 - D7 และแอดเดรส A0 - A7
- P2 (ขา 21-28) จะเป็นแอดเดรส A8-A15 ในการใช้งานลักษณะนี้ พอร์ต P0 จะต้องมิตัวต้านทานพูลอัพด้วย
- ขา 30 (ALE/P) จะเป็นสัญญาณแอดเดรสแลตซ์อื่นาเบิ้ล ออกจากตัว CPU นอกอุปกรณ์ภายนอกให้ทำการแลตซ์แอดเดรสไบต์ต่ำที่ถูกส่งออกมาในขณะเฟลช์คำสั่ง
- ขา 17 (RD) เป็นขาสัญญาณการอ่านข้อมูลจากหน่วยความจำข้อมูลภายนอก
- ขา 16 (WR) เป็นขาสัญญาณการเขียนข้อมูลจาก CPU ไปหน่วยความจำข้อมูลภายนอก
- ขา 29 (PSEN : Program Store Enable) เป็นขาสัญญาณออกจาก CPU ขณะทำการเฟลช์สัญญาณจากหน่วยความจำโปรแกรม

แรมจาก ROM สัญญาณ PSEN จะแอดทิฟส่วนการอ่าน/เขียน RAM CPU จะใช้ (RD/WR) ในการติดต่อ จึงทำให้สามารถใช้หน่วยความจำในตำแหน่งเดียวกันได้



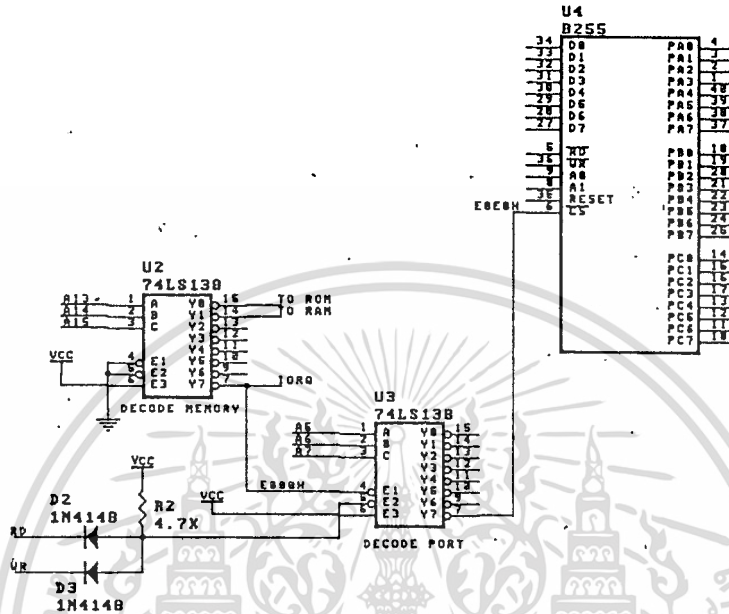
รูปที่ 2.3.3 การต่อ 8031 กับหน่วยความจำ โดยการรวมหน่วยความจำโปรแกรม และหน่วยความจำข้อมูลได้ในหน้าเดียวกัน

จากรูปที่ 2.3.3 เป็นการต่อ 8031 กับหน่วยความจำ ROM และ RAM โดยหน่วยความจำทั้งสองชนิดจะอยู่ในหน้าเดียวกัน และการต่อแบบนี้จะอ้างหน่วยความจำได้สูงสุด 64 KB วิธีการต่อทำได้โดยการนำสัญญาณ PSEN และ RD มา AND กัน แล้วจึงในรูปที่ 3 ใช้ไดโอดมาแทน แอนด์เกต เมื่อมีตำแหน่งหน่วยความจำอยู่ในหน้าเดียวกัน จำเป็นจะต้องกำหนดตำแหน่งของหน่วยความจำแต่ละตัว ว่าอยู่ที่ตำแหน่งใด วงจรในรูปที่ 2.3.3 จึงใช้ 74LS138 เป็นตัวดีโค๊ด โดย ROM มีตำแหน่งอยู่ที่ 0000-1FFFFH และ RAM ที่ตำแหน่งอยู่ที่ 2000-3FFFFH

ถึงแม้ว่าจะเหลือ พอร์ตอินพุต/เอาต์พุต ขนาด 8 บิตไว้ใช้ (P1) แต่ในบางครั้งก็ไม่เพียงพอกับความต้องการของผู้ใช้ จำเป็นต้องต่อพอร์ทยาวออกไปอีก บอร์ด CP-SB31 ที่ใช้นี้เลือกขยายด้วย 8255

ปัญหามีอยู่ที่ว่า 8031 ไม่มีสัญญาณ IORQ เหมือน Z80 ที่ใช้ติดต่อกับพอร์ตภายนอก ดังนั้นเราจึงใช้วิธี I/O map memory คือการให้ CPU มองหน่วยความจำตำแหน่งหนึ่งเป็นพอร์ต ในที่นี้เราใช้หน่วยความจำตำแหน่ง E000 H เป็นต้นไปเป็นตำแหน่งของ I/O และที่ตำแหน่ง E000 H นี้เราใช้สัญญาณ IORQ เทียม เพื่อให้บอร์ดที่เราใช้นี้สามารถเข้ากับ

อุปกรณ์ที่ต้องการสัญญาณ IORQ ด้วย ส่วนตำแหน่งของพอร์ทจะถูกกำหนดด้วยแอดเดรสไบต์ค่า A5-A7 โดยผ่าน 74LS138 เป็นตัวดีโค้ดเบอร์พอร์ทจากวงจรใช้การดีโค้ด 2 ชั้นแบบนี้ ในทางทฤษฎีแล้วอาจจะวิฤตไปบ้าง แต่ในทางปฏิบัติแล้วใช้งานได้



รูปที่ 2.3.4 การขยาย I/O พอร์ท ด้วย 8255

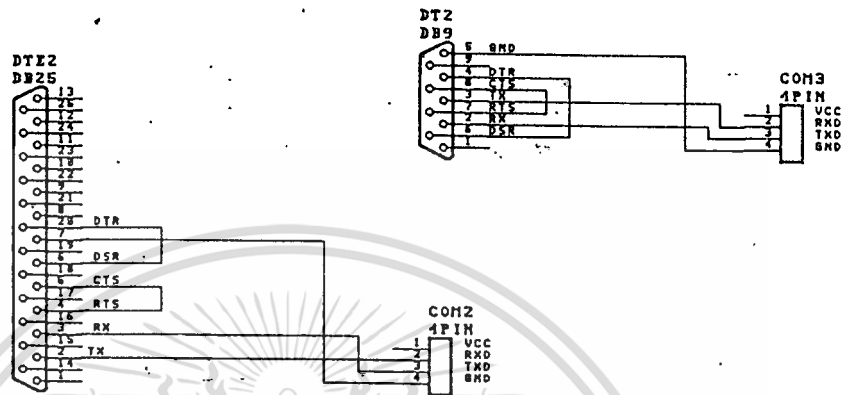
พอร์ทสื่อสารข้อมูลอนุกรม

8031 มีขา TX และ RX สำหรับส่งข้อมูลอนุกรมตามลำดับ ซึ่งอยู่ที่ขา 11 และขา 10 แต่เนื่องจากสัญญาณที่ขาทั้งสองนี้อยู่ในระดับ TTL แต่สัญญาณที่ช่อง RS-232 อยู่ในช่วง ± 15 โวลต์ ฉะนั้นจึงต้องมีวงจรอินเวอร์ตเฟส เพื่อจัดระดับสัญญาณให้เหมาะสมจากวงจรสมบูร์ณในรูป 5 จะได้ระดับสัญญาณประมาณ ± 8 โวลต์ซึ่งก็เพียงพอกับการสื่อสารข้อมูลในระยะที่ไม่ไกลนักจากการทดลองสามารถทำงานที่ Baud Rate 115200 ได้

สร้างสายเชื่อมต่อ RS-232 กับ CP-SB31

การสื่อสารข้อมูลอนุกรมระหว่าง CP-SB31 กับ PC นั้นเราใช้สัญญาณเพียง 3 เส้น คือ TX, RX และ GND แต่ในการส่ง-รับข้อมูลของ PC นั้นโปรแกรมจะต้องทำแฮนด์เช็กทางฮาร์ดแวร์ด้วย แต่เนื่องจาก CP-SB31 ไม่มีสัญญาณที่จะทำแฮนด์เช็กโดยตรง เราจึงจำเป็นต้อง

หลุด PC กันอีกแล้ว ด้วยการทำการป้อนกลับสัญญาณนี้ให้ตัวเองโดยต่อขา RTS เข้ากับขา DSR เข้ากับ DTR (ขา 6 กับขา 20)



รูปที่ 2.3.5 การต่อสัญญาณ RS-232C

รูปที่ 2.3.6 เป็นวงจรสมบูร์กของบอร์ด CP-SB31 ซึ่งจะแตกต่างกับวงจรที่อธิบายมาข้างต้นบ้างเพื่อความเหมาะสมและใช้งานจริง

หมายเหตุ จากวงจรในรูปที่ 2.3.6 IC₃ (U₃) และ IC₄ (U₄) ผู้ใช้สามารถเลือกตำแหน่ง และขนาดของหน่วยความจำได้อย่างอิสระ แต่ถ้าจะไม่ให้เกิดปัญหา การซ้อนทับของหน่วยความจำขึ้น เมื่อขนาดของ IC₃ (U₃) และ IC₄ (U₄) ไม่เท่ากันให้ใส่ไดโอด 1N418 หรือ 1N60 แทนจัมเปอร์โดยต่อทางด้านแคโทดเข้ากับ 74LS156 (IC₇ หรือ U₇)

รายการอุปกรณ์

ตัวต้านทานขนาด 1/4 w 5%

อุปกรณ์สารกึ่งตัวนำ

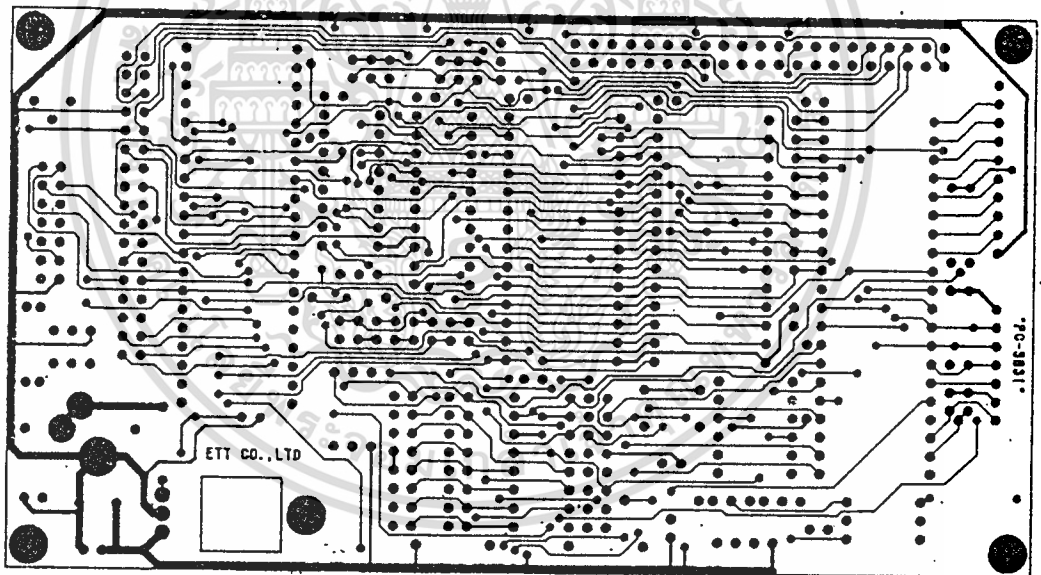
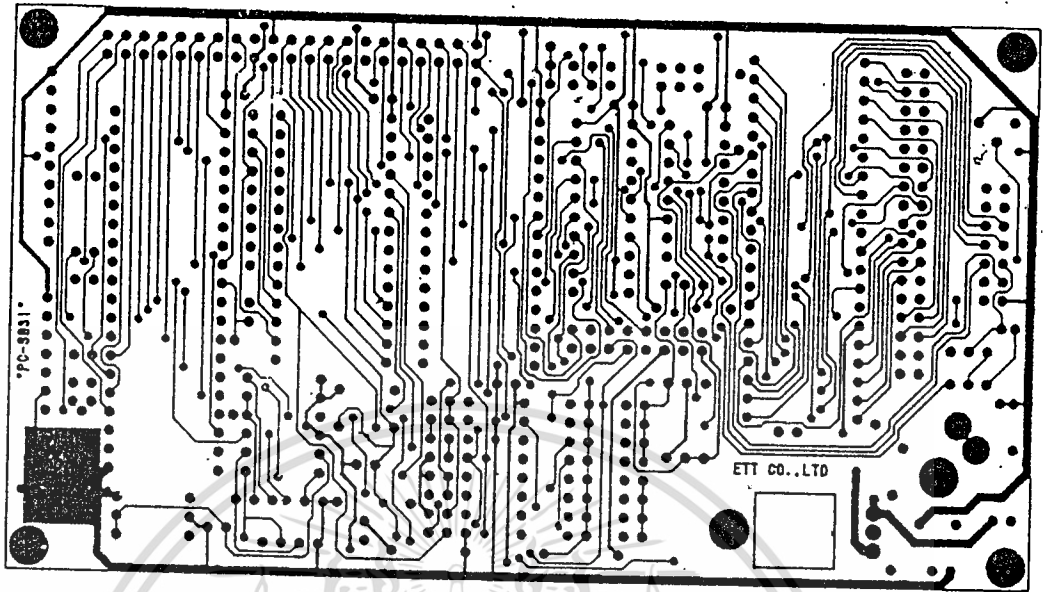
R1, R2, R4, R5, R7, R8, R9, R16	4.7	K	IC1	8031 CPU
R3	100		IC2	74LS373
R6	1.8	K	IC3, IC4	RAM
R10, R12, R17, R20	10	K	IC5	8255
R13	270		IC6	74LS138
R14	47	K	IC7	74LS156
R15	1	K	IC8	7805
R PACK, 1	*8	10	K	Q1 BC557

Q2 BC547

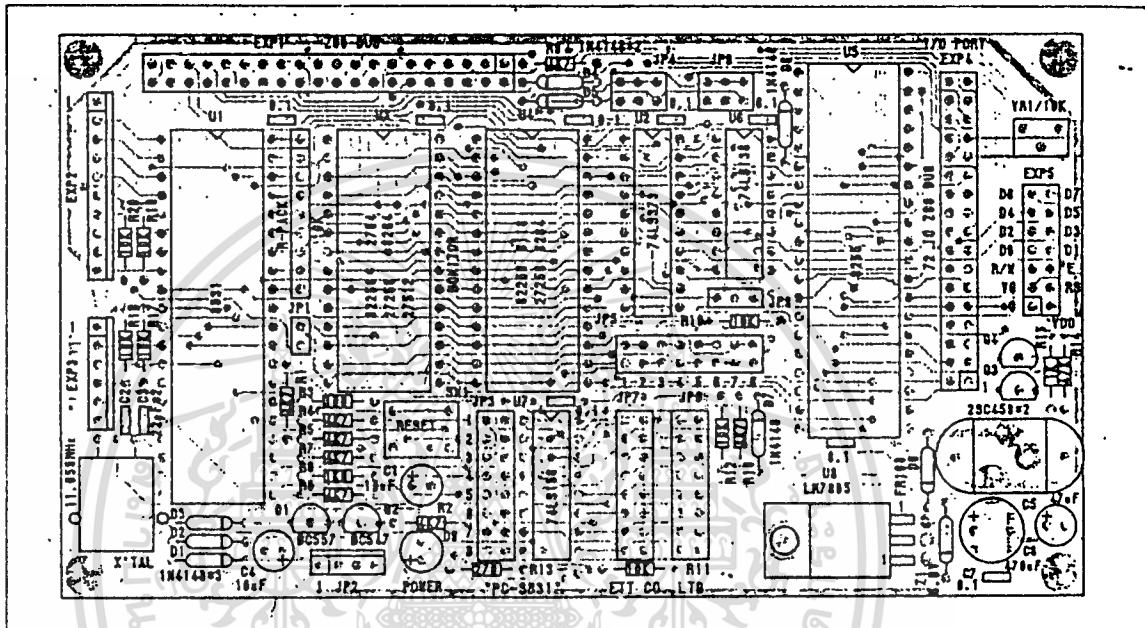
Q3, Q4 2SC 458

ตัวเก็บประจุ

C1	ELECTROLYTE	10	uF	16 V	D1-D7	1N4148
C2, C3	CERAMIC	22	pF	50 V	D8	FR100
C4	TANTALUM	10	uF	16 V	D9	LED
C5	ELECTROLYTE	47	uF	16 V	ZD1	5.6 V 1/2 W
C6	:	470	uF	16 V	อื่นๆ	
C7-C14	MULTILAYER	0.1	uF	50 V	JACK ADAPTER, HEADER	
					JUMPPER, MINISWITCH	
					PRINT CIRCUIT, ...	



รูปที่ 2.3.7 ลายวงจรพิมพ์ขนาดเท่าแบบทั้ง 2 ด้าน ซึ่งเป็นชนิด 2 หน้า เฟลตทูลูโซลด์



รูปที่ 2.3.8 ตำแหน่งการวางอุปกรณ์ของบอร์ด CP-SB31

การติดตั้งหน่วยความจำให้กับ PC-SB31

เนื่องจาก PC-SB31 ถูกสร้างมาให้อิสระในการเลือกใช้หน่วยความจำได้หลายขนาด ทั้ง EPROM และ RAM รวมทั้งตำแหน่งแอดเดรสของหน่วยความจำ ผู้ใช้ก็สามารถกำหนดได้ตามต้องการ ซึ่งทั้งหมดนี้ขึ้นอยู่กับตำแหน่งของจัมเปอร์ต่างๆ ให้ถูกต้องซึ่ง IC₃ (U₃) และ IC₄ (U₄) จะถูกควบคุมด้วยจัมเปอร์ ซึ่งมีรายละเอียดดังนี้

JP₃ เลือกเบอร์ของชิพหน่วยความจำที่ใส่อยู่บน IC₃ (U₃) (2764, 27256, 27512, 6264, 62256)

JP₄ สำหรับเลือกจะให้หน่วยความจำที่ IC₃ (U₃) หน่วยความจำข้อมูลหรือหน่วยความจำโปรแกรมหรือทั้ง 2 อย่าง

JP₅ เลือกเบอร์ของชิพหน่วยความจำที่ใส่อยู่บน IC₄ (U₄) (27256, 6116, 6264, 62256)

JP₆ เลือกลักษณะการทำงานของ IC₄ (U₄) ว่าจะให้เป็นหน่วยความจำข้อมูลหรือหน่วยความจำโปรแกรมหรือทั้ง 2 อย่าง

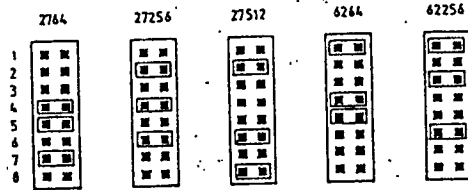
JP₇ เลือกตำแหน่งเริ่มต้นและขนาดของหน่วยความจำ IC₃ (U₃)

JP₈ เลือกตำแหน่งเริ่มต้นและขนาดของหน่วยความจำ IC₄ (U₄)

JP₉ เลือกว่าจะอนุญาตให้มีการใช้ I/O พอร์ตภายนอกอีกหรือไม่ ถ้าไม่มีพอร์ตภายนอก IC₄ (U₄) จะมีขนาดสูงสุดได้ถึง 32 KB

รายละเอียดที่กล่าวมานี้ได้จากรูปที่ 9

JP3 สำหรับ IC₃ (U₃)



JP5 สำหรับ IC₃ (U₄)



JP7 & JP8 เลือกตำแหน่ง
เริ่มต้นและขนาดของหน่วยความจำ



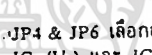
6116



6264



62256



27256

JP4 & JP6 เลือกลักษณะของ
IC₃ (U₃) และ IC₄ (U₄)



COMBINE



CODE



DATA

DATA = หน่วยความจำข้อมูล

CODE = หน่วยความจำโปรแกรม

COMBINE = เป็นทั้งหน่วยความจำข้อมูล
และหน่วยความจำโปรแกรม

■	E000-FFFF
■	C000-DFFF
■	A000-BFFF
■	8000-9FFF
■	6000-7FFF
■	4000-5FFF
■	2000-3FFF
■	0000-1FFF

แต่ละจุดของตัวจัมพ์ที่ใส่ลงไปจะมี
ขนาดของหน่วยความจำ 8 KB เพิ่มขึ้น
จากจุดเริ่มต้น



PORT



MEMORY

รูปที่ 2.3.9 การใส่ JUMPER ให้ถูกต้องตามเบอร์ของชิพหน่วยความจำ

บทที่ 3

รายละเอียดเกี่ยวกับโครงการ

ชื่อโครงการ

เครื่องหยอดของเหลวโปรแกรมได้

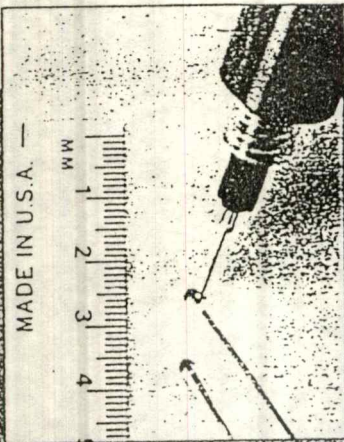
PROGRAMMABLE LIQUID DISPENSER

ความสำคัญและที่มาของโครงการ

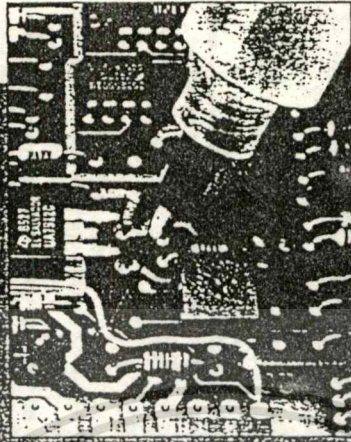
ในปัจจุบันบริษัทอิเล็กทรอนิกส์และโรงงานอุตสาหกรรมต่าง ๆ โดยทั่วไป ในสายงานการผลิต มีการหยอดของเหลวลงบนงานในลักษณะต่าง ๆ กัน ซึ่งของเหลวที่หยอดก็มีหลายประเภท เช่น กาว, น้ำมันซิลิโคน, อีพ็อกซี, ครีมนัดกรี, สารหล่อลื่น จึงมีผู้ผลิตเครื่องหยอดของเหลวเพื่ออำนวยความสะดวกในสายงานการผลิต ทำให้งานการผลิตรวดเร็วขึ้น คุณภาพดีขึ้น ลดของเสียให้น้อยลง

แต่เครื่องหยอดของเหลวที่มีขายอยู่ในท้องตลาดโดยทั่วไป ก็มีข้อจำกัดของขอบเขตการทำงาน คือ เครื่องหยอดของเหลวที่มีขายในท้องตลาด สามารถกำหนดค่าเวลาของการเปิดปิดโซลินอยด์วาล์ว หรือกล่าวอีกนัยหนึ่งก็คือ การกำหนดขนาดของของเหลวที่หยอดได้เพียงขนาดเดียวต่อการตั้งค่าเวลาแต่ละครั้ง แต่งานสายงานการผลิตจริงนั้นบางครั้งมีขนาดของจุดที่จะหยอดของเหลวแตกต่างกัน มากมายหลายขนาดและหลายจุด ซึ่งถ้าเป็นเครื่องหยอดของเหลวที่มีขายอยู่ในท้องตลาดโดยทั่วไปจะมีความยุ่งยากในการทำงานมาก คือต้องคอยเปลี่ยนค่าเวลาของการเปิดปิดของโซลินอยด์วาล์วบ่อยครั้ง ในการผลิตชิ้นงานแต่ละชิ้น ซึ่งทำให้เป็นการเสียเวลาด้วย และที่สำคัญอีกประการหนึ่งก็คือ ขนาดของของเหลวที่หยอดอาจไม่ได้ตามมาตรฐานที่กำหนด เนื่องจากการที่พนักงานเป็นผู้ปรับแต่งค่าเวลา

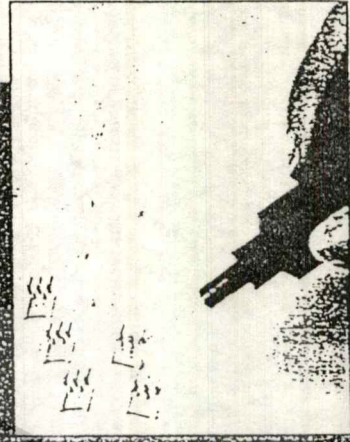
ดังนั้น คณะผู้จัดทำโครงการจึงได้นำเสนอโครงการที่ชื่อว่าเครื่องหยอดของเหลวโปรแกรมได้หรือในชื่อภาษาอังกฤษว่า PROGRAMMABLE LIQUID DISPENSER ขึ้นเพื่อตอบสนองความต้องการของบริษัทอิเล็กทรอนิกส์และโรงงานอุตสาหกรรมทั่วไป เช่น บริษัทซีเทค เทคโนโลยี ประเทศไทย จำกัด ฯลฯ



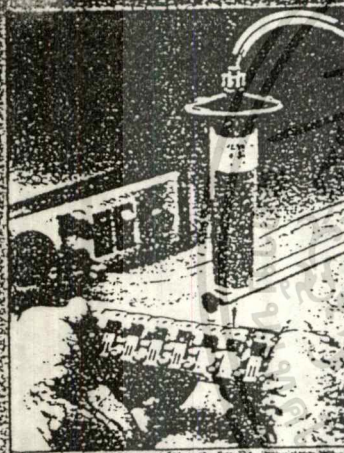
ใช้หยอดอีพ็อกซีที่ปลายเทอร์โมเซต ด้วย
ปริมาณที่พอเหมาะ



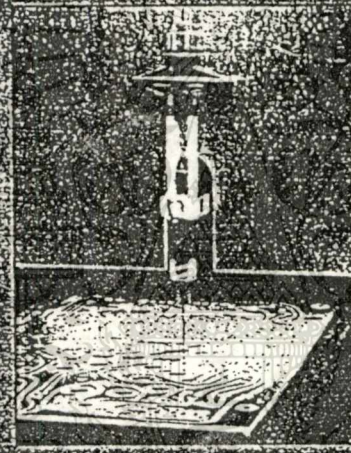
ใช้หยอดกาวเพื่อยึดสายจัมพ์กับแผ่นวงจร
พิมพ์



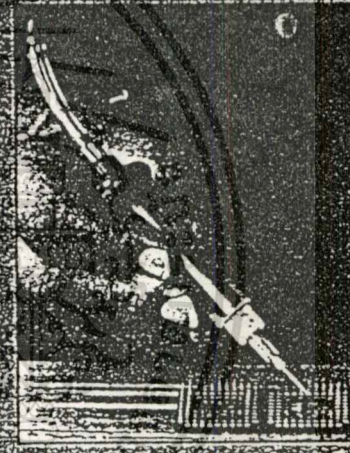
ใช้หยอดเรซินที่โคนรางของดีทอลด์



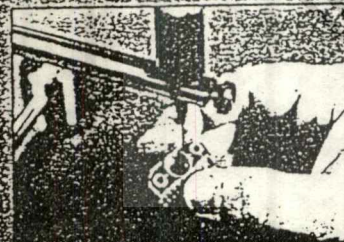
ใช้หยอดทองเหลืองที่ปลายเทอร์โมเซต
โดยเร็ว



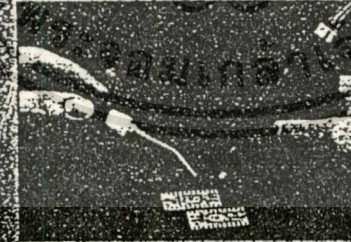
ใช้หยอด solder mask บนแผ่นวงจรพิมพ์
สำหรับประกอบเครื่องดนตรี



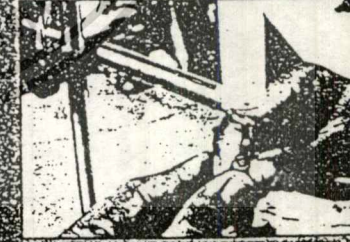
ใช้หยอด glue coming silicone ที่
ร่องราง IC บน PCB



ใช้หยอดจาระบีในงานผลิตตัวพิมพ์บน

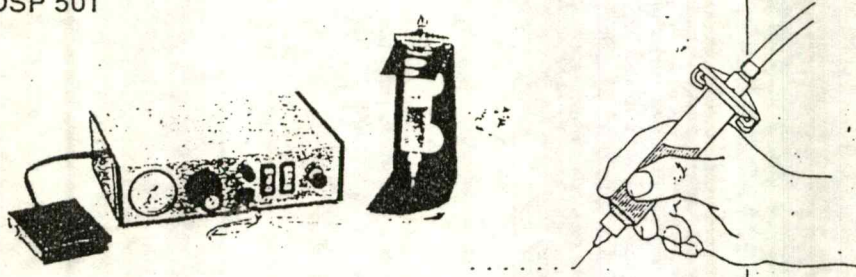


ใช้หยอดของเหลวและวางอุปกรณ์ประเภท
surface mount บน pcb

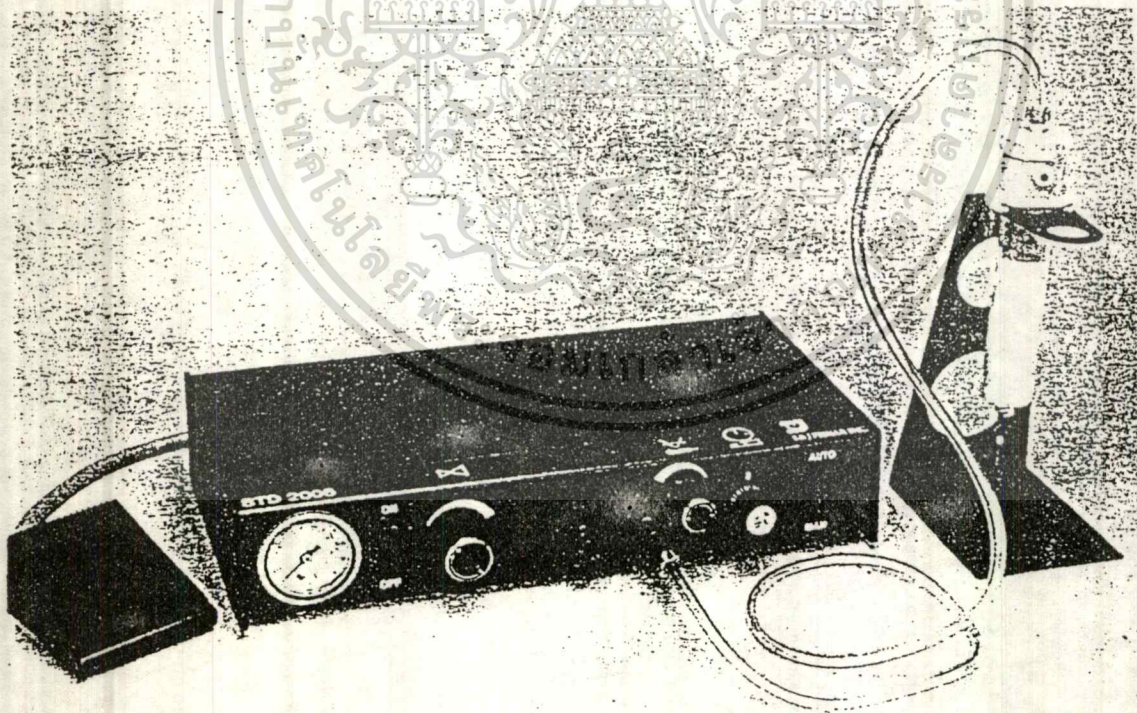
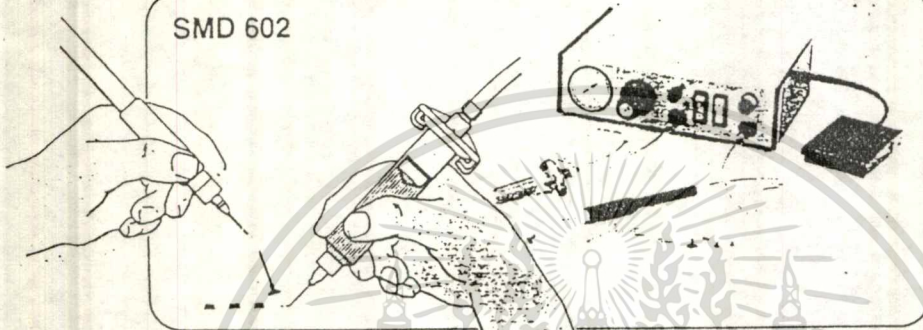


ใช้หยอดยางซิลิโคนเพื่อปิดในสวิตช์กับน้ำ

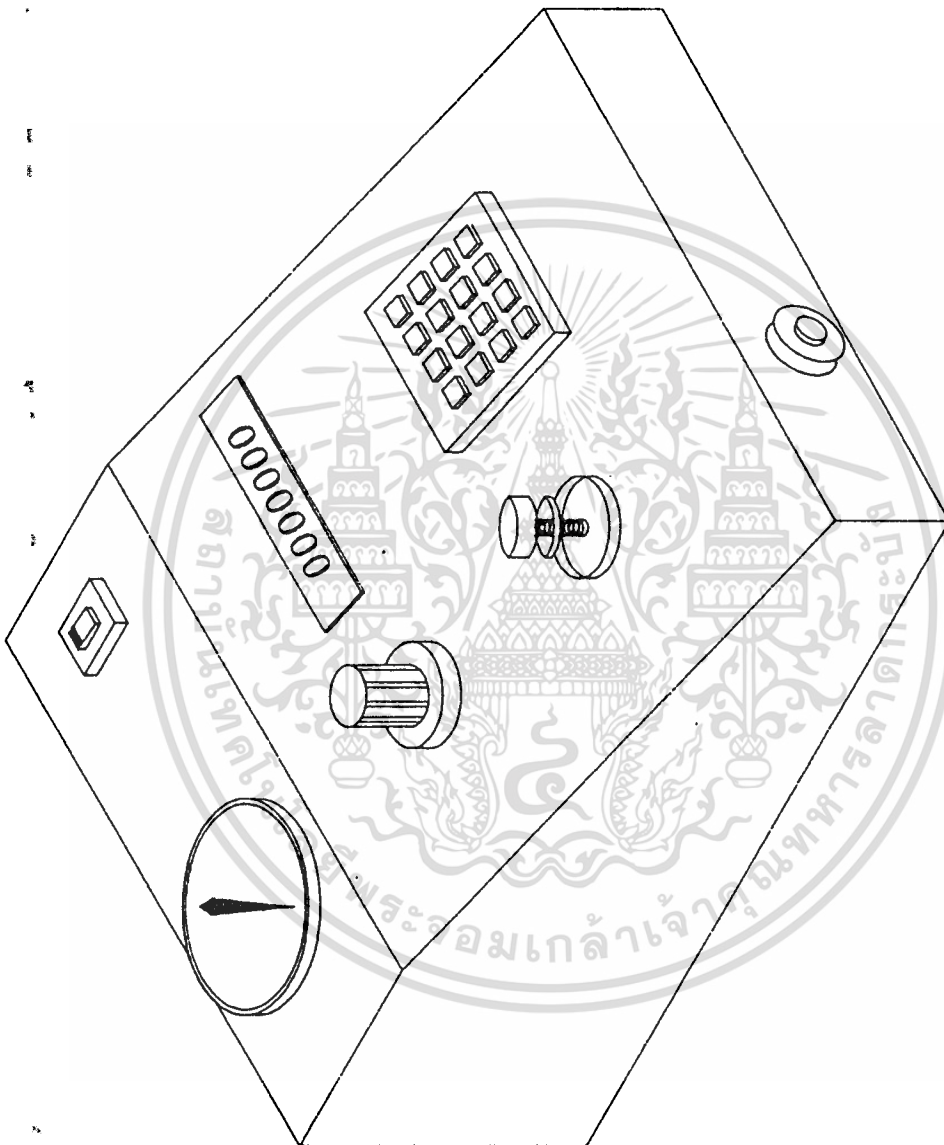
DSP 501



SMD 602



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



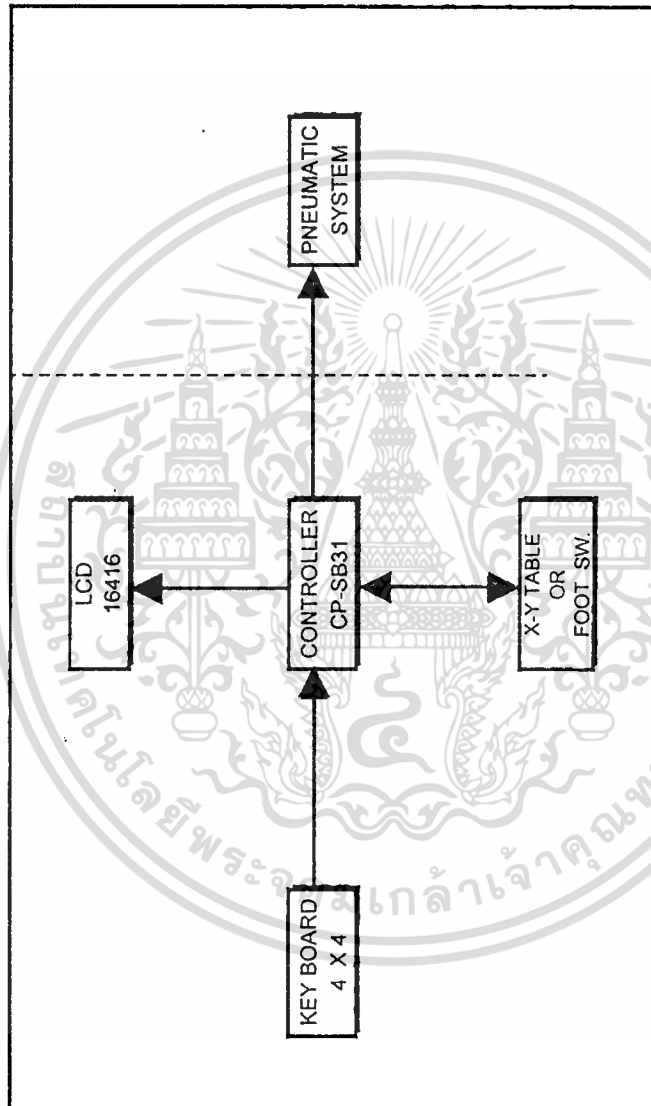
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษานี้เท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านกรทำ
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนประกอบและหลักการทำงานของโครงงาน

โครงงานนี้ ประกอบด้วย 2 ส่วนใหญ่ ๆ คือ

1. ระบบนิวแมติกส์ (Neumatic system)
2. ระบบควบคุม (Controller system)





CONTROLLER SYSTEM

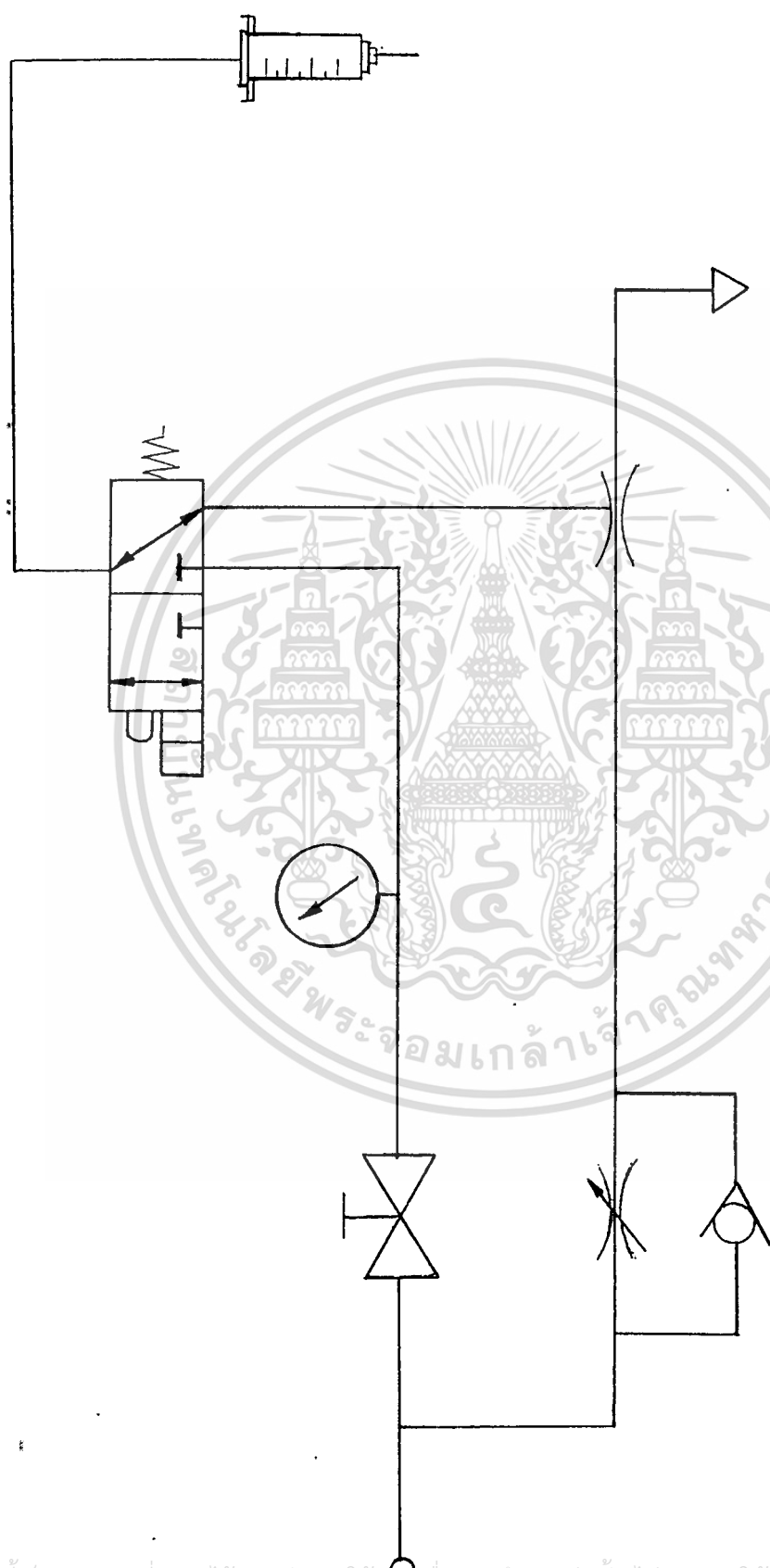
PNEUMATIC SYSTEM

BLOCK DIAGRAM OF PROGRAMMABLE LIQUID DISPENSER

ระบบนิวแมติกส์

ส่วนของระบบนิวแมติกส์ประกอบด้วยอุปกรณ์ต่าง ๆ ดังวงจรนิวแมติกส์ข้างล่างนี้ซึ่งอุปกรณ์หลักของส่วนนี้ก็คือ โซลินอยด์วาล์วที่เป็นวาล์ว 3/2 ทำหน้าที่เปิดปิดลมที่จะเข้าไปดันของเหลวในหลอดฉีดของเหลว



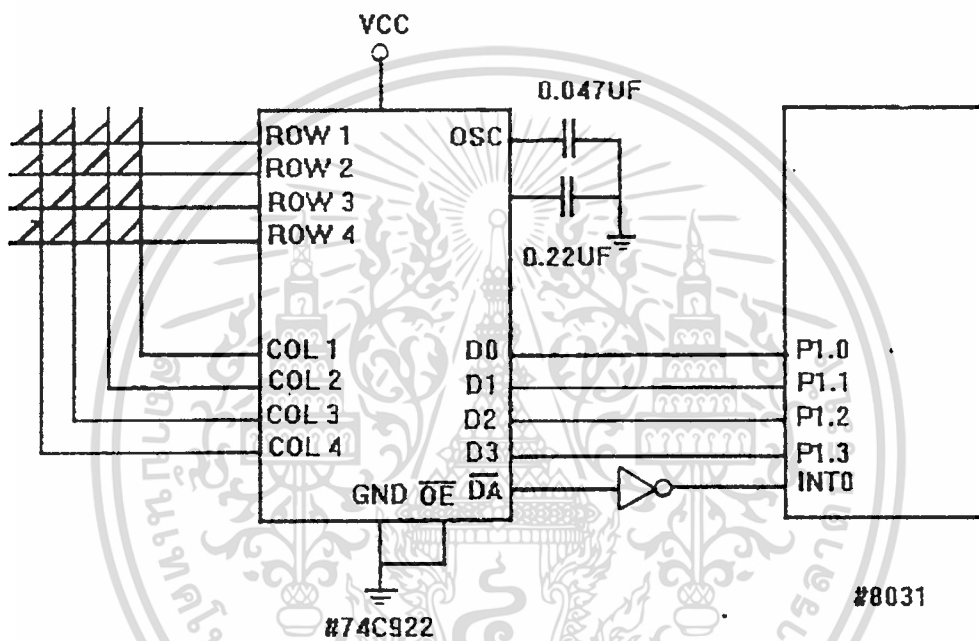


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

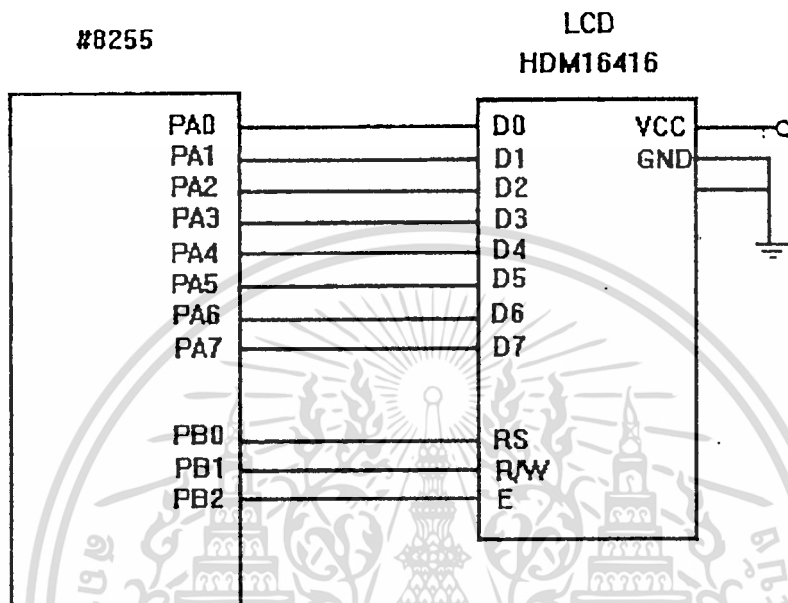
ระบบควบคุม

ส่วนของระบบควบคุมแบ่งออกเป็น 2 ส่วน คือ

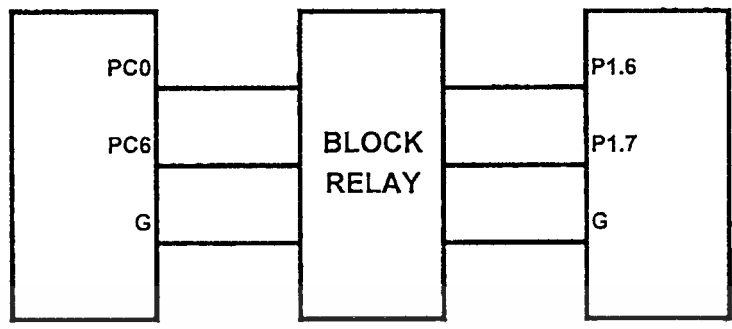
1. ส่วน Hard ware ซึ่งจาก Block Diagram จะเห็นได้ว่ามีส่วนของ บอร์ดควบคุม ซึ่งเป็นบอร์ดสำเร็จรูปชื่อ CP-SB 31 โดยใช้ไมโครคอนโทรลเลอร์ MCS-51 เบอร์ 8031 เป็นตัวหลักในการควบคุม ซึ่งรายละเอียดของบอร์ดได้อธิบายไว้ในหัวข้อที่ 2.3 บอร์ดควบคุมอินเตอร์เฟสกับคีย์บอร์ด เพื่อใช้ในการป้อนโปรแกรมค่าเวลาต่าง ๆ และกำหนดฟังก์ชันการทำงานของเครื่องอีกส่วนหนึ่งที่อินเตอร์เฟสกับบอร์ดควบคุมก็คือ จอ LCD เพื่อสำหรับการแสดงผลการทำงานในสถานะต่าง ๆ ของเครื่อง
2. ส่วน Soft Ware เป็นโปรแกรมที่เขียนด้วยภาษาแอสเซมบลี โดยเขียนโปรแกรมให้ Hard ware ทำหน้าที่เป็น timer และมีรายละเอียดของฟังก์ชันต่าง ๆ ดังที่จะแสดงไว้ในส่วนของวิธีการใช้เครื่อง ฯ



การต่อชุดคีย์บอร์ด

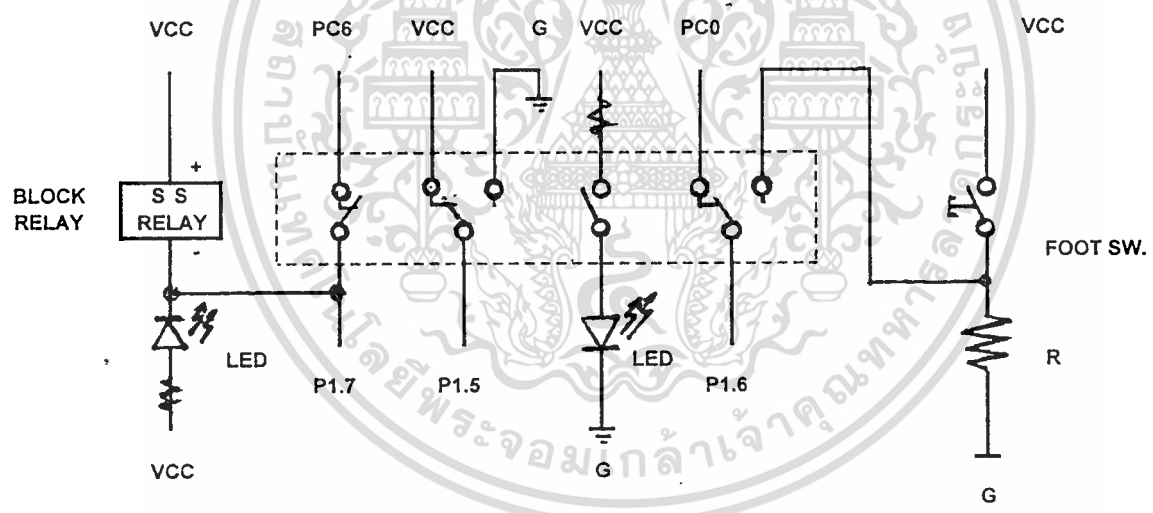


การต่อ LCD เข้ากับพอร์ต 8255



(ET-PC) 8255

8031



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

FUNCTION KEY

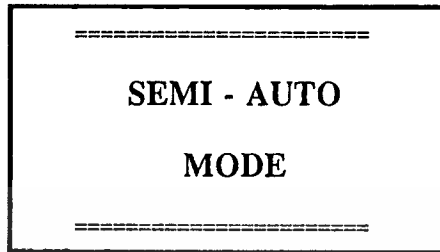
0	1	2	3
4	5	6	7
8	9	PROG	STEP
TIME	ENTER	MENU	RESET

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

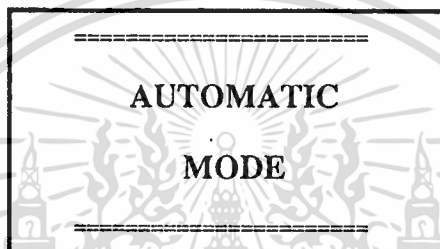
ขั้นตอนการทำงานและการใช้งานเครื่อง

1 เมื่อเริ่มเปิดเครื่อง เครื่องจะแสดง Title และสถานะของเครื่องว่าอยู่ในโหมดใด

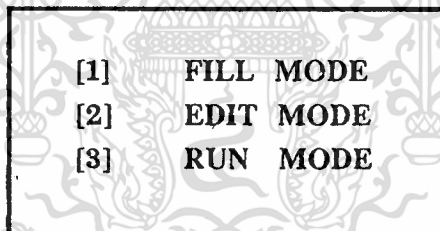
SEMI MODE



AUTOMATIC MODE



2 เมื่อแสดง Title และสถานะแล้ว หน้าจอจะแสดง MENU MODE



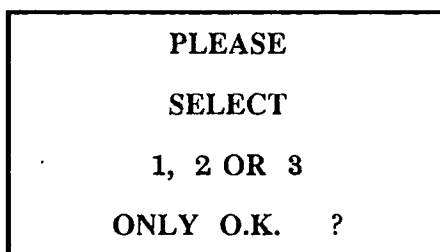
เพื่อให้เลือก โหมดการทำงาน คือ

ถ้ากดหมายเลข 1 จะเข้าสู่ FILL MODE คือการเก็บค่าเวลาไว้ในโปรแกรมต่าง ๆ

ถ้ากดหมายเลข 2 จะเข้าสู่ EDIT MODE คือ การแก้ไขค่าเวลาใน STEP ต่าง ๆ ในแต่ละโปรแกรม

ถ้ากดหมายเลข 3 จะเข้าสู่ RUN MODE คือ การให้เครื่องทำงานตามขบวนการผลิตที่กำหนด

แต่ถ้ากดคีย์อื่นที่ไม่ใช่ 1, 2 หรือ 3 จอจะแสดง



เพื่อเตือนว่ากดคีย์ผิด ให้กดคีย์ MENU เพื่อเข้าสู่โหมดเมนู เพื่อป้อนหมายเลข 1, 2 หรือ 3 เท่านั้น
3 เมื่อกดคีย์หมายเลข 1 จาก MENU MODE เครื่องจะเข้าสู่ FILL MODE เพื่อทำการเก็บค่าเวลา

=> FILL MODE <=	
PROGRAM.....	00
STEP.....	00
TIME.....	000

เมื่อโปรแกรมเข้าสู่การทำงานใน FILL MODE แล้วจอ LCD จะแสดงดังรูปในขั้นตอนนี้จะสามารถป้อนค่าต่าง ๆ ตามวิธีการดังนี้

- 3.1 ป้อนหมายเลข PROGRAM ที่ต้องการจะเก็บค่าเวลา 2 หลัก แล้วกด ENTER
 - 3.2 ป้อนค่าเวลา 2 หลัก แล้วกด ENTER โดยจะเริ่มเรียงจาก STEP ที่เพิ่มขึ้นใหม่
 - 3.3 STEP จะเพิ่มขึ้น 1 STEP โดยอัตโนมัติ เพื่อรอการป้อนค่าเวลาใน STEP ที่เพิ่มขึ้นใหม่
 - 3.4 ป้อนค่าเวลา 2 หลัก ใน STEP ที่เพิ่มขึ้นใหม่
 - 3.5 ถ้าป้อนค่าเวลาตามขบวนการผลิตแต่ละ STEP ครบแล้ว ให้จบที่ STEP สุดท้ายเป็นค่า 00
- 4 เมื่อกดคีย์หมายเลข 2 จาก MENU MODE เครื่องจะเข้าสู่ EDIT MODE เพื่อทำการแก้ไขค่าเวลาใน STEP ต่าง ๆ ของแต่ละโปรแกรม

=> EDIT MODE <=	
PROGRAM.....	00
STEP.....	00
TIME.....	000

เมื่อโปรแกรมเข้าสู่การทำงานใน EDIT MODE แล้วจอ LCD จะแสดงดังรูปในขั้นตอนนี้จะป้อนค่าต่าง ๆ ตามวิธีการดังนี้

- 4.1 ป้อนหมายเลข PROGRAM ที่ต้องการจะแก้ไขค่าเวลา 2 หลัก แล้วกด ENTER
 - 4.2 ป้อนหมายเลข STEP ที่ต้องการจะแก้ไขค่าเวลา 2 หลัก แล้วกด ENTER
 - 4.3 ป้อนค่าเวลาที่ต้องการจะแก้ไข 2 หลัก แล้วกด ENTER เพื่อเก็บค่าเวลาไว้ใน PROGRAM, STEP ที่ต้องการ
 - 4.4 เมื่อป้อนค่าเวลา และกด ENTER แล้วเคอร์เซอร์จะไปกระพริบที่ STEP เพื่อให้ป้อนค่า STEP ที่ต้องการจะแก้ไขต่อไป
 - 4.5 เมื่อไม่ต้องการแก้ไขค่าเวลาอีก ให้กดคีย์ MENU เพื่อเข้าสู่ MENU MODE
- 5 เมื่อกดคีย์หมายเลข 3 จาก MENU MODE เครื่องจะเข้าสู่ RUN MODE เพื่อให้เครื่องทำงานตามขบวนการผลิต

=> RUN MODE <=	
START PROG.....	00
STOP PROG.....	00
ENTER TO START	

เมื่อโปรแกรมเข้าการทำงานใน RUN MODE แล้วจอ LCD จะแสดงดังรูปในขั้นตอนนี้จะป้อนค่าต่าง ๆ ตามวิธีการดังนี้

5.1 ป้อนหมายเลข PROGRAM เริ่มต้น 2 หลัก

5.2 ป้อนหมายเลข PROGRAM สิ้นสุด 2 หลัก

5.3 กด ENTER เพื่อให้เครื่องทำงานตามขบวนการผลิต คือรอรับ I/P จาก FOOT SWITCH หรือ

X - Y TABLE

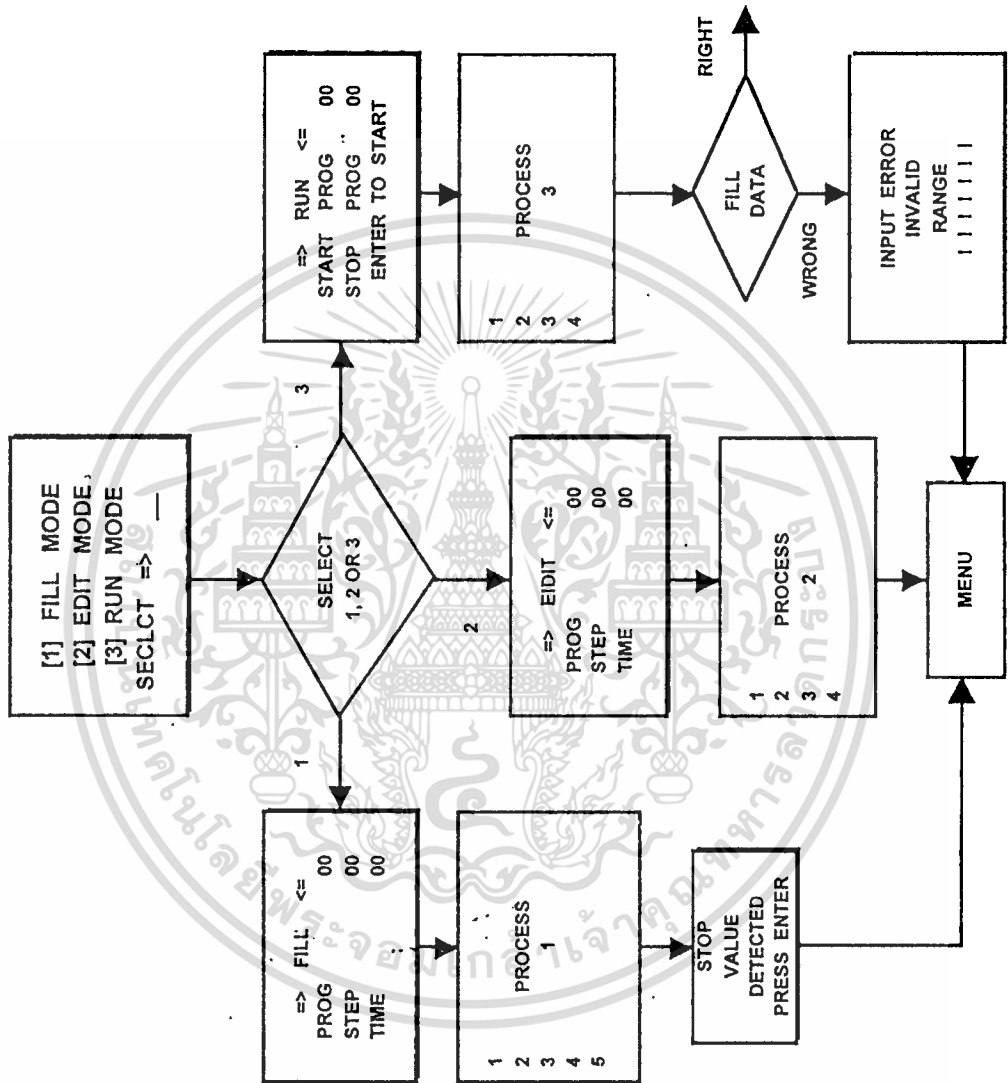
ตัวอย่าง

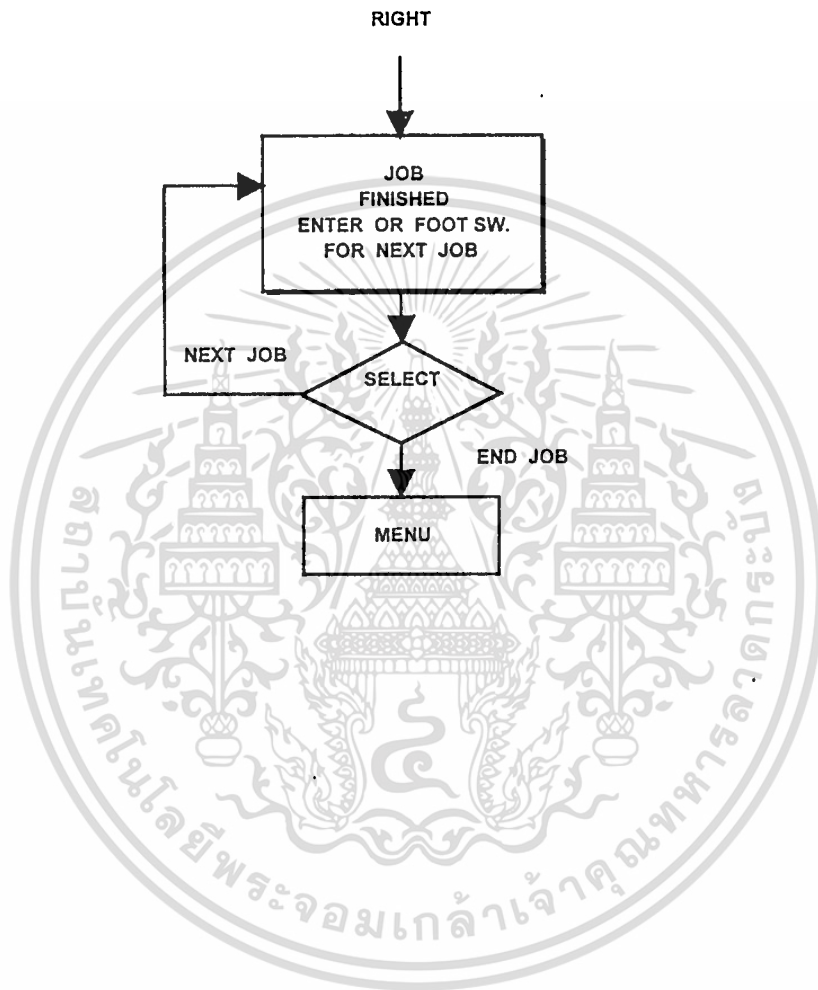
ถ้าต้องการให้เครื่องทำงานตั้งแต่ PROGRAM 00 ถึง 06 เราก็จะป้อนหมายเลข 00 ใน START PROG และป้อนค่า 06 ใน STOP PROG เมื่อกด ENTER เครื่องก็จะทำงานตามขบวนการผลิตจาก PROGRAM 00 จนถึง PROGRAM 06 แล้ว จอ LCD จะแสดงดังรูป

JOB FINISHED ENTER & FOOT SW FOR NEXT JOB

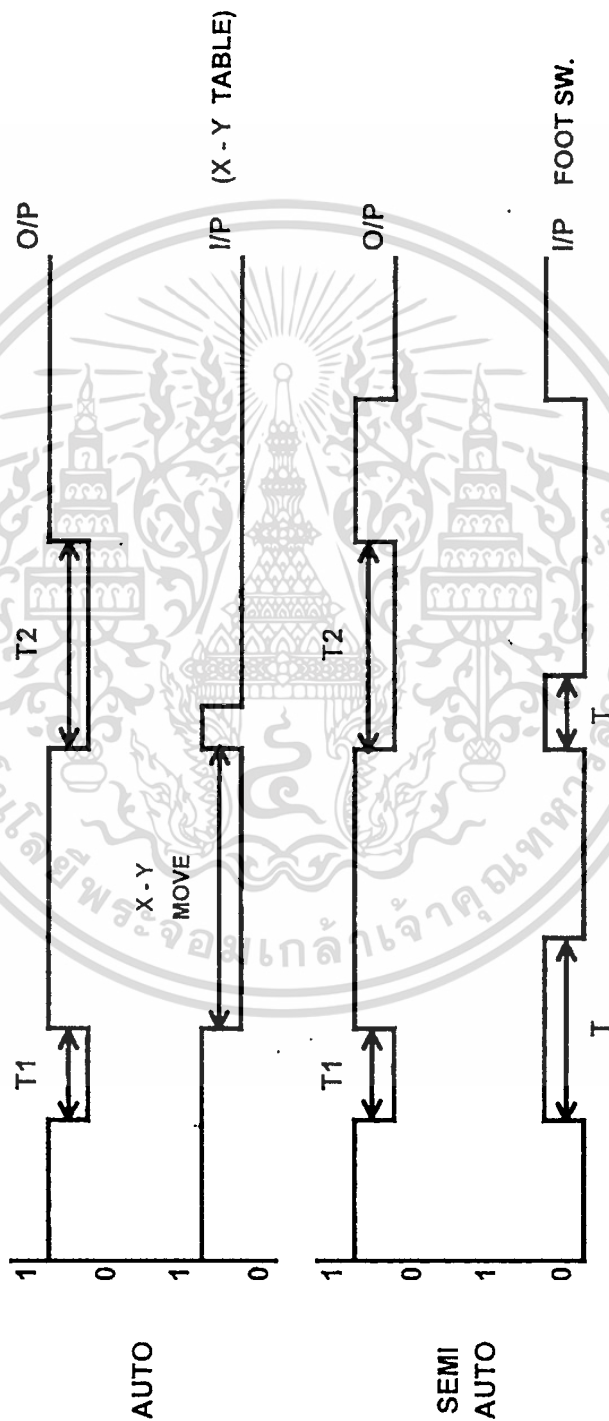
ซึ่งหมายความว่าสิ้นสุดขบวนการผลิต 1 ชิ้น ถ้าต้องการทำงานชิ้นใหม่ ก็ให้กด ENTER สำหรับการทำงาน ใน AUTOMATIC MODE หรือการกด ENTER และ เหยียบ FOOT SW. ใน SEMI - AUTO MODE

FLOW CHART FOR OPERATION





TIMING DIAGRAM



TIME TABLE

PROGRAM #			
STEP	TIME	STEP	TIME
0		50	
1		51	
2		52	
3		53	
4		54	
5		55	
6		56	
7		57	
8		58	
9		59	
10		60	
11		61	
12		62	
13		63	
14		64	
15		65	
16		66	
17		67	
18		68	
19		69	
20		70	
21		71	
22		72	
23		73	
24		74	
25		75	
26		76	
27		77	
28		78	
29		79	
30		80	
31		81	
32		82	
33		83	
34		84	
35		85	
36		86	
37		87	
38		88	
39		89	
40		90	
41		91	
42		92	
43		93	
44		94	
45		95	
46		96	
47		97	
48		98	
49		99	

บทที่ 4

การทดลองการทำงานของเครื่อง

PROGRAMMABLE LIQUID DISPENSER

ขั้นตอนการทดลอง

1. ป้อนค่า TIME ลงใน โปรแกรมต่าง ๆ ดังนี้

PROGRAM	TIME (10 STEP)
00	30, 40, 50, ... , 120
01	30, 50, 70, ... , 190
02	500, 500, ... , 500
03	100, 200, 300, ... , 900
04	100, 100, ... , 100
05	900, 900, ... , 900

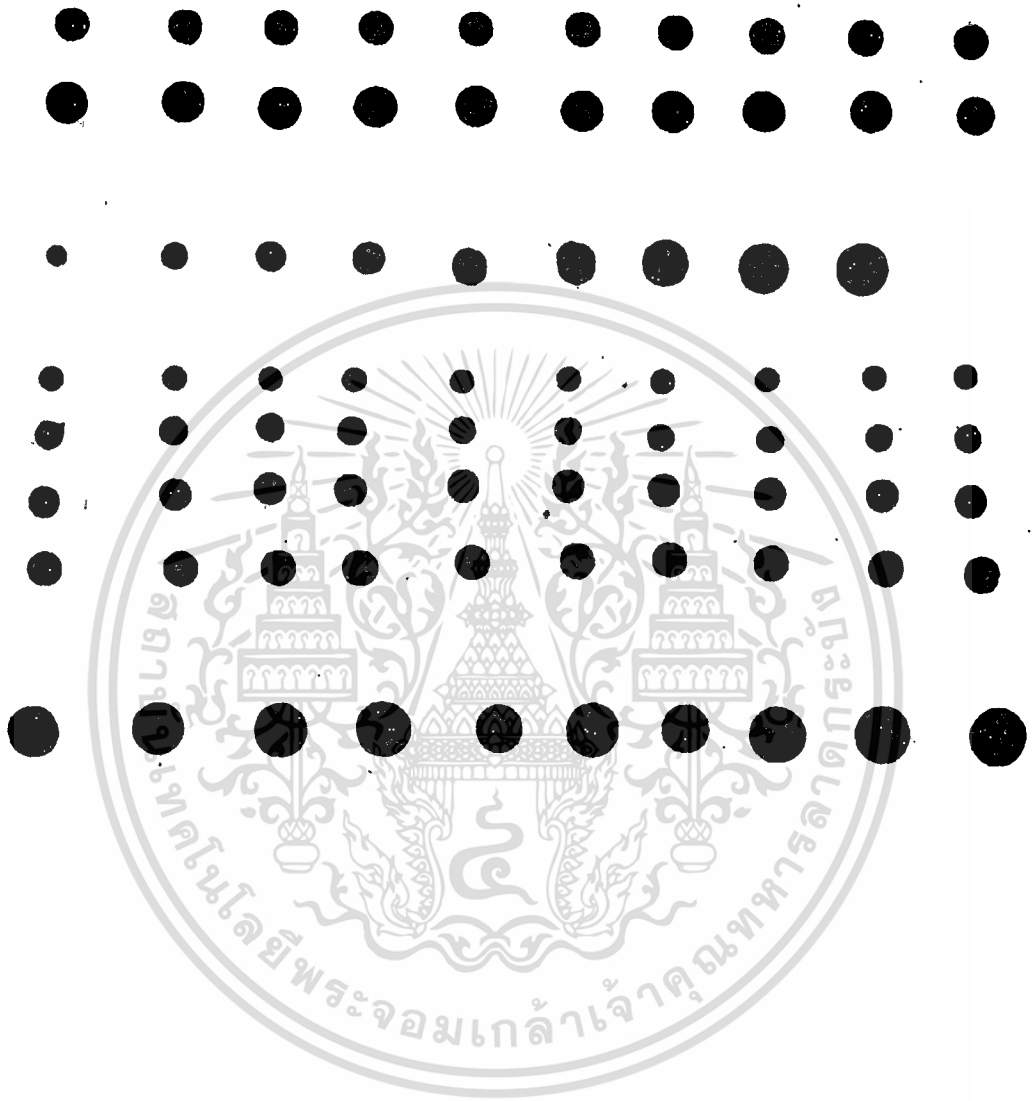
2 ทำการหกดของเหลวตามตารางข้างล่างนี้

ITEM	PROGRAM	PRESSURE	REMARK
1	00	2	รูปที่ 1
2	00	3	รูปที่ 1
3	00	4	รูปที่ 1
4	01	1	รูปที่ 1
5	01	2	รูปที่ 1
6	01	3	รูปที่ 1
7	01	4	รูปที่ 1
8	02	1	รูปที่ 2
9	02	2	รูปที่ 2
10	02	3	รูปที่ 2
11	03	1	รูปที่ 2
12	04	1	รูปที่ 2
13	04	2	รูปที่ 2
14	04	3	รูปที่ 2
15	04	4	รูปที่ 2
16	05	1	รูปที่ 2

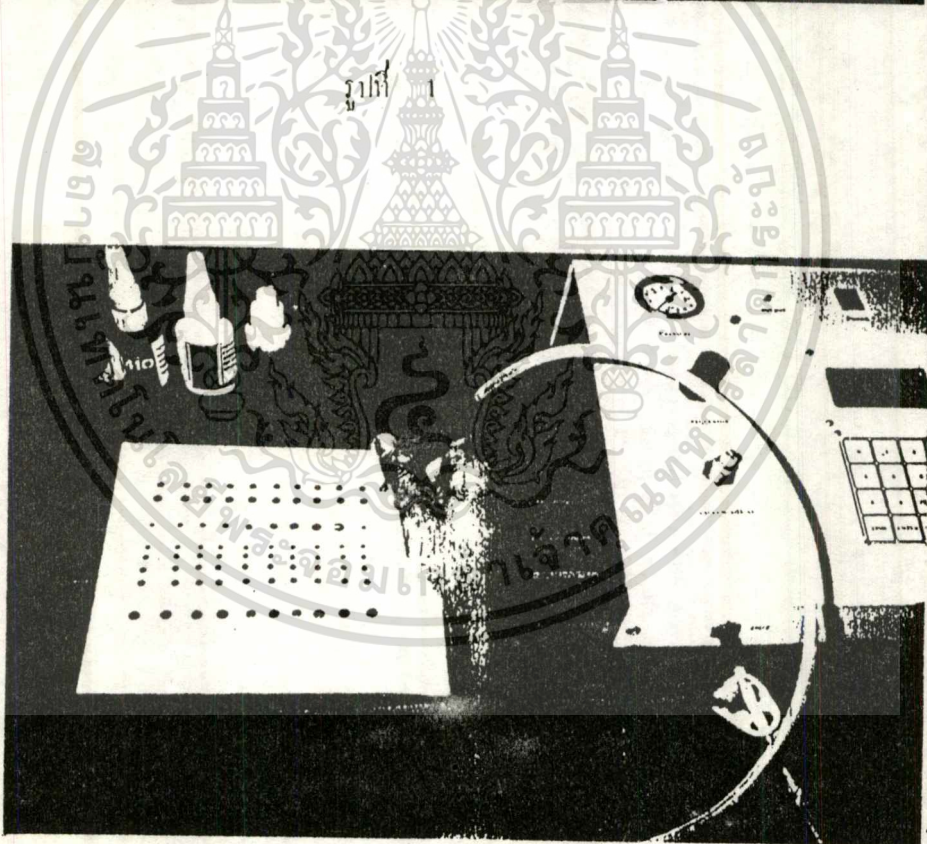
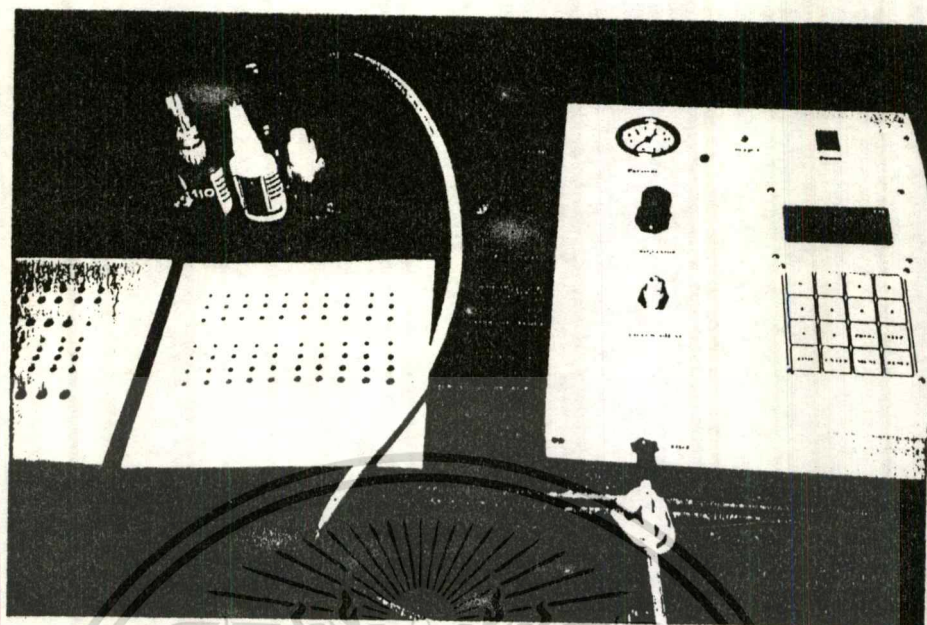


- รูปที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2



รูปที่ 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สรุปผลการทดลอง

จากการทดลองหยอดของเหลวพบว่า การหยอดของเหลวให้ได้ขนาดตามที่ต้องการนั้น ขึ้นอยู่กับองค์ประกอบหลายประการเช่น PRESSURE, TIME, GLUETYPE โดยที่เราสามารถปรับองค์ประกอบต่าง ๆ ข้างต้น ที่สัมพันธ์กันเพื่อให้ได้ขนาดของของเหลวตามที่เราต้องการ

ดังนั้นก่อนที่จะนำไปใช้งานจริงจะต้องทำการทดสอบการหยอดโดยปรับองค์ประกอบต่าง ๆ คือ ความดัน, ค่าเวลา และชนิดของกาว เพื่อให้ได้ขนาดของจุดต่าง ๆ ที่เราต้องการ และตั้งไว้เป็นมาตรฐาน



บทสรุป

เครื่อง PROGRAMABLE LIQUID DISPENSER นี้สามารถทำงานได้อย่างมีประสิทธิภาพเทียบเท่ากับเครื่องขนาดใหญ่แต่มีราคาถูกกว่ามากโดยที่สามารถนำไปใช้ในกระบวนการผลิตจริงได้อย่างสะดวก นอกจากนี้ยังเพิ่มขีดความสามารถของเครื่องโดยต่อเข้ากับ XY Table ทำให้การทำงานสะดวกและรวดเร็วยิ่งขึ้นแต่เนื่องจากโครงการนี้ไม่ได้ทำ X-Y Table จึงใช้คอมพิวเตอร์เป็นตัวจำลองการทำงานของ X-Y Table แทน





เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

;*****
;**      PROJECT APPLIED MCS 51      ***
;**      PROGRAMABLE DISPENSER      ***
;*****
0000 CPU      "8051.TBL"
0000 HOF      "INT8"
;*****
;      MCS-51 INTERNAL REGISTERS
;*****

00F0 =      B:      EQU      0F0H      ;B REGISTER
00E0 =      ACC:    EQU      0E0H      ;ACCUMULATOR
00E1 =      ACC1:   EQU      0E1H      ;ACCUMULATOR BIT 1
00E2 =      ACC2:   EQU      0E2H      ;ACCUMULATOR BIT 2
00D0 =      PSW:    EQU      0D0H      ;PROGRAM STATUS WORD
00B8 =      IPC:    EQU      0B8H      ;INTERRUPT PRIORITY
00B0 =      P3:     EQU      0B0H      ;PORT 3
00A8 =      IEC:    EQU      0A8H      ;INTERRUPT ENABLE
00A0 =      P2:     EQU      0A0H      ;PORT 2
0099 =      SBUF:   EQU      99H      ;SEND BUFFER
0098 =      SCON:   EQU      98H      ;SERIAL CONTROL
0090 =      P1:     EQU      90H      ;PORT 1
008D =      TH1:    EQU      8DH      ;TIMER 1 HIGH
008C =      TH0:    EQU      8CH      ;TIMER 0 HIGH
008B =      TL1:    EQU      8BH      ;TIMER 1 LOW
008A =      TL0:    EQU      8AH      ;TIMER 0 LOW
0089 =      TMOD:   EQU      89H      ;TIMER MODE
0088 =      TCON:   EQU      88H      ;TIMER CONTROL
0087 =      PCON:   EQU      87H      ;POWER CONTROL REGISTER
0083 =      DPH:    EQU      83H      ;DATA POINTER HIGH
0082 =      DPL:    EQU      82H      ;DATA POINTER LOW
0081 =      SP:     EQU      81H      ;STACK POINTER
0080 =      P0:     EQU      80H      ;PORT 0
;*****
;      MCS-51 INTERNAL BIT ADDRESSES
;*****

00D7 =      CY:     EQU      0D7H      ;CARRY FLAG
00D6 =      AC:     EQU      0D6H      ;AUXILIARY-CARRY FLAG
00D5 =      F0:     EQU      0D5H      ;USER FLAG 0
00D4 =      RS1:    EQU      0D4H      ;REGISTER SELECT MSB
00D3 =      RS0:    EQU      0D3H      ;REGISTER SELECT LSB
00D2 =      OV:     EQU      0D2H      ;OVERFLOW FLAG
00D0 =      P:      EQU      0D0H      ;PARITY FLAG
00BC =      PS:     EQU      0BCH      ;PRIORITY SERIAL PORT
00BB =      PT1:    EQU      0BBH      ;PRIORITY TIMER 1
00BA =      PX1:    EQU      0BAH      ;PRIORITY EXTERNAL 1
00B9 =      PT0:    EQU      0B9H      ;PRIORITY TIMER 0
00B8 =      PX0:    EQU      0B8H      ;PRIORITY EXTERNAL 0
00AF =      EA:     EQU      0AFH      ;ENABLE ALL INTERRUPT
00AC =      ES:     EQU      0ACH      ;ENABLE SERIAL INTERRUPT
00AB =      ET1:    EQU      0ABH      ;ENABLE TIMER 1 INTERRUPT
00AA =      EX1:    EQU      0AAH      ;ENABLE EXTERNAL 1 INTERRUPT
00A9 =      ET0:    EQU      0A9H      ;ENABLE TIMER 0 INTERRUPT
00A8 =      EX0:    EQU      0A8H      ;ENABLE EXTERNAL 0 INTERRUPT

```

```

009F =      SM0:   EQU      09FH      ; SERIAL MODE 0
009E =      SM1:   EQU      09EH      ; SERIAL MODE 1
009D =      SM2:   EQU      09DH      ; SERIAL MODE 2
009C =      REN:   EQU      09CH      ; SERIAL RECEPTION ENABLE
009B =      TB8:   EQU      09BH      ; TRANSMITT BIT 8
009A =      RB8:   EQU      09AH      ; RECEIVE BIT 8
0099 =      TI:    EQU      099H      ; TRANSMIT INTERRUPT FLAG
0098 =      RI:    EQU      098H      ; RECEIVE INTERRUPT FLAG
008F =      TF1:   EQU      08FH      ; TIMER 1 OVERFLOW FLAG
008E =      TR1:   EQU      08EH      ; TIMER 1 RUN CONTROL BIT
008D =      TF0:   EQU      08DH      ; TIMER 0 OVERFLOW FLAG
008C =      TR0:   EQU      08CH      ; TIMER 0 RUN CONTROL BIT
008B =      IE1:   EQU      08BH      ; EXT INTERRU. 1 EDGE FLAG
008A =      IT1:   EQU      08AH      ; EXT INTERRU. 1 TYPE FLAG
0089 =      IE0:   EQU      089H      ; EXT INTERRU. 0 EDGE FLAG
0088 =      IT0:   EQU      088H      ; EXT INTERRU. 0 TYPE FLAG

0097 =      TX:    EQU      097H      ; P1 BIT 7
0096 =      RX:    EQU      096H      ; P1 BIT 6
0095 =      SEMI:  EQU      095H      ; P1 BIT 5 FOR SEMI RUNING
; *****
;
;                               DEFINE I/O PORT
; *****

E0E0 =      PORTA: EQU      0E0E0H    ; PORT A LCD DATA
E0E1 =      PORTB: EQU      0E0E1H    ; PORT B LCD CONTROL PORT
E0E2 =      PORTC: EQU      0E0E2H    ; PORT C EXPAND
E0E3 =      PORTP: EQU      0E0E3H    ; 8255 CONTROL PORT
; *****
;
;                               INTERNAL MEMORY
; *****

0070 =      KEYBUFFER: EQU 070H      ; KEEP KEY PRESS VALUE
006F =      MODE:   EQU      06FH      ; KEEP MODE STATUS
006E =      COUNTKEY: EQU 06EH      ; COUNT KEY ENTER
; *****
;
;                               DEFINE BEGIN DATA SEGMENT AT 2200H-3FFFH
; *****

2200 =      DATASEGMENT: EQU 2200H    ; ADDRESS BEGIN SEGMENT
006C =      POINTERHIGH: EQU 06CH      ; HIGH BYTE FOR POINTER
006D =      POINTERLOW: EQU 06DH      ; LOW BYTE FOR POINTER
007F =      DISPLAYTIME0: EQU 07FH      ; ASCII TIME DIGIT0
007E =      DISPLAYTIME1: EQU 07EH      ; ASCII TIME DIGIT1
007D =      DISPLAYTIME2: EQU 07DH      ; ASCII TIME DIGIT2
007C =      DISPLAYSTEP0: EQU 07CH      ; ASCII STEP DIGIT0
007B =      DISPLAYSTEP1: EQU 07BH      ; ASCII STEP DIGIT1
007A =      DISPLAYPROG0: EQU 07AH      ; ASCII PROGRAM DIGIT0
0079 =      DISPLAYPROG1: EQU 079H      ; ASCII PROGRAM DIGIT1
0066 =      DISPLAYSTART0: EQU 066H      ;
0065 =      DISPLAYSTART1: EQU 065H      ;
0064 =      DISPLAYSTOP0: EQU 064H      ;
0063 =      DISPLAYSTOP1: EQU 063H      ;
0078 =      ASCIISELECT: EQU 078H      ; ASCII SELECT MENU

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

0074 = BCDPROG:      EQU    074H    ; BCD OF CURRENT TIME
0075 = BCDSTEP:      EQU    075H    ; BCD OF CURRENT STEP
0076 = BCDTIME:      EQU    076H    ; BCD OF CURRENT TIME
0060 = BCDSTART:     EQU    060H    ; START PROGRAM
005F = BCDSTOP:      EQU    05FH    ; STOP PROGRAM
0073 = BINARYTIME:   EQU    073H    ; BINARY FOR TIME
0072 = BINARYSTEP:   EQU    072H    ; BINARY FOR STEP
0071 = BINARYPROG:   EQU    071H    ; BINARY FOR PROGRAM
006B = BINARYSTART:  EQU    06BH    ; START PROGRAM
006A = BINARYSTOP:   EQU    06AH    ; STOP PROGRAM

005E = BINARYCOUNTDOT0:EQU    05EH
005D = BINARYCOUNTDOT1:EQU    05DH
005C = BCDCOUNTDOT0: EQU    05CH
005B = BCDCOUNTDOT1: EQU    05BH
005A = BCDCOUNTDOT2: EQU    05AH

0059 = DISPLAYDOT4:  EQU    059H
0058 = DISPLAYDOT3:  EQU    058H
0057 = DISPLAYDOT2:  EQU    057H
0056 = DISPLAYDOT1:  EQU    056H
0055 = DISPLAYDOT0:  EQU    055H

0054 = COUNTINTTIME0: EQU    054h
0053 = COUNTINTTIME1: EQU    053h
0052 = TIME VALUE    EQU    052H ;KEEP TIME RANGGE
0051 = TIMECOUNT:   EQU    051H ;KEEP TIME COUNT TIMER
0050 = MSL:          EQU    050H
004F = MSH:          EQU    04FH
0064 = DELAY100MS:   EQU    100
0069 = COUNTPROGDIGIT: EQU    069H
0068 = COUNTSTEPDIGIT: EQU    068H
0067 = COUNTTIMEDIGIT: EQU    067H
006E = COUNTSELECTDIGIT: EQU    06EH
0062 = COUNTSTARTDIGIT: EQU    062H
0061 = COUNTSTOPDIGIT: EQU    061H

;*****
;          I N T E R N A L   F L A G
;*****

007F = MENUFLAG:      EQU    07FH    ; IN MENU MODE
007E = FILLMODEFLAG:  EQU    07EH    ; FILL MODE FLAG
007D = EDITMODEFLAG:  EQU    07DH    ; EDIT MODE FLAG
007C = RUNMODEFLAG:   EQU    07CH    ; RUN MODE FLAG
0076 = INFOFLAG:      EQU    076H    ; INFOMATION FLAG
007B = INPROGFIELD:   EQU    07BH    ; IN PROGRAM FIELD
007A = INSTEPFIELD:   EQU    07AH    ; IN STEP FIELD
0079 = INTIMEFIELD:   EQU    079H    ; IN TIME FIELD

0078 = INSTARTFIELD:  EQU    078H    ; FLAG IN START FIELD
0077 = INSTOPFIELD:   EQU    077H    ; FLAG IN STOP FIELD
;*****

0000      ORG      0000H
0000 020030  LJMP    START

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

0003          ORG      0003H      ;FOR INTO KEYBOARD
0003 0202D9    LJMP      ISRKB

;***** MAIN PROGRAM *****
0030          ORG      0030H
;*****
;          INITAIL I/O 8255 PORT
;*****
0030 758130    START: MOV      SP,#30H      ;SET STACK POINTER
0033 7E03      MOV      R6,#3          ;WAIT FOR 8255
0035 1207D2    LCALL     DELAY
0038 7910      MOV      R1,#16
003A 90E0E3    POINT1:MOV     DPTR,#PORTP ;SET 8255 PORT
003D 7480      MOV      A,#80H
003F F0        MOVX     @DPTR,A        ;FIRST DELAY
0040 D9F8      DJNZ     R1,POINT1
;*****
;          INTERRUPT HANDLER INT 0 FOR KEYBOARD
;*****
0042 75A881    MOV      IEC,#81H      ;ENABLE INTERRUPT INTO,ITO
0045 75880F    MOV      TCON,#0FH     ;ACTIVE AT POSITIVE EGDE

0048 1207DD    LCALL     INITLCD
;*****
;          CLEAR GROBAL VARIABLE AND SYSTEM FLAG
;          I N I T I A L      S Y S T E M      F L A G
;*****
004B C27F      CLR      MENUFLAG      ;MENU FLAG
004D C27E      CLR      FILLMODEFLAG   ;FILL MODE FLAG
004F C27D      CLR      EDITMODEFLAG   ;EDIT MODE FLAG
0051 C27C      CLR      RUNMODEFLAG    ;RUN MODE FLAG
0053 C276      CLR      INFOFLAG       ;INFOMATION FLAG

0055 C27B      CLR      INPROGFIELD    ;PROGRAM FIELD
0057 C27A      CLR      INSTEPFIELD    ;STEP FIELD
0059 C279      CLR      INTIMEFIELD    ;TIME FIELD
005B C278      CLR      INSTARTFIELD   ;START FIELD
005D C277      CLR      INSTOPFIELD    ;STOP FILED
;*****
;          INITAIL DEFAULT VALUE FOR SYSTEM VARIABLE
;          INITIAL SYSTEM VARIABLE
;*****

005F 7430      MOV      A,#30H        ;ASCII 0

;*****
;          ASCII FOR TIME
;*****
0061 F57F      MOV      DISPLAYTIME0,A ; DIGIT2 DIGIT1 DIGIT0
0063 F57E      MOV      DISPLAYTIME1,A ; 7D 7E 7F
0065 F57D      MOV      DISPLAYTIME2,A
;*****
;          ASCII FOR STEP
;*****

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

0067 F57C      MOV      DISPLAYSTEP0,A      ; DIGIT1 DIGIT0
0069 F57B      MOV      DISPLAYSTEP1,A      ; 7B      7C
;*****
;ASCII FOR PROGRAM
;*****
006B F57A      MOV      DISPLAYPROG0,A      ;      DIGIT1      DIGIT0
006D F579      MOV      DISPLAYPROG1,A      ;      79      7A
;*****
; ASCII FOR START & STOP
;*****

006F F566      MOV      DISPLAYSTART0,A
0071 F565      MOV      DISPLAYSTART1,A
0073 F564      MOV      DISPLAYSTOP0,A
0075 F563      MOV      DISPLAYSTOP1,A
;*****
;INITIAL VARIABLE
;*****
0077 7400      MOV      A,#0
0079 F573      MOV      BINARYTIME,A
007B F572      MOV      BINARYSTEP,A
007D F571      MOV      BINARYPROG,A
007F F56B      MOV      BINARYSTART,A
0081 F56A      MOV      BINARYSTOP,A
0083 F576      MOV      BCDTIME,A
0085 F575      MOV      BCDSTEP,A
0087 F574      MOV      BCDPROG,A
0089 F560      MOV      BCDSTART,A
008B F55F      MOV      BCDSTOP,A
008D F56C      MOV      POINTERHIGH,A
008F F56D      MOV      POINTERLOW,A
0091 F578      MOV      ASCIISELECT,A
0093 F569      MOV      COUNTPROGDIGIT,A
0095 F568      MOV      COUNTSTEPDIGIT,A
0097 F567      MOV      COUNTTIMEDIGIT,A
0099 F56E      MOV      COUNTSELECTDIGIT,A
009B F562      MOV      COUNTSTARTDIGIT,A
009D F561      MOV      COUNTSTOPDIGIT,A
009F 755E00    MOV      BINARYCOUNTDOT0,#0
00A2 755D00    MOV      BINARYCOUNTDOT1,#0
00A5 755400    MOV      COUNTINTTIME0,#0
00A8 755300    MOV      COUNTINTTIME1,#0
00AB 755200    MOV      TIMEVALUE,#0
00AE 755100    MOV      TIMECOUNT,#0
00B1 C28C      CLR      TRO
;*****
;      BEGIN SYSTEM BY DISPLAY MENU
;      INITIAL VALUE FOR START MENU
;*****

00B3 900A23    MOV      DPTR,#NAME
00B6 120826    LCALL     LCDLD
00B9 7E08      MOV      R6,#8
00BB 1207D2    LCALL     DELAY

```

```

00BE D297          SETB      TX          ; OUT TO XY TABLE
                                ; CHECK FOR MODE SW
00C0 309517        JNB       SEMI,SS1
00C3 D296          SETB      RX

00C5 C083          PUSH      DPH
00C7 C082          PUSH      DPL
00C9 900DE3        MOV       DPTR,#AUTOMODE
00CC 120826        LCALL     LCDLD
00CF D082          POP       DPL
00D1 D083          POP       DPH

00D3 7E05          MOV       R6,#5
00D5 1207D2        LCALL     DELAY
00D8 8015          SJMP      CC
00DA C296          CLR       RX          S1:

00DC C083          PUSH      DPH
00DE C082          PUSH      DPL
00E0 900DA3        MOV       DPTR,#SEMIMODE
00E3 120826        LCALL     LCDLD
00E6 D082          POP       DPL
00E8 D083          POP       DPH

00EA 7E0D          MOV       R6,#13
00EC 1207D2        LCALL     DELAY

00EF D27F          CC:       SETB      MENUFLAG
00F1 12071C        LCALL     MODEMENU
00F4 1206E8        LCALL     GOTOMENU
00F7 80FE          SJMP      $

; *****
;   S U B R O U T I N E   A R E A
; *****

; *****
;   INPUT ==> BCDPROG <== VALUE
; *****

00F9 E574          CALPROGVALUE: MOV      A,BCDPROG
00FB FF            MOV       R7,A
00FC 1207A4        LCALL     BCDTOBINARY
00FF EE            MOV       A,R6
0100 F571          MOV       BINARYPROG,A

0102 E574          MOV       A,BCDPROG
0104 540F          ANL       A,#0FH
0106 4430          ORL       A,#30H
0108 F57A          MOV       DISPLAYPROG0,A
010A E574          MOV       A,BCDPROG

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

010C 54F0      ANL      A,#0F0H
010E 03        RR      A
010F 03        RR      A
0110 03        RR      A
0111 03        RR      A
0112 540F      ANL      A,#0FH
0114 4430      ORL      A,#30H
0116 F579      MOV      DISPLAYPROG1,A

```

```

0118 22        RET

```

```

;*****

```

```

0119 E575      CALSTEPVALUE:  MOV      A,BCDSTEP
011B FF        MOV      R7,A
011C 1207A4    LCALL     BCDTOBINARY
011F EE        MOV      A,R6
0120 F572      MOV      BINARYSTEP,A
0122 E575      MOV      A,BCDSTEP
0124 540F      ANL      A,#0FH
0126 4430      ORL      A,#30H
0128 F57C      MOV      DISPLAYSTEP0,A
012A E575      MOV      A,BCDSTEP
012C 54F0      ANL      A,#0F0H
012E 03        RR      A
012F 03        RR      A
0130 03        RR      A
0131 03        RR      A
0132 540F      ANL      A,#0FH
0134 4430      ORL      A,#30H
0136 F57B      MOV      DISPLAYSTEP1,A

```

```

0138 22        RET

```

```

;*****

```

```

0139 E576      CALTIMEVALUE:  MOV      A,BCDTIME
013B FF        MOV      R7,A
013C 1207A4    LCALL     BCDTOBINARY
013F EE        MOV      A,R6
0140 F573      MOV      BINARYTIME,A
0142 E576      MOV      A,BCDTIME
0144 540F      ANL      A,#0FH
0146 4430      ORL      A,#30H
0148 F57F      MOV      DISPLAYTIME0,A
014A E576      MOV      A,BCDTIME
014C 54F0      ANL      A,#0F0H
014E 03        RR      A
014F 03        RR      A
0150 03        RR      A
0151 03        RR      A
0152 540F      ANL      A,#0FH
0154 4430      ORL      A,#30H
0156 F57E      MOV      DISPLAYTIME1,A

```

```

0158 22        RET

```

```

;*****
;*****

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

0159 E560          CALSTARTVALUE: MOV      A,BCDSTART
015B FF            MOV      R7,A
015C 1207A4        LCALL     BCDTOBINARY
015F EE            MOV      A,R6
0160 F56B          MOV      BINARYSTART,A
0162 E560          MOV      A,BCDSTART
0164 540F          ANL      A,#0FH
0166 4430          ORL      A,#30H
0168 F566          MOV      DISPLAYSTART0,A
016A E560          MOV      A,BCDSTART
016C 54F0          ANL      A,#0F0H
016E 03            RR      A
016F 03            RR      A
0170 03            RR      A
0171 03            RR      A
0172 540F          ANL      A,#0FH
0174 4430          ORL      A,#30H
0176 F565          MOV      DISPLAYSTART1,A

0178 22            RET
;*****
;*****
0179 E55F          CALSTOPVALUE:  MOV      A,BCDSTOP
017B FF            MOV      R7,A
017C 1207A4        LCALL     BCDTOBINARY
017F EE            MOV      A,R6
0180 F56A          MOV      BINARYSTOP,A
0182 E55F          MOV      A,BCDSTOP
0184 540F          ANL      A,#0FH
0186 4430          ORL      A,#30H
0188 F564          MOV      DISPLAYSTOP0,A
018A E55F          MOV      A,BCDSTOP
018C 54F0          ANL      A,#0F0H
018E 03            RR      A
018F 03            RR      A
0190 03            RR      A
0191 03            RR      A
0192 540F          ANL      A,#0FH
0194 4430          ORL      A,#30H
0196 F563          MOV      DISPLAYSTOP1,A

0198 22            RET
;*****
;*****
0199 E579 DISPPROGFIELD:MOV A,DISPLAYPROG1 ; DISPLAY UPDATE
019B FF            MOV R7,A ;ASCII IN R7 DISPLAY
019C 74CC          MOV A,#0CCH;LOCATION ADDRESS AT LCD
019E 12080E        LCALL     WRIBYTE
01A1 E57A          MOV A,DISPLAYPROG0
01A3 FF            MOV R7,A ;ASCII IN R7 FOR DISPLAY
01A4 74CD          MOV A,#0CDH;LOCATION ADDRESS AT LCD
01A6 12080E        LCALL     WRIBYTE

01A9 22            RET
;*****
;*****

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

01AA E57B DISPSTEPFIELD:MOV A,DISPLAYSTEP1 ; DISPLAY UPDATE
01AC FF          MOV R7,A    ;ASCII IN R7 FOR DISPLAY
01AD 749C        MOV A,#09CH;LOCATION ADDRESS AT LCD
01AF 12080E      LCALL  WRIBYTE
01B2 E57C        MOV A,DISPLAYSTEP0
01B4 FF          MOV R7,A    ;ASCII IN R7 FOR DISPLAY
01B5 749D        MOV A,#09DH;LOCATION ADDRESS AT LCD
01B7 12080E      LCALL  WRIBYTE

01BA 22          RET
;*****

01BB E57E DISPTIMEFIELD:MOV A,DISPLAYTIME1 ; DISPLAY UPDATE,
01BD FF          MOV R7,A    ;ASCII IN R7 FOR DISPLAY
01BE 74DB        MOV A,#0DBH;LOCATION ADDRESS AT LCD
01C0 12080E      LCALL WRIBYTE
01C3 E57F        MOV A,DISPLAYTIME0
01C5 FF          MOV R7,A    ;ASCII IN R7 FOR DISPLAY
01C6 74DC        MOV A,#0DCH;LOCATION ADDRESS AT LCD
01C8 12080E      LCALL WRIBYTE
01CB 7430        MOV A,#30H
01CD FF          MOV R7,A    ;ASCII IN R7 FOR DISPLAY
01CE 74DD        MOV A,#0DDH;LOCATION ADDRESS AT LCD
01D0 12080E      LCALL WRIBYTE
01D3 22          RET
;*****

01D4 E565        DISPSTARTFIELD: MOV A,DISPLAYSTART1
01D6 FF          MOV R7,A
01D7 74CC        MOV A,#0CCH
01D9 12080E      LCALL WRIBYTE
01DC E566        MOV A,DISPLAYSTART0
01DE FF          MOV R7,A
01DF 74CD        MOV A,#0CDH
01E1 12080E      LCALL WRIBYTE
01E4 22          RET
;*****

01E5 E563        DISPSTOPFIELD:  MOV A,DISPLAYSTOP1
01E7 FF          MOV R7,A
01E8 749C        MOV A,#09CH
01EA 12080E      LCALL WRIBYTE
01ED E564        MOV A,DISPLAYSTOP0
01EF FF          MOV R7,A
01F0 749D        MOV A,#09DH
01F2 12080E      LCALL WRIBYTE
01F5 22          RET
;*****

01F6             DISPDOTFIELD:
; BLANK LEADING ZERO DISPLAY

01F6 E555        MOV A,55H
01F8 B4301F      CJNE A,#30H,DIS00

```

```

01FB 7420                                MOV     A,#20H
01FD F555                                MOV     55H,A
01FF E556                                MOV     A,56H
0201 B43016                             CJNE    A,#30H,DIS00
0204 7420                                MOV     A,#20H
0206 F556                                MOV     56H,A
0208 E557                                MOV     A,57H
020A B4300D                             CJNE    A,#30H,DIS00
020D 7420                                MOV     A,#20H
020F F557                                MOV     57H,A
0211 E558                                MOV     A,58H
0213 B43004                             CJNE    A,#30H,DIS00
0216 7420                                MOV     A,#20H
0218 F558                                MOV     58H,A

021A E555 DIS00: MOV     A,DISPLAYDOT0
021C FF                                MOV     R7,A           ;ASCII IN R7 FOR DISPLAY
021D 74C3                             MOV     A,#0C3H       ;LOCATION ADDRESS AT LCD
021F 12080E                             LCALL   WRIBYTE

0222 E556                                MOV     A,DISPLAYDOT1
0224 FF                                MOV     R7,A           ;ASCII IN R7 FOR DISPLAY
0225 74C4                             MOV     A,#0C4H       ;LOCATION ADDRESS AT LCD
0227 12080E                             LCALL   WRIBYTE

022A E557                                MOV     A,DISPLAYDOT2
022C FF                                MOV     R7,A           ;ASCII IN R7 FOR DISPLAY
022D 74C5                             MOV     A,#0C5H       ;LOCATION ADDRESS AT LCD
022F 12080E                             LCALL   WRIBYTE

0232 E558                                MOV     A,DISPLAYDOT3
0234 FF                                MOV     R7,A           ;ASCII IN R7 FOR DISPLAY
0235 74C6                             MOV     A,#0C6H       ;LOCATION ADDRESS AT LCD
0237 12080E                             LCALL   WRIBYTE

023A E559                                MOV     A,DISPLAYDOT4
023C FF                                MOV     R7,A           ;ASCII IN R7 FOR DISPLAY
023D 74C7                             MOV     A,#0C7H       ;LOCATION ADDRESS AT LCD
023F 12080E                             LCALL   WRIBYTE

0242 22                                RET

;*****
; UPDATE PROGRAM AFTER PRESS ENTER
; INPUT ===> BCDPROG
; CONVERT TO BINARY TO ASCII
;*****

0243 1200F9                             UPDATEPROGFIELD:LCALL  CALPROGVALUE
0246 120199                             LCALL   DISPPROGFIELD
0249 12026C                             LCALL   GETTIMEVALUE
024C 22                                RET

;*****
; UPDATE STEP AFTER PRESS ENTER
; INPUT == > BCDSTEP
;*****

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

024D 120119      UPDATESTEPFIELD:      LCALL  CALSTEPVALUE
0250 1201AA      LCALL  DISPSTEPFIELD
0253 12026C      LCALL  GETTIMEVALUE

0256 22          RET

; *****

0257 120159      UPDATESTARTFIELD:    LCALL  CALSTARTVALUE

025A 1201D4      LCALL  DISPSTARTFIELD

025D 22          RET

; *****

025E 120179      UPDATESTOPFIELD:     LCALL  CALSTOPVALUE
0261 1201E5      LCALL  DISPSTOPFIELD

0264 22          RET

; *****
; PUT TIME VALUE IN MEMORY
; *****
0265 1202A8      PUTTIMEVALUE:  LCALL  FINDADDRESS
0268 120294      LCALL  WRITEADDRESS
026B 22          RET

; *****
; READ TIME FROM MEMORY
; *****

026C 1202A8      GETTIMEVALUE:  LCALL  FINDADDRESS
026F 12029E      LCALL  READADDRESS
0272 E573        MOV          A,BINARYTIME
0274 FB          MOV          R3,A
0275 12088F      LCALL  BINARYTOBCD
0278 8C76        MOV          BCDTIME,R4

027A E576        MOV          A,BCDTIME
027C 540F        ANL          A,#0FH
027E 4430        ORL          A,#30H
0280 F57F        MOV          DISPLAYTIME0,A
0282 E576        MOV          A,BCDTIME
0284 54F0        ANL          A,#0F0H
0286 03          RR          A
0287 03          RR          A
0288 03          RR          A
0289 03          RR          A
028A 540F        ANL          A,#0FH
028C 4430        ORL          A,#30H
028E F57E        MOV          DISPLAYTIME1,A
0290 1201BB      LCALL  DISPTIMEFIELD
0293 22          RET

```

```

;*****
; WRITE VALUE TO ADDRESS AT CURRENT POINTER
; WRITE VALUE => A
; 73H BINARY TIME VALUE
;*****

```

```

0294 E573
0296 856C83
0299 856D82
029C F0
029D 22

```

```

WRITEADDRESS:  MOV    A,BINARYTIME
                MOV    DPH,POINTERHIGH
                MOV    DPL,POINTERLOW
                MOVX   @DPTR,A
                RET

```

```

;*****
; READE VALUE TO ADDRESS AT CURRENT POINTER
; READ VALUE => A
; ALWAYS READ TO 73H
;*****

```

```

029E 856C83
02A1 856D82
02A4 E0
02A5 F573
02A7 22

```

```

READADDRESS:   MOV    DPH,POINTERHIGH
                MOV    DPL,POINTERLOW
                MOVX   A,@DPTR
                MOV    BINARYTIME,A
                RET

```

```

;*****
; FIND ADDRESS
; ABSOLUTE ADDRESS = PROGRAM + STEP
; INPUT  <= BINPROG,BINSTEP
; OUTPUT <= POINTERHIGH,POINTERLOW
; USE
;*****

```

```

02A8 902200 FINDADDRESS:MOV DPTR,#DASEGMENT;BEGIN DATA SEG
02AB AE71                MOV    R6,BINARYPROG
02AD BE0003                CJNE   R6,#0,LOOKUP1
02B0 0202BF                LJMP   DD1
02B3 902200                LOOKUP1: MOV    DPTR,#DASEGMENT
02B6 AE71                MOV    R6,BINARYPROG
02B8 7964                LOOKUP2: MOV    R1,#100
02BA A3                LOOKUP3:  INC    DPTR
02BB D9FD                DJNZ    R1,LOOKUP3
02BD DEF9                DJNZ    R6,LOOKUP2
02BF 85836C                DD1:    MOV    POINTERHIGH,DPH
02C2 85826D                MOV    POINTERLOW,DPL
02C5 A872                ADDOFFSET: MOV R0,BINARYSTEP ; ADD OFFSET STEP
02C7 B80003                CJNE   R0,#0,LOOKUP5
02CA 0202D2                LJMP   DD2
02CD A872                LOOKUP5: MOV    R0,BINARYSTEP
02CF A3                LOOKUP6:  INC    DPTR
02D0 D8FD                DJNZ    R0,LOOKUP6
02D2 85836C                DD2:    MOV    POINTERHIGH,DPH
02D5 85826D                MOV    POINTERLOW,DPL
02D8 22                FINISHED: RET

```

```

;*****

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;*****
; INTERRUPT SERVICE ROUTINE FOR KEYBOARD INTO
;           I       S       R
;*****
```

```
02D9 C0E0      ISRKB: PUSH      ACC
02DB C0D0              PUSH      PSW
02DD E590              MOV       A,P1 ;GET DATA FROM PORT P1
02DF 540F              ANL       A,#0FH ;GET KEY VALUE IN A
02E1 F570              MOV       KEYBUFFER,A ;KEEP IN MEMORY
02E3 E570              MOV       A,KEYBUFFER
```

```
;***** CHECK FOR KEY A *****
;           SELECT P R O G R A M FIELD KEY
;*****
```

```
02E5 B40A1F      CJNE      A,#0AH,GONEXT1
02E8 207F19      JB      MENUFLAG,GOOUT0 ;IF MODE MENU GO OUT
02EB 207E08      JB      FILLMODEFLAG,DOEDIT0;IF PROG FIELD GO OUT
02EE 207D05              JB      EDITMODEFLAG,DOEDIT0
02F1 207C10              JB      RUNMODEFLAG,GOOUT0
02F4 800E              SJMP      GOOUT0
02F6 207B0B      DOEDIT0: JB      INPROGFIELD,GOOUT0
02F9 207A05              JB      INSTEPFIELD,DOIT0
02FC 207902              JB      INTIMEFIELD,DOIT0
02FF 8003              SJMP      GOOUT0
0301 120666      DOIT0:  LCALL     GOTOPROG;MOVE CURSOR TO PROGRAM
0304 020661      GOOUT0: LJMP      GOHOME
```

```
;***** CHECK FOR KEY B *****
;           SELECT S T E P FIELD KEY
;*****
```

```
0307 B40B1F      GONEXT1:  CJNE      A,#0BH,GONEXT2
030A 207F19              JB      MENUFLAG,GOOUT1 ;IF MENU GO OUT
030D 207E08              JB      FILLMODEFLAG,DOEDIT1
0310 207D05      JB      EDITMODEFLAG,DOEDIT1;IF STEP FIELD GO OUT
0313 207C10              JB      RUNMODEFLAG,GOOUT1
0316 800E              SJMP      GOOUT1
0318 207B08      DOEDIT1:  JB      INPROGFIELD,DOIT1
031B 207A08              JB      INSTEPFIELD,GOOUT1
031E 207902              JB      INTIMEFIELD,DOIT1
0321 8003              SJMP      GOOUT1
0323 120680      DOIT1:  LCALL     GOTOSTEP
0326 020661      GOOUT1:  LJMP      GOHOME
```

```
;***** CHECK FOR KEY C *****
;           SELECT T I M E FIELD KEY
;*****
```

```
0329 B40C1F      GONEXT2:  CJNE      A,#0CH,GONEXT3
032C 207F19              JB      MENUFLAG,GOOUT2 ;IF MENU GO OUT
032F 207E08              JB      FILLMODEFLAG,DOEDIT2
0332 207D05      JB      EDITMODEFLAG,DOEDIT2 ;IF TIME GO OUT
0335 207C10              JB      RUNMODEFLAG,GOOUT2
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

0338 800E          SJMP      GOOUT2
033A 207B08        DOEDIT2:  JB      INPROGFIELD,DOIT2
033D 207A05        JB      INSTEPFIELD,DOIT2
0340 207905        JB      INTIMEFIELD,GOOUT2
0343 8003          SJMP      GOOUT2
0345 12069A        DOIT2:    LCALL   GOTOTIME
0348 020661        GOOUT2:   LJMP    GOHOME

```

```

;***** CHECK FOR KEY D *****
;                               E N T E R       KEY
;*****

```

```

034B B40D11        GONEXT3:   CJNE    A,#0DH,S2
034E 207F0C        JB      MENUFLAG,S1
0351 207E12        JB      FILLMODEFLAG,FILLDOIT
0354 207D6F        JB      EDITMODEFLAG,EDITDOIT
0357 207C08        JB      RUNMODEFLAG,RUNDOITXX
035A 207607        JB      INFOFLAG,INFODOITXX
035D 8146          S1:      AJMP     GOOUT3
035F 020449        S2:      LJMP     GONEXT4
0362 8107          RUNDOITXX: AJMP     RUNDOIT
0364 8140          INFODOITXX: AJMP     INFODOIT
0366 207B0B        FILLDOIT:  JB      INPROGFIELD,FILLTOSTEP
                                JB      INSTEPFIELD,FILLTOTIME
                                JB      INTIMEFIELD,FILLTOTIME
0369 207916        LCALL   MODEFILL
036C 120735        LCALL   GOTOPROG
036F 120666        AJMP     GOOUT3
0372 8146          FILLTOSTEP: LCALL   UPDATEPROGFIELD
0374 120243        MOV      BCDSTEP,#0
0377 757500        LCALL   UPDATESTEPFIELD
037A 12024D        LCALL   GOTOTIME; FIRST GOTO TIME FIELD
037D 12069A        AJMP     GOOUT3
0380 8146          FILLTOTIME: LCALL   PUTTIMEVALUE
0382 120265        MOV      A,BINARYTIME
0385 E573          CJNE     A,#0,VALIDTIME ; STOP VALUE DETECT
0387 B40013        PUSH     DPH
038A C083          PUSH     DPL
038C C082          MOV      DPTR,#STOPVALUE
038E 900BE3        LCALL   LCDLD
0391 120826        LCALL   GOTOINFO
0394 1206FB        POP      DPL
0397 D082          POP      DPH
0399 D083          AJMP     GOOUT3
039B 8146          VALIDTIME: INC      BINARYSTEP
039D 0572          MOV      A,BINARYSTEP
039F E572          CJNE     A,#99,AUTOINC; FOR TEST 9 VALUE
03A1 B46313        PUSH     DPH
03A4 C083          PUSH     DPL
03A6 C082          MOV      DPTR,#DATAFULL
03A8 900C63        LCALL   LCDLD
03AB 120826        LCALL   GOTOINFO
03AE 1206FB        POP      DPL
03B1 D082          POP      DPH
03B3 D083          AJMP     GOOUT3
03B5 8146

```

```

03B7 AB72          AUTOINC:    MOV      R3,BINARYSTEP
03B9 12088F        LCALL     BINARYTOBCD
03BC 8C75          MOV      BCDSTEP,R4
03BE 12024D        LCALL     UPDATESTEPFIELD
03C1 12069A        LCALL     GOTOTIME
03C4 8146          AJMP      GOOUT3

;*****
03C6 207B0E        EDITDOIT:   JB      INPROGFIELD,EDITTOSTEP
03C9 207A13        JB      INSTEPFIELD,EDITTOTIME
03CC 207918        JB      INTIMEFIELD,EDITTOSTEP1
03CF 12075B        LCALL     MODEEDIT
03D2 120666        LCALL     GOTOPROG
03D5 806F          SJMP      GOOUT3
03D7 120243        EDITTOSTEP: LCALL     UPDATEPROGFIELD
03DA 120680        LCALL     GOTOSTEP
03DD 8067          SJMP      GOOUT3
03DF 12024D        EDITTOTIME: LCALL     UPDATESTEPFIELD

03E2 12069A        LCALL     GOTOTIME
03E5 805F          SJMP      GOOUT3
03E7 120265        EDITTOSTEP1: LCALL     PUTTIMEVALUE

03EA E573          MOV      A,BINARYTIME
03EC B40013        CJNE     A,#0,EDITVALIDDIME ;STOP VALUE DETECT
03EF C083          PUSH     DPH
03F1 C082          PUSH     DPL
03F3 900BÉ3        MOV      DPTR,#STOPVALUE
03F6 120826        LCALL     LCDLD
03F9 1206FB        LCALL     GOTOINFO
03FC D082          POP      DPL
03FE D083          POP      DPH
0400 8146          AJMP      GOOUT3

Q402 120680        EDITVALIDDIME: LCALL     GOTOSTEP
Q405 8146          AJMP      GOOUT3

;*****
0407 207808        RUNDOIT:   JB      INSTARTFIELD,RUNTOSTOP
040A 20770D        JB      INSTOPFIELD,RUNPROGRAM
040D 1208FF        LCALL     RUN
0410 8034          SJMP      GOOUT3
0412 120257        RUNTOSTOP: LCALL     UPDATESTARTFIELD
0415 1206CE        LCALL     GOTOSTOP
0418 802C          SJMP      GOOUT3
041A 12025E        RUNPROGRAM: LCALL     UPDATESTOPFIELD
041D C3            CLR      C
041E E56A          MOV      A,BINARYSTOP
0420 A86B          MOV      R0,BINARYSTART
0422 98            SUBB     A,R0
0423 5013          JNC      VALIDRUN
0425 C083          PUSH     DPH
0427 C082          PUSH     DPL
0429 900BA3        MOV      DPTR,#ERRORRANGE
042C 12070E        LCALL     MODEINFO
042F 1206FB        LCALL     GOTOINFO

```

```

0432 D082          POP      DPL
0434 D083          POP      DPH
0436 800E          SJMP     GOOUT3
0438 1208FF        VALIDRUN: LCALL   RUN
043B 1206FB        LCALL   GOTOINFO
043E 8006          SJMP     GOOUT3

```

;*****

```

0440 12071C        INFODOIT: LCALL   MODEMENU
0443 1206E8        LCALL   GOTOMENU

```

```

0446 020661        GOOUT3:   LJMP     GOHOME

```

;***** CHECK FOR KEY E *****
;
; MAIN MENU
;*****

```

0449 B40E1A        GONEXT4:  CJNE     A,#0EH,GONEXT5
044C 207F14                JB      MENUFLAG,GOOUT4 ;IF MENU GO OUT
044F 207E0B                JB      FILLMODEFLAG,DOIT4
0452 207D08                JB      EDITMODEFLAG,DOIT4
0455 207C05                JB      RUNMODEFLAG,DOIT4
0458 207602                JB      INFOFLAG,DOIT4
045B 8006           SJMP     GOOUT4

```

```

045D 12071C        DOIT4:    LCALL   MODEMENU ;DISPLAY MAIN MENU
0460 1206E8        LCALL   GOTOMENU ;GOTO CURSOR INPUT
0463 020661        GOOUT4:   LJMP     GOHOME

```

;***** CHECK FOR KEY F *****
;
; START/STOP & FOOT SWITCH KEY
;*****

```

0466 B40F03        GONEXT5:  CJNE     A,#0FH,NUMKEY
0469 020661        LJMP     GOHOME

```

;*****
;
; NUMERIC KEY 0 - 9
;*****

```

046C 207F0C        NUMKEY:JB  MENUFLAG,MENUSTATE ;IF IN MENU MODE
046F 207E0C                JB  FILLMODEFLAG,FILLSTATE;IF IN FILL MODE
0472 207D0C                JB  EDITMODEFLAG,EDITSTATE;IF IN EDIT MODE
0475 207C0C                JB  RUNMODEFLAG,RUNSTATE ;IF IN RUN MODE
0478 020661                LJMP   GOHOME
047B 020487        MENUSTATE: LJMP   ENTERMENU
047E 0204E5        FILLSTATE: LJMP   ENTERFILL
0481 0204FA        EDITSTATE: LJMP   ENTEREDIT
0484 02050F        RUNSTATE:  LJMP   ENTERRUN

```

;*****
; WHEN ENTER 1 ,2 OR 3 FOR SELECT MENU
;*****

```

0487 E570      ENTERMENU:  MOV      A,KEYBUFFER
0489 B40115    FILLMODE:   CJNE     A,#01H,EDITMODE
048C E570      MOV        A,KEYBUFFER ;WRITE LCD
048E 2430      ADD        A,#30H
0490 F578      MOV        78H,A; ASCII FOR SELECT
0492 FF        MOV        R7,A
0493 74DC      MOV        A,#0DCH
0495 12080E    LCALL     WRIBYTE
0498 120735    LCALL     MODEFILL;CALL FILL MODE
049B 120666    LCALL     GOTOPROG;CURSOR TO PROG
049E 020661    LJMP      GOHOME
04A1 B40215    EDITMODE:   CJNE     A,#02H,RUNMODE
04A4 E570      MOV        A,KEYBUFFER
04A6 2430      ADD        A,#30H
04A8 F578      MOV        78H,A
04AA FF        MOV        R7,A
04AB 74DC      MOV        A,#0DCH
04AD 12080E    LCALL     WRIBYTE
04B0 12075B    LCALL     MODEEDIT;CALL EDIT MODE
04B3 120666    LCALL     GOTOPROG
04B6 020661    LJMP      GOHOME
04B9 B40315    RUNMODE:    CJNE     A,#03H,NOKEY
04BC E570      MOV        A,KEYBUFFER
04BE 2430      ADD        A,#30H
04C0 F578      MOV        78H,A
04C2 FF        MOV        R7,A
04C3 74DC      MOV        A,#0DCH
04C5 12080E    LCALL     WRIBYTE
04C8 120781    LCALL     MODERUN ;CALL PROG MODE
04CB 1206B4    LCALL     GOTOSTART
04CE 020661    LJMP      GOHOME
04D1 C083      NOKEY:      PUSH     DPH
04D3 C082      PUSH     DPL
04D5 900C23    MOV        DPTR,#SELECTWRONG
04D8 12070E    LCALL     MODEINFO
04DB 1206FB    LCALL     GOTOINFO
04DE D082      POP       DPL
04E0 D083      POP       DPH
04E2 020661    LJMP      GOHOME

```

```

;*****
; TILL HERE WE MANAGE MENU KEY ALREADY
;*****

```

```

;*****
; BEGIN FILL MODE
;*****

```

```

04E5 207B09    ENTERFILL:   JB      INPROGFIELD,ENTERFILLPROG
04E8 207A09    JB      INSTEPFIELD,ENTERFILLSTEP
04EB 207909    JB      INTIMEFIELD,ENTERFILLTIME
04EE 020661    LJMP     GOHOME
04F1 02051E    ENTERFILLPROG: LJMP    ENTERPROGVALUE
04F4 02055D    ENTERFILLSTEP: LJMP    ENTERSTEPVALUE
04F7 02059C    ENTERFILLTIME: LJMP    ENTERTIMEVALUE

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;*****
;
; BEGIN EDIT MODE
;*****
```

```
04FA 207B09 ENTEREDIT: JB INPROGFIELD,ENTEREDITPROG
04FD 207A09 JB INSTEPFIELD,ENTEREDITSTEP
0500 207909 JB INTIMEFIELD,ENTEREDITTIME
0503 020661 LJMP GOHOME
0506 02051E ENTEREDITPROG: LJMP ENTERPROGVALUE
0509 02055D ENTEREDITSTEP: LJMP ENTERSTEPVALUE
050C 02059C ENTEREDITTIME: LJMP ENTERTIMEVALUE
```

```
;*****
; BEGIN RUN MODE
;*****
```

```
050F 207806 ENTERRUN: JB INSTARTFIELD,ENTERRUNSTART
0512 207706 JB INSTOPFIELD,ENTERRUNSTOP
0515 020661 LJMP GOHOME
0518 0205E3 ENTERRUNSTART: LJMP ENTERSTARTVALUE
051B 020622 ENTERRUNSTOP: LJMP ENTERSTOPVALUE
```

```
;*****
; ENTER PROGRAM VALUE
;*****
```

```
051E E569 ENTERPROGVALUE: MOV A,COUNTPROGDIGIT
0520 B40019 CJNE A,#0,PROGDIGIT1
0523 E570 MOV A,KEYBUFFER
0525 2430 ADD A,#30H
0527 F579 MOV DISPLAYPROG1,A
0529 FF MOV R7,A
052A 74CC MOV A,#0CCH
052C 12080E LCALL WRIBYTE
052F E570 MOV A,KEYBUFFER
0531 C4 SWAP A
0532 F574 MOV BCDPROG,A
0534 0569 INC COUNTPROGDIGIT
0536 E569 MOV A,COUNTPROGDIGIT
0538 FA MOV R2,A
0539 020661 LJMP GOHOME
053C B4011B PROGDIGIT1: CJNE A,#1,PROGDIGIT_OUT
053F E570 MOV A,KEYBUFFER
0541 2430 ADD A,#30H
0543 F57A MOV DISPLAYPROG0,A
0545 FF MOV R7,A
0546 74CD MOV A,#0CDH
0548 12080E LCALL WRIBYTE
054B E570 MOV A,KEYBUFFER
054D A874 MOV R0,BCDPROG
054F 48 ORL A,R0
0550 F574 MOV BCDPROG,A
0552 1200F9 LCALL CALPROGVALUE
0555 0569 INC COUNTPROGDIGIT
0557 E569 MOV A,COUNTPROGDIGIT
0559 FA MOV R2,A
055A 020661 PROGDIGIT_OUT: LJMP GOHOME
```

เอกสารนี้อาจมีข้อผิดพลาดได้ กรุณาตรวจสอบข้อมูลก่อนนำไปใช้

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดลอกเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

;*****
;                               ENTER STEP VALUE
;*****

```

```

055D E568      ENTERSTEPVALUE: MOV      A,COUNTSTEPDIGIT
055F B40019    CJNE      A,#0,STEPDIGIT1
0562 E570      MOV      A,KEYBUFFER
0564 2430      ADD      A,#30H
0566 F57B      MOV      DISPLAYSTEP1,A
0568 FF        MOV      R7,A
0569 749C      MOV      A,#09CH
056B 12080E    LCALL     WRIBYTE
056E E570      MOV      A,KEYBUFFER
0570 C4        SWAP     A
0571 F575      MOV      BCDSTEP,A
0573 0568      INC      COUNTSTEPDIGIT
0575 E568      MOV      A,COUNTSTEPDIGIT
0577 FA        MOV      R2,A
0578 020661    LJMP     GOHOME
057B B4011B    STEPDIGIT1: CJNE      A,#1,STEPDIGIT_OUT
057E E570      MOV      A,KEYBUFFER
0580 2430      ADD      A,#30H
0582 F57C      MOV      DISPLAYSTEP0,A
0584 FF        MOV      R7,A
0585 749D      MOV      A,#09DH
0587 12080E    LCALL     WRIBYTE
058A E570      MOV      A,KEYBUFFER
058C A875      MOV      R0,BCDSTEP
058E 48        ORL      A,R0
058F F575      MOV      BCDSTEP,A
0591 120119    LCALL     CALSTEPVALUE
0594 0568      INC      COUNTSTEPDIGIT
0596 E568      MOV      A,COUNTSTEPDIGIT
0598 FA        MOV      R2,A
0599 020661    STEPDIGIT_OUT: LJMP     GOHOME

```

```

;*****
;                               ENTER TIME VALUE
;*****

```

```

059C E567      ENTERTIMEVALUE: MOV      A,COUNTTIMEDIGIT
059E B40019    CJNE      A,#0,TIMEDIGIT1
05A1 E570      MOV      A,KEYBUFFER
05A3 2430      ADD      A,#30H
05A5 F57E      MOV      DISPLAYTIME1,A
05A7 FF        MOV      R7,A
05A8 74DB      MOV      A,#0DBH
05AA 12080E    LCALL     WRIBYTE
05AD E570      MOV      A,KEYBUFFER
05AF C4        SWAP     A
05B0 F576      MOV      BCDTIME,A
05B2 0567      INC      COUNTTIMEDIGIT
05B4 E567      MOV      A,COUNTTIMEDIGIT
05B6 FA        MOV      R2,A
05B7 020661    LJMP     GOHOME

```

```

05BA B40123      TIMEDIGIT1:    CJNE      A,#1,TIMEDIGIT_OUT
05BD E570        MOV            A,KEYBUFFER
05BF 2430        ADD            A,#30H
05C1 F57F        MOV            DISPLAYTIME0,A
05C3 FF          MOV            R7,A
05C4 74DC        MOV            A,#0DCH
05C6 12080E      LCALL         WRIBYTE
05C9 7430        MOV            A,#30H      ; FILL ZERO
05CB FF          MOV            R7,A
05CC 74DD        MOV            A,#0DDH
05CE 12080E      LCALL         WRIBYTE
05D1 E570        MOV            A,KEYBUFFER
05D3 A876        MOV            R0,BCDTIME
05D5 48          ORL            A,R0
05D6 F576        MOV            BCDTIME,A

05D8 120139      LCALL         CALTIMEVALUE

05DB 0567        INC            COUNTTIMEDIGIT
05DD E567        MOV            A,COUNTTIMEDIGIT
05DF FA          MOV            R2,A

05E0 020661      TIMEDIGIT_OUT: LJMP      GOHOME
;*****
; ENTER START VALUE
;*****

05E3 E562        ENTERSTARTVALUE:MOV      A,COUNTSTARTDIGIT
05E5 B40019      CJNE          A,#0,STARTDIGIT1
05E8 E570        MOV            A,KEYBUFFER
05EA 2430        ADD            A,#30H
05EC F565        MOV            DISPLAYSTART1,A
05EE FF          MOV            R7,A
05EF 74CC        MOV            A,#0CCH
05F1 12080E      LCALL         WRIBYTE
05F4 E570        MOV            A,KEYBUFFER
05F6 C4          SWAP          A
05F7 F560        MOV            BCDSTART,A
05F9 0562        INC            COUNTSTARTDIGIT
05FB E562        MOV            A,COUNTSTARTDIGIT
05FD FA          MOV            R2,A
05FE 020661      LJMP            GOHOME

0601 B4011B      STARTDIGIT1:    CJNE      A,#1,STARTDIGIT_OUT

0604 E570        MOV            A,KEYBUFFER
0606 2430        ADD            A,#30H
0608 F566        MOV            DISPLAYSTART0,A
060A FF          MOV            R7,A
060B 74CD        MOV            A,#0CDH
060D 12080E      LCALL         WRIBYTE
0610 E570        MOV            A,KEYBUFFER
0612 A860        MOV            R0,BCDSTART

```

```

0614 48                                ORL      A,R0
0615 F560                              MOV      BCDSTART,A

0617 120159                            LCALL    CALSTARTVALUE

061A 0562                              INC      COUNTSTARTDIGIT
061C E562                              MOV      A,COUNTSTARTDIGIT
061E FA                                MOV      R2,A
061F 020661    STARTDIGIT_OUT:  LJMP     GOHOME

;*****
; ENTER STOP VALUE
;*****

0622 E561    ENTERSTOPVALUE:  MOV      A,COUNTSTOPDIGIT
0624 B40019  CJNE             A,#0,STOPDIGIT1
0627 E570    MOV              A,KEYBUFFER
0629 2430    ADD              A,#30H
062B F563    MOV              DISPLAYSTOP1,A
062D FF      MOV              R7,A
062E 749C    MOV              A,#09CH
0630 12080E  LCALL            WRIBYTE
0633 E570    MOV              A,KEYBUFFER
0635 C4      SWAP             A
0636 F55F    MOV              BCDSTOP,A
0638 0561    INC              COUNTSTOPDIGIT
063A E561    MOV              A,COUNTSTOPDIGIT
063C FA      MOV              R2,A
063D 020661  LJMP             GOHOME

0640 B4011B  STOPDIGIT1:     CJNE     A,#1,STOPDIGIT_OUT
0643 E570    MOV              A,KEYBUFFER
0645 2430    ADD              A,#30H
0647 F564    MOV              DISPLAYSTOP0,A
0649 FF      MOV              R7,A
064A 749D    MOV              A,#09DH
064C 12080E  LCALL            WRIBYTE
064F E570    MOV              A,KEYBUFFER
0651 A85F    MOV              R0,BCDSTOP
0653 48      ORL              A,R0
0654 F55F    MOV              BCDSTOP,A

0656 120179                            LCALL    CALSTOPVALUE

0659 0561                              INC      COUNTSTOPDIGIT
065B E561                              MOV      A,COUNTSTOPDIGIT
065D FA                                MOV      R2,A
065E 020661    STOPDIGIT_OUT:  LJMP     GOHOME

0661 D0D0    GOHOME:         POP      PSW
0663 D0E0    POP      ACC
0665 32      RETI

```

```

;*****
; END OF INTERRUPT SERVICE ROUTINE FOR KEYBOARD
;*****

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

;*****
;   SUBROUTINE GOTO PROG FIELD
;   SET MODE PROG FLAG AND GOTO CURSOR
;   READ VALUE FROM BIN OR BCD CONVERT TO ASCII
;   AND DISPLAY
;*****

0666 D27B GOTOPROG:SETB INPROGFIELD ;SET FLAG TO PROGRAM
0668 C27A          CLR  INSTEPFIELD ;CLEAR STEP FIELD
066A C279          CLR  INTIMEFIELD ;CLEAR TIME FIELD
066C C278          CLR  INSTARTFIELD
066E C277          CLR  INSTOPFIELD

0670 7400          MOV   A,#0
0672 F569          MOV   COUNTPROGDIGIT,A
0674 1201BB        LCALL  DISPTIMEFIELD
0677 7420          MOV   A,#20H
0679 FF            MOV   R7,A
067A 74CB          MOV   A,#0CBH
067C 12080E        LCALL  WRIBYTE
067F 22            RET

;*****
;*****
;   SUBROUTINE GOTO STEP FIELD
;   SET MODE FLAG AND GOTO CURSOR
;*****

0680 D27A GOTOSTEP:SETB INSTEPFIELD ;SET FLAG TO STEP FIELD
0682 C27B          CLR  INPROGFIELD ;CLEAR PROG FIELD
0684 C279          CLR  INTIMEFIELD ;CLEAR TIME FIELD
0686 C278          CLR  INSTARTFIELD
0688 C277          CLR  INSTOPFIELD

068A 7400          MOV   A,#0
068C F568          MOV   COUNTSTEPDIGIT,A
068E 1201AA        LCALL  DISPSTEPFIELD
0691 7420          MOV   A,#20H
0693 FF            MOV   R7,A
0694 749B          MOV   A,#09BH
0696 12080E        LCALL  WRIBYTE
0699 22            RET

;*****
;
;   SUBROUTINE GOTO TIME FIELD
;   SET MODE TIME FLAG AND GOTO CURSOR
;*****

069A D279 GOTOTIME:SETB INTIMEFIELD ;SET FLAG TO TIME FIELD
069C C27B          CLR  INPROGFIELD ;CLEAR PROG FIELD
069E C27A          CLR  INSTEPFIELD ;CLEAR STEP FIELD
06A0 C278          CLR  INSTARTFIELD
06A2 C277          CLR  INSTOPFIELD

```

```

06A4 7400      MOV      A,#0
06A6 F567      MOV      COUNTTIMEDIGIT,A
06A8 1201BB    LCALL     DISPTIMEFIELD
06AB 7420      MOV      A,#20H
06AD FF        MOV      R7,A
06AE 74DA      MOV      A,#0DAH
06B0 12080E    LCALL     WRIBYTE
06B3 22        RET

; *****

;
;      SUBROUTINE GOTO START FIELD
;      SET MODE FLAG AND GOTO CURSOR
; *****

06B4 D278 GOTOSTART:SETB INSTARTFIELD ;SET FLAG TO STEP
06B6 C277      CLR      INSTOPFIELD ;CLEAR PROG FIELD
06B8 C279      CLR      INTIMEFIELD ;SET FLAG TO TIME
06BA C27B      CLR      INPROGFIELD ;CLEAR PROG FIELD
06BC C27A      CLR      INSTEPFIELD ;CLEAR STEP FIELD
06BE 7400      MOV      A,#0
06C0 F562      MOV      COUNTSTARTDIGIT,A

06C2 1201D4    LCALL     DISPSTARTFIELD
06C5 7420      MOV      A,#20H
06C7 FF        MOV      R7,A
06C8 74CB      MOV      A,#0CBH
06CA 12080E    LCALL     WRIBYTE
06CD 22        RET

; *****

;
;      SUBROUTINE GOTO STOP FIELD
;      SET MODE FLAG AND GOTO CURSOR
; *****

06CE D277 GOTOSTOP:SETB INSTOPFIELD ;SET FLAG TO STEP
06D0 C278      CLR      INSTARTFIELD ;CLEAR PROG FIELD
06D2 C279      CLR      INTIMEFIELD ;SET FLAG TO TIME
06D4 C27B      CLR      INPROGFIELD ;CLEAR PROG FIELD
06D6 C27A      CLR      INSTEPFIELD ;CLEAR STEP FIELD
06D8 7400      MOV      A,#0
06DA F561      MOV      COUNTSTOPDIGIT,A
06DC 1201E5    LCALL     DISPSTOPFIELD
06DF 7420      MOV      A,#20H
06E1 FF        MOV      R7,A
06E2 749B      MOV      A,#09BH
06E4 12080E    LCALL     WRIBYTE
06E7 22        RET

; *****

;
;      SUBROUTINE GOTO MENU FIELD
;      WRITE ZERO AND GOTO MENU DISPLAY
; *****

06E8 C277 GOTOMENU:CLR INSTOPFIELD ;SET FLAG TO STEP
06EA C278      CLR      INSTARTFIELD ;CLEAR PROG FIELD
06EC C279      CLR      INTIMEFIELD ;SET FLAG TO TIME

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

06EE C27B          CLR    INPROGFIELD    ;CLEAR PROG FIELD
06F0 C27A          CLR    INSTEPFIELD    ;CLEAR STEP FIELD
06F2 7420          MOV     A,#20H
06F4 FF            MOV     R7,A
06F5 74DB          MOV     A,#0DBH
06F7 12080E        LCALL    WRIBYTE
06FA 22            RET

;*****

06FB C277 GOTOINFO:CLR    INSTOPFIELD    ;SET FLAG TO STEP FIELD
06FD C278          CLR    INSTARTFIELD    ;CLEAR PROG FIELD
06FF C279          CLR    INTIMEFIELD    ;SET FLAG TO TIME FIELD
0701 C27B          CLR    INPROGFIELD    ;CLEAR PROG FIELD
0703 C27A          CLR    INSTEPFIELD    ;CLEAR STEP FIELD

0705 7420          MOV     A,#20H
0707 FF            MOV     R7,A
0708 74DF          MOV     A,#0DFH
070A 12080E        LCALL    WRIBYTE
070D 22            RET

;*****

070E 120826        MODEINFO:  LCALL    LCDLD
0711 C27E          CLR     FILLMODEFLAG
0713 C27D          CLR     EDITMODEFLAG
0715 C27C          CLR     RUNMODEFLAG
0717 C27F          CLR     MENUFLAG
0719 D276          SETB    INFOFLAG
071B 22            RET

;*****
; SUBROUTINE MENU MODE SHOW
;*****

071C 900A63        MODEMENU:  MOV     DPTR,#MENU
071F 120826        LCALL    LCDLD
0722 C27E          CLR     FILLMODEFLAG
0724 C27D          CLR     EDITMODEFLAG
0726 C27C          CLR     RUNMODEFLAG
0728 D27F          SETB    MENUFLAG
072A C276          CLR     INFOFLAG

072C 7420          MOV     A,#20H
072E FF            MOV     R7,A ;ASCII IN R7 FOR DISPLAY
072F 74DB          MOV     A,#0DBH;LOCATION ADDRESS AT LCD
0731 12080E        LCALL    WRIBYTE
0734 22            RET

;*****
;*****
; SUBROUTINE PROGRAM MODE SHOW
;*****

0735 900AA3        MODEFILL:  MOV     DPTR,#MESSPROG
0738 120826        LCALL    LCDLD
073B D27E          SETB    FILLMODEFLAG
073D C27D          CLR     EDITMODEFLAG
073F C27C          CLR     RUNMODEFLAG

```

```

;*****
;  L I B R A R Y    F U N C T I O N
;*****

;*****
;*****
;
;      SUBROUTINE CHANGE BCD TO BINARY
;      BCD 1 BYTE TO BINARY 1 BYTE
;      INPUT : BCD      IN R7
;      OUTPUT: BINARY IN R6
;      USE R6,R7
;      R7 =====> BCD INPUT
;      R6 =====> BINARY OUTPUT
;*****

```

```

07A4 7E00      BCDTOBINARY:  MOV     R6,#0
07A6 C3                CLR     C
07A7 7D08                MOV     R5,#8 ;FOR COUNTER 8 BIT
07A9 EF      NEXTBIT:   MOV     A,R7
07AA 13                RRC     A
07AB C0D0                PUSH    PSW
07AD C3                CLR     C
07AE 30E702          JNB     0E7H,X1
07B1 9430          SUBB     A,#30H
07B3 30E303          JNB     0E3H,X2;R7 BCD 8 9
07B6 C3                CLR     C
07B7 9403          SUBB     A,#03H ;R6 BINARY 1000 0001
07B9 FF      X2:        MOV     R7,A
07BA D0D0                POP     PSW
07BC EE                MOV     A,R6
07BD 13                RRC     A
07BE FE                MOV     R6,A
07BF DDE8          DJNZ     R5,NEXTBIT
07C1 22                RET
;*****
;*****
;
;      SUBROUTINE WRITE ONE PAGE
;      INPUT : POINTER TO ADDRESS OF BUFFER
;*****

```

```

07C2 C083      WR_PAGE:  PUSH     DPH
07C4 C082                PUSH    DPL
07C6 120826          LCALL     LCDLD
07C9 D082                POP     DPL
07CB D083                POP     DPH
07CD 22                RET
;*****
;*****
;
;      INPUT: POINTER TO ADDRESS OF EACH LINE
;      IN REGISTER A
;      LINE 1 => 80H - 8FH
;      LINE 2 => C0H - CFH
;      LINE 3 => 90H - 9FH
;      LINE 4 => D0H - DFH

```

```
07CE 12083B WR_LINE:LCALL LCDLDS ;WRITE 1 LINE FOR 16 CHAR
07D1 22 RET
```

```
;*****
```

```
;***** DELAY SUBROUTINE*****
; DELAY SUBROUTINE
; INPUT = R6
; USED REGISTER = R3,R4
```

```
07D2 7B00 DELAY: MOV R3,#0
07D4 7C00 DELAY1: MOV R4,#0
07D6 DCFE DJNZ R4,$
07D8 DBFA DJNZ R3,DELAY1
07DA DEF6 DJNZ R6,DELAY
07DC 22 RET
```

```
;*****
```

```
;***** SUBROUTINE LCD INITIALIZE ****
```

```
07DD 7438 INITLCD: MOV A,#00111000B ;FUNCTION SET
07DF 120874 LCALL LCDWI
07E2 740F MOV A,#00001111B ;DISPLAY ON/OFF
07E4 120874 LCALL LCDWI
07E7 7406 MOV A,#6 ;CURSOR NOT INC WHEN R/W
07E9 120874 LCALL LCDWI
07EC 7401 MOV A,#01H ;CLEAR
07EE 120874 LCALL LCDWI
07F1 22 RET
```

```
;*****
```

```
;*****
```

```
; CURSOR MOVEMENT FUNCTION
```

```
;*****
```

```
07F2 C082 GORIGHT: PUSH DPL
07F4 C083 PUSH DPH
07F6 7414 MOV A,#14H
07F8 120874 LCALL LCDWI
07FB D083 POP DPH
07FD D082 POP DPL
07FF 22 RET
0800 C082 GOLEFT: PUSH DPL
0802 C083 PUSH DPH
0804 7410 MOV A,#10H
0806 120874 LCALL LCDWI
0809 D083 POP DPH
080B D082 POP DPL
080D 22 RET
```

```
;*****
```

```
;*****
```

```
; WRITE BYTE AT CURSOR
```

```
; WRITE ASCII ONE CHAR AT POSITION OF CURSOR
```

```
; INPUT ADDRESS IN REGISTER A
```

```
; DATA IN => R7
```

```
;*****
```

เอกสารนี้เป็นเอกสารที่สงวนไว้;*****

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

080E C083      WRIBYTE:  PUSH    DPH    ; WRITE MODE AT ADDRESS
0810 C082                PUSH    DPL
0812 120874      LCALL    LCDWI
0815 D082                POP     DPL
0817 D083                POP     DPH
0819 EF          MOV     A,R7 ;GET ASCII DATA
081A C083                PUSH    DPH
081C C082                PUSH    DPL
081E 120859      LCALL    LCDWD
0821 D082                POP     DPL
0823 D083                POP     DPH
0825 22          RET

```

```

;*****

```

```

;*****

```

```

; LCD DISPLAY
; LOAD DATA TO LCD MODULE
; IN = DPTR START BLOCK (64 BYTE)
; REG = R2,R3,R4

```

```

0826 7480      LCDLD:  MOV     A,#80H    ;SET ADDRESS LINE 1
0828 12083B      LCALL    LCDLDS
082B 74C0      MOV     A,#0C0H    ;SET ADDRESS LINE 2
082D 12083B      LCALL    LCDLDS
0830 7490      MOV     A,#90H    ;SET ADDRESS LINE 3
0832 12083B      LCALL    LCDLDS
0835 74D0      MOV     A,#0D0H    ;SET ADDRESS LINE 4
0837 12083B      LCALL    LCDLDS
083A 22          RET

```

```

083B C083      LCDLDS:  PUSH    DPH    ;LOAD SUB.
083D C082                PUSH    DPL
083F 120874      LCALL    LCDWI    ;WRITE ADDRESS
0842 D082                POP     DPL
0844 D083                POP     DPH

```

```

0846 7810      MOV     R0,#16    ;16 CHARACTER
0848 E4        LCDLDS1: CLR     A    ;DPTR => RAM
0849 93        MOV     A,@A+DPTR;GET DATA FROM RAM
084A C083      PUSH    DPH
084C C082      PUSH    DPL
084E 120859      LCALL    LCDWD
0851 D082      POP     DPL
0853 D083      POP     DPH
0855 A3        INC     DPTR
0856 D8F0      DJNZ    R0,LCDLDS1
0858 22          RET

```

```

;*****

```

```

;*****

```

```

; LCD WRITE DATA (RS = 1)

```

```

; IN = A

```

```

; REG = A,DPTR

```

```

;*****

```

```

0859 90E0E0 . LCDWD:  MOV     DPTR,#PORTA ;DATA TO PORT A
085C F0        MOVX    @DPTR,A

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

085D 90E0E1      MOV    DPTR,#PORTB ;PORTB RD MODIFY WRI
0860 E0          MOVX  A,@DPTR
0861 D2E0        SETB  0E0H          ; RS = 1
0863 C2E1        CLR   0E1H          ; R/W = 0
0865 C2E2        CLR   0E2H          ; ENABLE = 0
0867 F0          MOVX  @DPTR,A
0868 D2E2        SETB  0E2H          ; ENABLE = 1
086A F0          MOVX  @DPTR,A
086B C2E2        CLR   0E2H          ; ENABLE = 0
086D F0          MOVX  @DPTR,A
086E 7400        MOV   A,#0          ; DELAY
0870 14          LCDWD1:DEC  A
0871 70FD        JNZ   LCDWD1
0873 22          RET

```

;*****

;***** LCD WI *****

; LCD WRITE INSTRUCTION (RS = 0)

; IN = A

; REG = A,DPTR

;*****

```

0874 90E0E0      LCDWI: MOV    DPTR,#PORTA ;DATA TO PORT A
0877 F0          MOVX  @DPTR,A
0878 90E0E1      MOV    DPTR,#PORTB ;PORTB READ MODIFY
087B E0          MOVX  A,@DPTR
087C C2E0        CLR   0E0H          ;RS = 0
087E C2E1        CLR   0E1H          ;R/W = 0
0880 C2E2        CLR   0E2H          ;ENABLE = 0
0882 F0          MOVX  @DPTR,A
0883 D2E2        SETB  0E2H          ;ENABLE = 1
0885 F0          MOVX  @DPTR,A
0886 C2E2        CLR   0E2H          ;ENABLE = 0
0888 F0          MOVX  @DPTR,A
0889 7400        MOV   A,#0          ;DELAY
088B 14          LCDWI1: DEC  A
088C 70FD        JNZ   LCDWI1
088E 22          RET

```

;*****

; CONVERT BINARY TO BCD CODE

; R3 =====> KEEP INPUT BINARY DATA

; R4 =====> KEEP BCD OUTPUT

;*****

```

088F 7C00 BINARYTOBCD:MOV    R4,#0 ; CLEAR R4 FOR KEEP BCD
0891 C2D7      CLR   CY
0893 85D700    MOV    00H,CY
0896 7A08      MOV    R2,#8 ; NOW WE GET NO. OF BIT
0898 8500D7    LOOP11:MOV   CY,00H
089B EB        MOV    A,R3 ; R3 KEEP BINARY
089C 33        RLC   A
089D FB        MOV    R3,A
089E 85D700    MOV    00H,CY
08A1 EC        MOV    A,R4
08A2 F8        MOV    R0,A

```

```

08A3 38          ADDC    A,R0
08A4 D4          DA      A
08A5 FC          MOV     R4,A
08A6 DAFO        DJNZ    R2,LOOP11
08A8 22          RET

;*****
;
; SUBROUTINE    CONVERT BINARY 16 BIT BINARY TO BCD
; AND          CONVERT BINARY IN BINARYCOUNTDOT0,1
;              CONVERT TO BCD    INBCDCOUNTDOT0,1,2
; INPUT :      R7,R6,R1,R0
;              BINARY VALUE IN
;              BY      MOV R7,##          LOW BYTE
;                      MOV R6,##          HIGH BYTE
;
;              R1 => POINT TO MEMORY THAT KEEP BCD DATA
;              BY      MOV R1,##          LSB
;              R0 => POINT TO MEMORY THAT KEEP ASCII DATA
;                      MOV R0,##XX        LSB
;
; OUTPUT:      ASCII IN SPECIFIED ADDRESS => XX,XX,XX,XX,XX
; USE:         R4 FOR COUNTER
;              R3 FOR KEEP VECTOR OF R1
;*****

08A9 C3          BINTOBCDDOT:CLR    C          ;CLEAR CARRY FLAG
08AA 9200         MOV     00H,C          ;SAVE CARRY FLAG
;
;              ;CLEAR BCD DATA FROM 5A,5B,5C

08AC 7C03         MOV     R4,#3
08AE E9          MOV     A,R1
08AF FB          MOV     R3,A          ;KEEP VECTOR OF BCD MEMORY
08B0 7400         MOV     A,#00
08B2 F7          DCLR1:  MOV     @R1,A
08B3 19          DEC     R1
08B4 DCFC         DJNZ    R4,DCLR1
08B6 EB          MOV     A,R3
08B7 F9          MOV     R1,A ;RETURN VALUE OF VECTOR R1
;
;              ;CLEAR ASCII DATA FROM 59,58,57,56,55

08B8 7C05         MOV     R4,#5
08BA E8          MOV     A,R0
08BB FB          MOV     R3,A ;KEEP VECTOR OF R1
08BC 7400         MOV     A,#00
08BE F6          DCLR:   MOV     @R0,A
08BF 18          DEC     R0
08C0 DCFC         DJNZ    R4,DCLR
08C2 EB          MOV     A,R3;RETURN VECTOR OF R1
08C3 F8          MOV     R0,A
08C4 7A10         MOV     R2,#16 ; NOW WE GET NUMBER OF BIT 16 BIT
08C6 E9          MOV     A,R1
08C7 FB          MOV     R3,A          ;R3 KEEP VECTOR

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

08C8 EB          DLOOP1:      MOV  A,R3
08C9 F9          MOV  R1,A
08CA A200        MOV  C,00H
08CC EF          MOV  A,R7
08CD 33          RLC  A
08CE FF          MOV  R7,A
08CF EE          MOV  A,R6
08D0 33          RLC  A
08D1 FE          MOV  R6,A
08D2 9200        MOV  00H,C
08D4 7C03        MOV  R4,#3;GET COUNTER 2 BYTE
08D6 E7          DREP:       MOV  A,@R1
08D7 FD          MOV  R5,A
08D8 3D          ADDC A,R5
08D9 D4          DA  A
08DA F7          MOV  @R1,A
08DB 19          DEC  R1
08DC DCF8        DJNZ R4,DREP      ;EVERY BYTE
08DE DAE8        DJNZ R2,DLOOP1
08E0 22          RET

;***** END OF SUBROUTINE *****
;*****
;SUBROUTINE CONVERT BCD TO ASCII CODE
;INPUT  :      R0 => ASCII MEMORY XX XX XX XX <=
;          R1 => BCD MEMORY  XX XX XX <=
;  NOW R1 => 5CH
;  R0  => 59H
;*****

08E1 7A03        BCDTOASCDOT: MOV  R2,#3          ;COUNTER
08E3 E7          DNEXT:      MOV  A,@R1
08E4 540F        ANL  A,#0FH
08E6 4430        ORL  A,#30H
08E8 F6          MOV  @R0,A;R0 POINT TO ASCII
08E9 18          DEC  R0
08EA B85401      CJNE R0,#54H,CONTINUE
08ED 22          RET          ; RETURN
08EE E7          CONTINUE:   MOV  A,@R1 ; R1 POINT TO BCD
08EF 54F0        ANL  A,#0F0H
08F1 03          RR  A
08F2 03          RR  A
08F3 03          RR  A
08F4 03          RR  A
08F5 540F        ANL  A,#0FH
08F7 4430        ORL  A,#30H
08F9 F6          MOV  @R0,A
08FA 18          DEC  R0
08FB 19          DEC  R1
08FC DAE5        DJNZ R2,DNEXT

08FE 22          OUT00:      RET
;***** END SUB ROUTINE *****

```

```

;*****
;
;          RUNNING PROGRAM
;*****
08FF A86B          RUN:          MOV    R0,BINARYSTART
0901 A96A          MOV    R1,BINARYSTOP
0903 7400          MOV    A,#0
0905 F577          MOV    77H,A
0907 F55E          MOV    5EH,A
0909 E56B          MOV    A,BINARYSTART
090B F571          EACHPROGRAM: MOV    BINARYPROG,A
090D 7400          MOV    A,#0
090F F572          MOV    BINARYSTEP,A;GET OFFSET
0911 1202A8        RUNLOOP1:    LCALL   FINDADDRESS
0914 12029E        LCALL   READADDRESS ; VALUE IN BINARYTIME

0917 E573          MOV    A,BINARYTIME ; IS THE END VALUE?
0919 B40002        CJNE    A,#0,PROCESS
091C 217A          AJMP    NEXTPROGRAM

091E FF           PROCESS:    MOV    R7,A ;R7 GET UNITS DELAY
091F C3           CLR    C
0920 E577          MOV    A,77H
0922 2F           ADD    A,R7 ;<== ADD FOR COUNT
0923 F577          MOV    77H,A

; CHECK IF XY READY LOGIC "1"

0925 00           WAITFORXY:  NOP
0926 3096FC        JNB     RX,WAITFORXY
0929 C297          CLR     TX ; HIGH TO XYTABLE
092B E573          MOV     A,BINARYTIME

092D FC           MOV     R4,A
092E 755064        waiting:  MOV     MSL,#100
0931 754F00        MOV     MSH,#0
0934 120E23        LCALL   MILO
0937 DCF5          DJNZ    R4,WAITING
0939 D297          SETB    TX ; LOW

;*****

; PROCESS COUNTING DOT

093B C083          PUSH     DPH
093D C082          PUSH     DPL
093F 855E82        MOV     DPL,BINARYCOUNTDOT0
0942 855D83        MOV     DPH,BINARYCOUNTDOT1
0945 A3            INC     DPTR
0946 85835D        MOV     BINARYCOUNTDOT1,DPH
0949 85825E        MOV     BINARYCOUNTDOT0,DPL
094C D082          POP      DPL
094E D083          POP      DPH

0950 AF5E          MOV     R7,BINARYCOUNTDOT0
0952 AE5D          MOV     R6,BINARYCOUNTDOT1

```

0954 795C	MOV	R1,#5CH
0956 7859	MOV	R0,#59H
0958 1208A9	LCALL	BINTOBCDDOT
095B 795C	MOV	R1,#5CH
095D 7859	MOV	R0,#59H
095F 1208E1	LCALL	BCDTOASCDOT
0962 C083	PUSH	DPH
0964 C082	PUSH	DPL
0966 900D23	MOV	DPTR,#COUNTDOT
0969 120826	LCALL	LCDLD
096C D082	POP	DPL
096E D083	POP	DPH
0970 1201F6	LCALL	DISPDOTFIELD
0973 0572	INC	BINARYSTEP;IF NOT 99
0975 E572	MOV	A,BINARYSTEP
0977 B46397	CJNE	A,#99,RUNLOOP1
097A E571	NEXTPROGRAM: MOV	A,BINARYPROG
097C B56A3B	CJNE	A,BINARYSTOP,NEXTPROCESS
	;*****	
	; FINISHED RUNNING	
	;*****	
	; DISPLAY DOT FIELD	
097F AF5E	MOV	R7,BINARYCOUNTDOT0
0981 AE5D	MOV	R6,BINARYCOUNTDOT1
0983 795C	MOV	R1,#5CH
0985 7859	MOV	R0,#59H
0987 1208A9	LCALL	BINTOBCDDOT
098A 795C	MOV	R1,#5CH
098C 7859	MOV	R0,#59H
098E 1208E1	LCALL	BCDTOASCDOT
0991 C083	PUSH	DPH
0993 C082	PUSH	DPL
0995 900D23	MOV	DPTR,#COUNTDOT
0998 120826	LCALL	LCDLD
099B D082	POP	DPL
099D D083	POP	DPH
099F 1201F6	LCALL	DISPDOTFIELD
09A2 C083	PUSH	DPH
09A4 C082	PUSH	DPL
09A6 900CE3	MOV	DPTR,#FINISHEDJOB
09A9 120826	LCALL	LCDLD
09AC D082	POP	DPL
09AE D083	POP	DPH
09B0 7414	MOV	A,#20
09B2 FF	MOV	R7,A
09B3 74DF	MOV	A,#0DFH
09B5 12080E	LCALL	WRIBYTE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

0A63	205B315D20MENU:	DFB	" [1] FILL`MODE	"
0A73	205B325D20	DFB	" [2] EDIT MODE	"
0A83	205B335D20	DFB	" [3] RUN MODE	"
0A93	2053454C45	DFB	" SELECT =>	"
0AA3	2020203D3EMESSPROG:	DFB	" => FILL <=	"
0AB3	202050524F	DFB	" PROGRAM.. XX	"
0AC3	2020535445	DFB	" STEP..... XX	"
0AD3	202054494D	DFB	" TIME.....XXX	"
0AE3	2020203D3EMESSEEDIT:	DFB	" => EDIT <=	"
0AF3	202050524F	DFB	" PROGRAM.. XX	"
0B03	2020535445	DFB	" STEP..... XX	"
0B13	202054494D	DFB	" TIME.....XXX	"
0B23	2020203D3EMESSRUN:	DFB	" => RUN <=	"
0B33	2053544152	DFB	" START PROG XX	"
0B43	2053544F50	DFB	" STOP PROG XX	"
0B53	20454E5445	DFB	" ENTER TO START	"
0B63	3D3D3D3D3DTESTPAT:	DFB	"=====	"
0B73	202020202E	DFB	" ..TEST..	"
0B83	2020204C43	DFB	" LCD MODULE	"
0B93	3D3D3D3D3D	DFB	"=====	"
0BA3	2020494E50ERRORRANGE:	DFB	" INPUT ERROR	"
0BB3	2020202049	DFB	" INVALID	"
0BC3	2020202020	DFB	" RANGE	"
0BD3	2020202121	DFB	" !!!!!!!!!!!	"
0BE3	2020202020STOPVALUE:	DFB	" STOP	"
0BF3	2020202020	DFB	" VALUE	"
0C03	2020202044	DFB	" DETECTED	"
0C13	2020505245	DFB	" PRESS ENTER	"
0C23	2020202020SELECTWRONG:	DFB	" PLEASE	"
0C33	2020202020	DFB	" SELECT	"
0C43	2020202031	DFB	" 1,2 OR 3	"
0C53	2020204F4E	DFB	" ONLY O.K.?	"
0C63	2020202046DATAFULL:	DFB	" FINISHED	"
0C73	2020505245	DFB	" PRESS ENTER	"
0C83	2020202054	DFB	" TO SELECT	"
0C93	20204E4557	DFB	" NEW PROGRAM	"
0CA3	2020202020DONEJOB:	DFB	" SYSTEM	"
0CB3	2020202020	DFB	" RUNNING	"
0CC3	2020202020	DFB	" O.K.	"
0CD3	2020202020	DFB	"	"
0CE3	2020202020FINISHEDJOB:	DFB	" JOB	"
0CF3	2020202046	DFB	" FINISHED	"
0D03	20454E5445	DFB	" ENTER & FOOTSW	"
0D13	2020464F52	DFB	" FOR NEXT JOB	"

0D23	2020203D3E	COUNTDOT:	DFB	"	=> DOING <=	"
0D33	2020202020		DFB	"	DOT	"
0D43	2020202050		DFB	"	PROCESS	"
0D53	2020202E2E		DFB	"		"
0D63	2020202020	BLANK:	DFB	"		"
0D73	2020202020		DFB	"		"
0D83	2020202020		DFB	"		"
0D93	2020202020		DFB	"		"
0DA3	3D3D3D3D3D	SEMIMODE:	DFB	"	=====	"
0DB3	2020202053		DFB	"	SEMI-AUTO	"
0DC3	2020202020		DFB	"	M O D E	"
0DD3	3D3D3D3D3D		DFB	"	=====	"
0DE3	3D3D3D3D3D	AUTOMODE:	DFB	"	=====	"
0DF3	2020202041		DFB	"	AUTOMATIC	"
0E03	2020202020		DFB	"	M O D E	"
0E13	3D3D3D3D3D		DFB	"	=====	"

```

;*****
;                                DELAYTIME      100 MS
;*****
0E23 E550      MILO:      MOV      A,MSL
0E25 B40007    CJNE      A,#0,MIL1
0E28 E54F      MOV      A,MSH
0E2A B40002    CJNE      A,#0,MIL1
0E2D 802A      SJMP     DONEDEE
0E2F 90FC7F    MOV      DPTR,#0FC7FH
0E32 C28D      MIL:      CLR      TF0
0E34 C28C      CLR      TR0
0E36 85838C    MOV      TH0,DPH
0E39 85828A    MOV      TL0,DPL
0E3C 758901    MOV      TMOD,#01
0E3F D28C      SETB     TR0
0E41 308DFD    JNB      TF0,$
0E44 D55013    DJNZ     MSL,MIL3

;-----
0E47 C0E0      PUSH     ACC
0E49 E54F      MOV      A,MSH
0E4B B40004    CJNE      A,#0,MSHDOWN
0E4E D0E0      POP      ACC
0E50 8007      SJMP     DONEDEE
0E52 14        MSHDOWN: DEC      A
0E53 F54F      MOV      MSH,A
0E55 D0E0      POP      ACC
0E57 80D9      SJMP     MIL

;-----
0E59 22        DONEDEE: RET
0E5A 7803      MIL3:    MOV      R0,#3
0E5C D8FE      DJNZ     R0,$
0E5E 00        NOP
0E5F 80D1      SJMP     MIL
;*****

```

0D23	2020203D3ECOUNTDOT:	DFB	"	=> DOING <=	"
0D33	2020202020	DFB	"	DOT	"
0D43	2020202050	DFB	"	PROCESS	"
0D53	2020202E2E	DFB	"		"
0D63	2020202020BLANK:	DFB	"		"
0D73	2020202020	DFB	"		"
0D83	2020202020	DFB	"		"
0D93	2020202020	DFB	"		"
0DA3	3D3D3D3D3DSEMIMODE:	DFB	"	=====	"
0DB3	2020202053	DFB	"	SEMI-AUTO	"
0DC3	2020202020	DFB	"	M O D E	"
0DD3	3D3D3D3D3D	DFB	"	=====	"
0DE3	3D3D3D3D3DAUTOMODE:	DFB	"	=====	"
0DF3	2020202041	DFB	"	AUTOMATIC	"
0E03	2020202020	DFB	"	M O D E	"
0E13	3D3D3D3D3D	DFB	"	=====	"

```

;*****
;                                DELAYTIME    100 MS
;*****
0E23 E550      MIL0:      MOV      A,MSL
0E25 B40007    CJNE      A,#0,MIL1
0E28 E54F      MOV      A,MSH
0E2A B40002    CJNE      A,#0,MIL1
0E2D 802A      SJMP      DONEDEE
0E2F 90FC7F    MIL1:      MOV      DPTR,#0FC7FH
0E32 C28D      MIL:      CLR      TF0
0E34 C28C      CLR      TR0
0E36 85838C    MOV      TH0,DPH
0E39 85828A    MOV      TL0,DPL
0E3C 758901    MOV      TMOD,#01
0E3F D28C      SETB     TR0
0E41 308DFD    JNB      TF0,$
0E44 D55013    DJNZ     MSL,MIL3

;-----
0E47 C0E0      PUSH     ACC
0E49 E54F      MOV      A,MSH
0E4B B40004    CJNE      A,#0,MSHDOWN
0E4E D0E0      POP      ACC
0E50 8007      SJMP     DONEDEE
0E52 14        MSHDOWN:  DEC      A
0E53 F54F      MOV      MSH,A
0E55 D0E0      POP      ACC
0E57 80D9      SJMP     MIL

;-----

0E59 22        DONEDEE:  RET

0E5A 7803      MIL3:      MOV      R0,#3
0E5C D8FE      DJNZ     R0,$
0E5E 00        NOP
0E5F 80D1      SJMP     MIL

;*****
0000          END

```

00D6	AC	00E0	ACC
00E2	ACC2	02C5	ADDOFFSET
0078	ASCIISELECT	03B7	AUTOINC
00F0	B	005C	BCDCOUNTDOT0
005A	BCDCOUNTDOT2	0074	BCDPROG
0075	BCDSTEP	005F	BCDSTOP
08E1	BCDTOASCDOT	07A4	BCDTOBINARY
005D	BINARYCOUNTDOT1	0071	BINARYPROG
0072	BINARYSTEP	006A	BINARYSTOP
088F	BINARYTOBCD	08A9	BINTOBCDDOT
00F9	CALPROGVALUE	0159	CALSTARTVALUE
0179	CALSTOPVALUE	0139	CALTIMEVALUE
08EE	CONTINUE	0D23	COUNTDOT
0053	COUNTINTTIME1	006E	COUNTKEY
006E	COUNTSELECTDIGIT	0062	COUNTSTARTDIGIT
0061	COUNTSTOPDIGIT	0067	COUNTTIMEDIGIT
0C63	DATAFULL	2200	DATASEGMENT
08B2	DCLR1	02BF	DD1
07D2	DELAY	07D4	DELAY1
021A	DIS00	01F6	DISPDOTFIELD
0056	DISPLAYDOT1	0057	DISPLAYDOT2
0059	DISPLAYDOT4	007A	DISPLAYPROG0
0066	DISPLAYSTART0	0065	DISPLAYSTART1
007B	DISPLAYSTEP1	0064	DISPLAYSTOPO
007F	DISPLAYTIME0	007E	DISPLAYTIME1
0199	DISPPROGFIELD	01D4	DISPSTARTFIELD
01E5	DISPSTOPFIELD	01BB	DISPTIMEFIELD
08E3	DNEXT	02F6	DOEDIT0
033A	DOEDIT2	0301	DOIT0
0345	DOIT2	045D	DOIT4
0CA3	DONEJOB	0083	DPH
08D6	DREP	00AF	EA
03C6	EDITDOIT	04A1	EDITMODE
0481	EDITSTATE	03D7	EDITTOSTEP
03DF	EDITTOTIME	0402	EDITVALIDTIME
Q506	ENTEREDITPROG	0509	ENTEREDITSTEP
04E5	ENTERFILL	04F1	ENTERFILLPROG
04F7	ENTERFILLTIME	0487	ENTERMENU
050F	ENTERRUN	0518	ENTERRUNSTART
05E3	ENTERSTARTVALUE	055D	ENTERSTEPVALUE
059C	ENTERTIMEVALUE	0BA3	ERRORRANGE
00A9	ETO	00AB	ET1
00AA	EX1	00D5	F0
Q489	FILLMODE	007E	FILLMODEFLAG
0374	FILLTOSTEP	0382	FILLTOTIME
02D8	FINISHED	0CE3	FINISHEDJOB
0661	GOHOME	0800	GOLEFT
0329	GONEXT2	034B	GONEXT3
0466	GONEXT5	0304	GOOUT0
0348	GOOUT2	0446	GOOUT3
07F2	GORIGHT	06FB	GOTOINFO
0666	GOTOPROG	06B4	GOTOSTART
06CE	GOTOSTOP	069A	GOTOTIME
008B	IE1	00A8	IEC
0364	INFODOITXX	0076	INFOFLAG

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น ลิขสิทธิ์ทางปัญญาให้ด้วยและต้องอ้างถึงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

007B	INPROGFIELD	0078	INSTARTFIELD
0077	INSTOPFIELD	0079	INTIMEFIELD
02D9	ISRKB	0088	ITO
0070	KEYBUFFER	0826	LCDLD
0848	LCDLDS1	0859	LCDWD
0874	LCDWI	088B	LCDWI1
02B3	LOOKUP1	02B8	LOOKUP2
02CD	LOOKUP5	02CF	LOOKUP6
0A63	MENU	007F	MENUFLAG
0AE3	MESSEEDIT	0AA3	MESSPROG
0E32	MIL	0E23	MIL0
0E5A	MIL3	006F	MODE
0735	MODEFILL	070E	MODEINFO
0781	MODERUN	004F	MSH
0050	MSL	0A23	NAME
09BA	NEXTPROCESS	097A	NEXTPROGRAM
046C	NUMKEY	08FE	OUT00
00D0	P	0080	P0
00A0	P2	00B0	P3
003A	POINT1	006C	POINTERHIGH
E0E0	PORTA	E0E1	PORTB
E0E3	PORTP	091E	PROCESS
055A	PROGDIGIT_OUT	00BC	PS
00B9	PT0	00BB	PT1
00B8	PX0	00BA	PX1
029E	READADDRESS	009C	REN
00D3	RS0	00D4	RS1
0407	RUNDOIT	0362	RUNDOITXX
04B9	RUNMODE	007C	RUNMODEFLAG
0484	RUNSTATE	0412	RUNTOSTOP
035D	S1	035F	S2
0098	SCON	0C23	SELECTWRONG
0DA3	SEMIMODE	009F	SM0
009D	SM2	0081	SP
0030	START	0601	STARTDIGIT1
057B	STEPDIGIT1	0599	STEPDIGIT_OUT
065E	STOPDIGIT_OUT	0BE3	STOPVALUE
0088	TCON	0B63	TESTPAT
008F	TF1	008C	TH0
0099	TI	0051	TIMECOUNT
05E0	TIMEDIGIT_OUT	0052	TIMEVALUE
008B	TL1	0089	TMOD
008E	TR1	0097	TX
09CD	UNITDELAY0	09CF	UNITDELAY1
0243	UPDATEPROGFIELD	0257	UPDATESTARTFIELD
025E	UPDATESTOPFIELD	0438	VALIDRUN
0925	WAITFORXY	092E	WAITING
0294	WRITEADDRESS	07CE	WR_LINE
07B3	X1	07B9	X2
09DE	XY1	09DA	XYDELAY
00E1	ACC1		
09C4	AFTERRUN		
0DE3	AUTOMODE		
005B	BCDCOUNTDOT1		
0060	BCDSTART		

0076 BCDTIME
 005E BINARYCOUNTDOT0
 006B BINARYSTART
 0073 BINARYTIME
 0D63 BLANK
 0119 CALSTEPVALUE
 00EF CC
 0054 COUNTINTTIME0
 0069 COUNTPROGDIGIT
 0068 COUNTSTEPDIGIT
 00D7 CY
 08BE DCLR
 02D2 DD2
 0064 DELAY100MS
 0055 DISPLAYDOT0
 0058 DISPLAYDOT3
 0079 DISPLAYPROG1
 007C DISPLAYSTEP0
 0063 DISPLAYSTOP1
 007D DISPLAYTIME2
 01AA DISPSTEPFIELD
 08C8 DLOOP1
 0318 DOEDIT1
 0323 DOIT1
 0E59 DONEDEE
 0082 DPL
 090B EACHPROGRAM
 007D EDITMODEFLAG
 03E7 EDITTOSTEP1
 04FA ENTEREDIT
 050C ENTEREDITTIME
 04F4 ENTERFILLSTEP
 051E ENTERPROGVALUE
 051B ENTERRUNSTOP
 0622 ENTERSTOPVALUE
 00AC ES
 00A8 EX0
 0366 FILLDOIT
 047E FILLSTATE
 02A8 FINDADDRESS
 026C GETTIMEVALUE
 0307 GONEXT1
 0449 GONEXT4
 0326 GOOUT1
 0463 GOOUT4
 06E8 GOTOMENU
 0680 GOTOSTEP
 0089 IE0
 0440 INFODOIT
 07DD INITLCD
 007A INSTEPFIELD
 00B8 IPC
 008A IT1
 083B LCDLDS
 0870 LCDWD1



09E3 LOGO
02BA LOOKUP3
0898 LOOP11
047B MENUSTATE
0B23 MESSRUN
0E2F MIL1
075B MODEEDIT
071C MODEMENU
0E52 MSHDOWN
07A9 NEXTBIT
04D1 NOKEY
00D2 OV
0090 P1
0087 PCON
006D POINTERLOW
E0E2 PORTC
053C PROGDIGIT1
00D0 PSW
0265 PUTTIMEVALUE
009A RB8
0098 RI
08FF RUN
0911 RUNLOOP1
041A RUNPROGRAM
0096 RX
0099 SBUF
0095 SEMI
009E SM1
00DA SS1
061F STARTDIGIT_OUT
0640 STOPDIGIT1
009B TB8
008D TF0
008D TH1
05BA TIMEDIGIT1
008A TLO
008C TRO
09CB UNITDELAY
09D1 UNITDELAY2
024D UPDATESTEPFIELD
039D VALIDTIME
080E WRIBYTE
07C2 WR_PAGE
09DC XY

ARITHMETIC OPERATIONS

Mnemonic	Description	Byte	Cyc
ADD A,Rn	Add register to Accumulator	1	1
ADD A,direct	Add direct byte to Accumulator	2	1
ADD A,@Ri	Add indirect RAM to Accumulator	1	1
ADD A,#data	Add immediate data to Accumulator	2	1
ADDC A,Rn	Add register to Accumulator with Carry	1	1
ADDC A,direct	Add direct byte to A with Carry flag	2	1
ADDC A,@Ri	Add indirect RAM to A with Carry flag	1	1
ADDC A,#data	Add immediate data to A with Carry flag	2	1
SUBB A,Rn	Subtract register from A with Borrow	1	1
SUBB A,direct	Subtract direct byte from A with Borrow	2	1
SUBB A,@Ri	Subtract indirect RAM from A w/ Borrow	1	1
SUBB A,#data	Subtract immed. data from A w/ Borrow	2	1
INC A	Increment Accumulator	1	1
INC Rn	Increment register	1	1
INC direct	Increment direct byte	2	1
INC @Ri	Increment indirect RAM	1	1
DEC A	Decrement Accumulator	1	1
DEC Rn	Decrement register	1	1
DEC direct	Decrement direct byte	2	1
DEC @Ri	Decrement indirect RAM	1	1
INC DPTR	Increment Data Pointer	1	2
MUL AB	Multiply A & B	1	4
DIV AB	Divide A by B	1	4
DA A	Decimal Adjust Accumulator	1	1

LOGICAL OPERATIONS

Mnemonic	Description	Byte	Cyc
ANL A,Rn	AND register to Accumulator	1	1
ANL A,direct	AND direct byte to Accumulator	2	1
ANL A,@Ri	AND indirect RAM to Accumulator	1	1
ANL A,#data	AND immediate data to Accumulator	2	1
ANL direct,A	AND Accumulator to direct byte	2	1
ANL @Ri,A	AND immediate data to direct byte	2	1
ORL A,Rn	OR register to Accumulator	1	1
ORL A,direct	OR direct byte to Accumulator	2	1
ORL A,@Ri	OR indirect RAM to Accumulator	1	1
ORL A,#data	OR immediate data to Accumulator	2	1
ORL direct,A	OR Accumulator to direct byte	2	1
ORL @Ri,A	OR immediate data to direct byte	2	1
XRL A,Rn	Exclusive-OR register to Accumulator	1	1
XRL A,direct	Exclusive-OR direct byte to Accumulator	2	1
XRL A,@Ri	Exclusive-OR indirect RAM to A	1	1
XRL A,#data	Exclusive-OR immediate data to A	2	1
XRL direct,A	Exclusive-OR Accumulator to direct byte	2	1
XRL @Ri,A	Exclusive-OR immediate data to direct	2	1
CLR A	Clear Accumulator	1	1
CPL A	Complement Accumulator	1	1
RL A	Rotate Accumulator Left	1	1
RLC A	Rotate A Left through the Carry flag	1	1
RR A	Rotate Accumulator Right	1	1
RRC A	Rotate A Right through Carry flag	1	1
SWAP A	Swap nibbles within the Accumulator	1	1

DATA TRANSFER

Mnemonic	Description	Byte	Cyc
MOV A,Rn	Move register to Accumulator	1	1
MOV A,direct	Move direct byte to Accumulator	2	1
MOV A,@Ri	Move indirect RAM to Accumulator	1	1
MOV A,#data	Move immediate data to Accumulator	2	1
MOV Rn,A	Move Accumulator to register	1	1
MOV Rn,direct	Move direct byte to register	2	1
MOV Rn,#data	Move immediate data to register	2	1
MOV direct,Rn	Move register to direct byte	2	1
MOV direct,direct	Move direct byte to direct	2	1
MOV direct,@Ri	Move indirect RAM to direct byte	2	1
MOV direct,#data	Move immediate data to direct byte	2	1
MOV @Ri,A	Move Accumulator to indirect RAM	1	1
MOV @Ri,direct	Move direct byte to indirect RAM	2	1
MOV @Ri,#data	Move immediate data to indirect RAM	2	1
MOV DPTR,#data16	Load Data Pointer with a 16-bit constant	3	2

DATA TRANSFER (cont.)

Mnemonic	Description	Byte	Cyc
MOVC A,@A+DPTR	Move Code byte relative to DPTR to A	1	2
MOVC A,@A+PC	Move Code byte relative to PC to A	1	2
MOVX A,@Ri	Move External RAM (8-bit addr) to A	1	2
MOVX A,@DPTR	Move External RAM (16-bit addr) to A	1	2
MOVX @Ri,A	Move A to External RAM (8-bit addr)	1	2
MOVX @DPTR,A	Move A to External RAM (16-bit addr)	1	2
PUSH direct	Push direct byte onto stack	2	2
POP direct	Pop direct byte from stack	2	2
XCH A,Rn	Exchange register with Accumulator	1	1
XCH A,direct	Exchange direct byte with Accumulator	2	1
XCH A,@Ri	Exchange indirect RAM with A	1	1
XCHD A,@Ri	Exchange low-order Digit ind. RAM w/A	1	1

BOOLEAN VARIABLE MANIPULATION

Mnemonic	Description	Byte	Cyc
CLR C	Clear Carry flag	1	1
CLR bit	Clear direct bit	2	1
SETB C	Set Carry flag	1	1
SETB bit	Set direct bit	2	1
CPL C	Complement Carry flag	1	1
CPL bit	Complement direct bit	2	1
ANL C,bit	AND direct bit to Carry flag	2	2
ORL C,bit	AND complement of direct bit to Carry	2	2
ORL C,bit	OR direct bit to Carry flag	2	2
MOV C,bit	OR complement of direct bit to Carry	2	2
MOV bit,C	Move direct bit to Carry flag	2	1
MOV bit,C	Move Carry flag to direct bit	2	2

PROGRAM AND MACHINE CONTROL

Mnemonic	Description	Byte	Cyc
ACALL addr11	Absolute Subroutine Call	3	2
LCALL addr16	Long Subroutine Call	3	2
RET	Return from subroutine	1	2
RETI	Return from interrupt	1	2
AJMP addr11	Absolute Jump	2	2
LJMP addr16	Long Jump	3	2
SJMP rel	Short Jump (relative addr)	2	2
JZ @A+DPTR	Jump indirect relative to the DPTR	1	2
JZ rel	Jump if Accumulator is Zero	2	2
JNC rel	Jump if Accumulator is Not Zero	2	2
JNC rel	Jump if Carry flag is set	2	2
JB bit,rel	Jump if No Carry flag	2	2
JNB bit,rel	Jump if direct Bit set	3	2
JBC bit,rel	Jump if direct Bit Not set	3	2
CJNE A,direct,rel	Jump if direct Bit is set & Clear bit	3	2
CJNE A,#data,rel	Compare direct to A & Jump if Not Equal	3	2
CJNE Rn,#data,rel	Comp. immed. to reg. & Jump if Not Equal	3	2
CJNE @Ri,#data,rel	Comp. immed. to ind. & Jump if Not Equal	3	2
DJNZ Rn,rel	Decrement register & Jump if Not Zero	2	2
DJNZ direct,rel	Decrement direct & Jump if Not Zero	3	2
NOP	No operation	1	1

Notes on data addressing modes:

- Rn - Working register R0-R7
- direct - 128 internal RAM locations, any I/O port, control or status register
- @Ri - indirect internal RAM location addressed by register R0 or R1
- #data8 - 8-bit constant included in instruction
- #data16 - 16-bit constant included as bytes 2 & 3 of instruction
- bit - 128 software flags, any I/O pin, control or status bit

Notes on program addressing modes:

- addr16 - Destination address for LCALL & LJMP may be anywhere within the 64-Kilobyte program memory address space.
- addr11 - Destination address for ACALL & AJMP will be within the same 2-Kilobyte page of program memory as the first byte of the following instruction.
- rel - SJMP and all conditional jumps include an 8-bit offset byte. Range is 127/-128 bytes relative to first byte of the following instruction.

All mnemonics copyrighted © Intel Corporation 1979



MM54C922/MM74C922 16-Key Encoder MM54C923/MM74C923 20-Key Encoder

general description

These CMOS key encoders provide all the necessary logic to fully encode an array of SPST switches. The keyboard scan can be implemented by either an external clock or external capacitor. These encoders also have on-chip pull-up devices which permit switches with up to 50 k Ω on resistance to be used. No diodes in the switch array are needed to eliminate ghost switches. The internal debounce circuit needs only a single external capacitor and can be defeated by omitting the capacitor. A Data Available output goes to a high level when a valid keyboard entry has been made. The Data Available output returns to a low level when the entered key is released, even if another key is depressed. The Data Available will return high to indicate acceptance of the new key after a normal debounce period; this two key roll over is provided between any two switches.

An internal register remembers the last key pressed even after the key is released. The TRI-STATE[®] outputs

provide for easy expansion and bus operation and are LPTTL compatible.

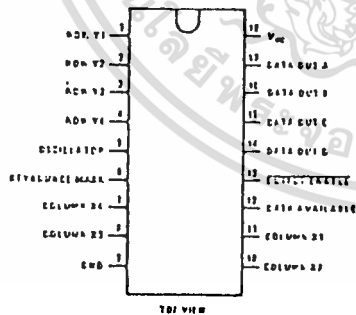
features

- 50 k Ω maximum switch on resistance
- On or off chip clock
- On chip row pull-up devices
- 2 key roll-over
- Keybounce elimination with single capacitor
- Last key register at outputs
- TRI-STATE outputs LPTTL compatible
- Wide supply range
- Low power consumption

3V to 15V

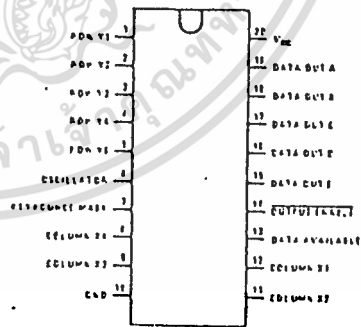
connection diagrams

Dual-In-Line Package



Order Number MM54C922N
or MM74C922N
See Package 20

Dual-In-Line Package



Order Number MM54C923N
or MM74C923N
See Package 20A

MM54C922/MM74C922, MM54C923/MM74C923

absolute maximum ratings

Voltage at Any Pin
Operating Temperature Range
MM54C922, MM54C923
MM74C922, MM74C923
Storage Temperature Range

 $V_{CC} = 0.3V \text{ to } V_{CC} + 0.3V$

55°C to +125°C
-40°C to +85°C
-65°C to +150°C

Package Dissipation
Operating V_{CC} Range
 V_{CC}
Lead Temperature (Soldering, 10 seconds)

500 mW
3V to 15V
18V
300°C

dc electrical characteristics Min/max limits apply across temperature range unless otherwise noted

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
CMOS TO CMOS					
V_{T+} Positive Going Threshold Voltage at Osc and KBM Inputs	$V_{CC} = 5V, I_{IN} \leq 0.7 \text{ mA}$	3	3.6	4.3	V
	$V_{CC} = 10V, I_{IN} \geq 1.4 \text{ mA}$	6	6.6	8.6	V
	$V_{CC} = 15V, I_{IN} \geq 2.1 \text{ mA}$	9	10	12.5	V
V_{T-} Negative Going Threshold Voltage at Osc and KBM Inputs	$V_{CC} = 5V, I_{IN} \geq 0.7 \text{ mA}$	0.7	1.4	2	V
	$V_{CC} = 10V, I_{IN} \geq 1.4 \text{ mA}$	1.4	3.2	4	V
	$V_{CC} = 15V, I_{IN} \geq 2.1 \text{ mA}$	2.1	5	6	V
$V_{IN(1)}$ Logical "1" Input Voltage, Except Osc and KBM Inputs	$V_{CC} = 5V$	2.5	4.5		V
	$V_{CC} = 10V$	5	9		V
	$V_{CC} = 15V$	12.5	13.5		V
$V_{IN(0)}$ Logical "0" Input Voltage, Except Osc and KBM Inputs	$V_{CC} = 5V$		0.5	1.5	V
	$V_{CC} = 10V$		1	2	V
	$V_{CC} = 15V$		1.5	2.5	V
I_{p} Row Pull Up Current at Y1, Y2, Y3, Y4 and Y5 Inputs	$V_{CC} = 5V, V_{IN} = 0.1 V_{CC}$	-2	-5		μA
	$V_{CC} = 10V$	-10	-20		μA
	$V_{CC} = 15V$	-22	-45		μA
$V_{OUT(1)}$ Logical "1" Output Voltage	$V_{CC} = 5V, I_O = -10 \mu A$	4.5			V
	$V_{CC} = 10V, I_O = -10 \mu A$	9			V
	$V_{CC} = 15V, I_O = -10 \mu A$	13.5			V
$V_{OUT(0)}$ Logical "0" Output Voltage	$V_{CC} = 5V, I_O = 10 \mu A$			0.5	V
	$V_{CC} = 10V, I_O = 10 \mu A$			1	V
	$V_{CC} = 15V, I_O = 10 \mu A$			1.5	V
R_{on} Column "ON" Resistance at X1, X2, X3 and X4 Outputs	$V_{CC} = 5V, V_O = 0.5V$		500	1400	Ω
	$V_{CC} = 10V, V_O = 1V$		300	700	Ω
	$V_{CC} = 15V, V_O = 1.5V$		200	500	Ω
I_{CC} Supply Current	$V_{CC} = 5V, \text{Osc at } 0V$		0.55	1.1	mA
	$V_{CC} = 10V$		1.1	1.9	mA
	$V_{CC} = 15V$		1.7	2.6	mA
$I_{IH(1)}$ Logical "1" Input Current at Output Enable	$V_{CC} = 15V, V_{IN} = 15V$		0.005	1.0	μA
$I_{IN(0)}$ Logical "0" Input Current at Output Enable	$V_{CC} = 15V, V_{IN} = 0V$	1.0	-0.005		μA
CMOS/LVTTL INTERFACE					
$V_{IN(1)}$ Logical "1" Input Voltage, Except Osc and KBM Inputs	54C, $V_{CC} = 4.5V$	$V_{CC} \cdot 1.5$			V
	74C, $V_{CC} = 4.75V$	$V_{CC} \cdot 1.5$			V
$V_{IN(0)}$ Logical "0" Input Voltage, Except Osc and KBM Inputs	54C, $V_{CC} = 4.5V$			0.8	V
	74C, $V_{CC} = 4.75V$			0.8	V
$V_{OUT(1)}$ Logical "1" Output Voltage	54C, $V_{CC} = 4.5V, I_O = -360 \mu A$	2.4			V
	74C, $V_{CC} = 4.75V, I_O = -360 \mu A$	2.4			V
$V_{OUT(0)}$ Logical "0" Output Voltage	54C, $V_{CC} = 4.5V, I_O = -360 \mu A$			0.4	V
	74C, $V_{CC} = 4.75V, I_O = -360 \mu A$			0.4	V

dc electrical characteristics (con't)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
OUTPUT DRIVE (See 54C/74C Family Characteristics Data Sheet)					
I_{SOURCE} Output Source Current (P-Channel)	$V_{CC} = 5V, V_{OUT} = 0V,$ $T_A = 25^\circ C$	-1.75	-3.3		mA
I_{SOURCE} Output Source Current (P-Channel)	$V_{CC} = 10V, V_{OUT} = 0V,$ $T_A = 25^\circ C$	-8	-15		mA
I_{SINK} Output Sink Current (N-Channel)	$V_{CC} = 5V, V_{OUT} = V_{CC},$ $T_A = 25^\circ C$	1.75	3.6		mA
I_{SINK} Output Sink Current (N-Channel)	$V_{CC} = 10V, V_{OUT} = V_{CC},$ $T_A = 25^\circ C$	8	16		mA

ac electrical characteristics $T_A = 25^\circ C$

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
t_{pd0}, t_{pd1} Propagation Delay Time to Logical "0" or Logical "1" from D.A.	$C_L = 50 \text{ pF},$ (Figure 1) $V_{CC} = 5V$ $V_{CC} = 10V$ $V_{CC} = 15V$		60 25 25	150 80 60	ns ns ns
t_{0H}, t_{1H} Propagation Delay Time from Logical "0" or Logical "1" into High Impedance State	$R_L = 10k, C_L = 5 \text{ pF},$ (Figure 2) $V_{CC} = 5V, R_L = 10k$ $V_{CC} = 10V, C_L = 10 \text{ pF}$ $V_{CC} = 15V$		60 65 50	200 150 110	ns ns ns
t_{H0}, t_{H1} Propagation Delay Time from High Impedance State to a Logical "0" or Logical "1"	$R_L = 10k, C_L = 50 \text{ pF},$ (Figure 2) $V_{CC} = 5V, R_L = 10k$ $V_{CC} = 10V, C_L = 50 \text{ pF}$ $V_{CC} = 15V$		100 55 40	250 125 90	ns ns ns
C_{IN} Input Capacitance	Any Input, (Note 2)		5	7.5	pF
C_{OUT} TRI-STATE Output Capacitance	Any Output, (Note 2)		10		pF

Note 1: "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. Except for "Operating Temperature Range" they are not meant to imply that the devices should be operated at these limits. The table of "Electrical Characteristics" provides conditions for actual device operation.

Note 2: Capacitance is guaranteed by periodic testing.

switching time waveforms

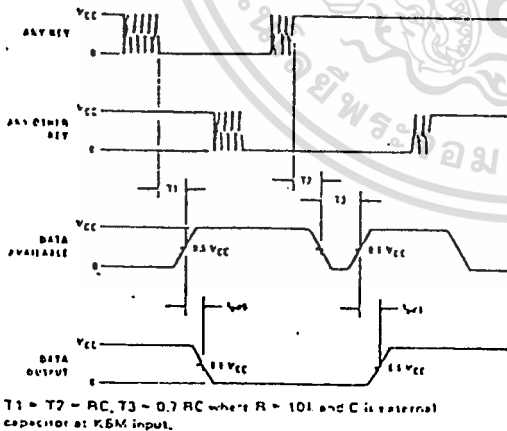


FIGURE 1

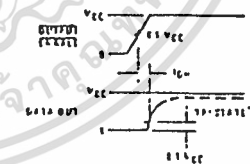
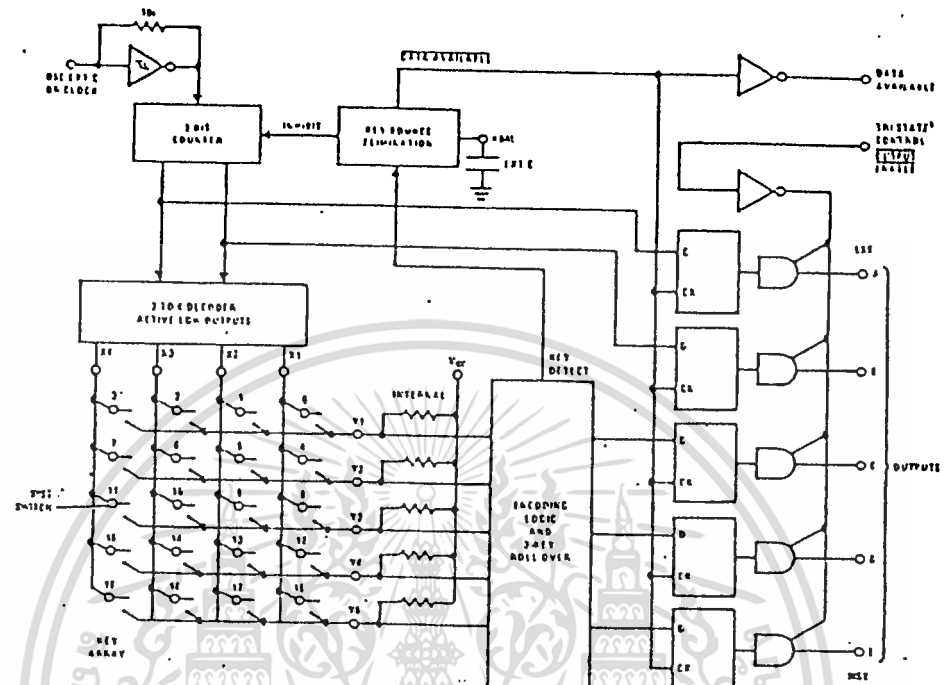


FIGURE 2

block diagram

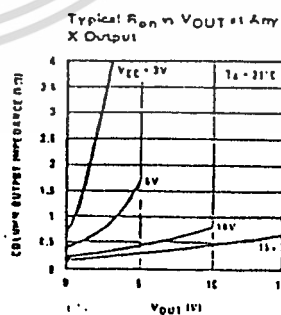
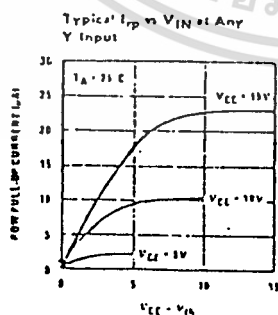


truth table

S-111CH POLY(100%)	1		2		3		4		5		6		7		8		9		10		11		12		13		14		15		16		17		18		19		20																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
	V1,F1	V1,S2	V1,R3	V1,B4	V1,G5	V1,Y6	V1,K7	V1,D8	V1,X9	V1,Z10	V1,M11	V1,J12	V1,Q13	V1,W14	V1,E15	V1,H16	V1,N17	V1,P18	V1,A19	V1,S20	V1,T21	V1,I22	V1,O23	V1,U24	V1,L25	V1,C26	V1,F27	V1,V28	V1,B29	V1,N30	V1,D31	V1,X32	V1,Z33	V1,M34	V1,J35	V1,Q36	V1,W37	V1,E38	V1,H39	V1,N40	V1,P41	V1,A42	V1,S43	V1,T44	V1,I45	V1,O46	V1,U47	V1,L48	V1,C49	V1,F50	V1,V51	V1,B52	V1,N53	V1,D54	V1,X55	V1,Z56	V1,M57	V1,J58	V1,Q59	V1,W60	V1,E61	V1,H62	V1,N63	V1,P64	V1,A65	V1,S66	V1,T67	V1,I68	V1,O69	V1,U70	V1,L71	V1,C72	V1,F73	V1,V74	V1,B75	V1,N76	V1,D77	V1,X78	V1,Z79	V1,M80	V1,J81	V1,Q82	V1,W83	V1,E84	V1,H85	V1,N86	V1,P87	V1,A88	V1,S89	V1,T90	V1,I91	V1,O92	V1,U93	V1,L94	V1,C95	V1,F96	V1,V97	V1,B98	V1,N99	V1,D100	V1,X101	V1,Z102	V1,M103	V1,J104	V1,Q105	V1,W106	V1,E107	V1,H108	V1,N109	V1,P110	V1,A111	V1,S112	V1,T113	V1,I114	V1,O115	V1,U116	V1,L117	V1,C118	V1,F119	V1,V120	V1,B121	V1,N122	V1,D123	V1,X124	V1,Z125	V1,M126	V1,J127	V1,Q128	V1,W129	V1,E130	V1,H131	V1,N132	V1,P133	V1,A134	V1,S135	V1,T136	V1,I137	V1,O138	V1,U139	V1,L140	V1,C141	V1,F142	V1,V143	V1,B144	V1,N145	V1,D146	V1,X147	V1,Z148	V1,M149	V1,J150	V1,Q151	V1,W152	V1,E153	V1,H154	V1,N155	V1,P156	V1,A157	V1,S158	V1,T159	V1,I160	V1,O161	V1,U162	V1,L163	V1,C164	V1,F165	V1,V166	V1,B167	V1,N168	V1,D169	V1,X170	V1,Z171	V1,M172	V1,J173	V1,Q174	V1,W175	V1,E176	V1,H177	V1,N178	V1,P179	V1,A180	V1,S181	V1,T182	V1,I183	V1,O184	V1,U185	V1,L186	V1,C187	V1,F188	V1,V189	V1,B190	V1,N191	V1,D192	V1,X193	V1,Z194	V1,M195	V1,J196	V1,Q197	V1,W198	V1,E199	V1,H200	V1,N201	V1,P202	V1,A203	V1,S204	V1,T205	V1,I206	V1,O207	V1,U208	V1,L209	V1,C210	V1,F211	V1,V212	V1,B213	V1,N214	V1,D215	V1,X216	V1,Z217	V1,M218	V1,J219	V1,Q220	V1,W221	V1,E222	V1,H223	V1,N224	V1,P225	V1,A226	V1,S227	V1,T228	V1,I229	V1,O230	V1,U231	V1,L232	V1,C233	V1,F234	V1,V235	V1,B236	V1,N237	V1,D238	V1,X239	V1,Z240	V1,M241	V1,J242	V1,Q243	V1,W244	V1,E245	V1,H246	V1,N247	V1,P248	V1,A249	V1,S250	V1,T251	V1,I252	V1,O253	V1,U254	V1,L255	V1,C256	V1,F257	V1,V258	V1,B259	V1,N260	V1,D261	V1,X262	V1,Z263	V1,M264	V1,J265	V1,Q266	V1,W267	V1,E268	V1,H269	V1,N270	V1,P271	V1,A272	V1,S273	V1,T274	V1,I275	V1,O276	V1,U277	V1,L278	V1,C279	V1,F280	V1,V281	V1,B282	V1,N283	V1,D284	V1,X285	V1,Z286	V1,M287	V1,J288	V1,Q289	V1,W290	V1,E291	V1,H292	V1,N293	V1,P294	V1,A295	V1,S296	V1,T297	V1,I298	V1,O299	V1,U300	V1,L301	V1,C302	V1,F303	V1,V304	V1,B305	V1,N306	V1,D307	V1,X308	V1,Z309	V1,M310	V1,J311	V1,Q312	V1,W313	V1,E314	V1,H315	V1,N316	V1,P317	V1,A318	V1,S319	V1,T320	V1,I321	V1,O322	V1,U323	V1,L324	V1,C325	V1,F326	V1,V327	V1,B328	V1,N329	V1,D330	V1,X331	V1,Z332	V1,M333	V1,J334	V1,Q335	V1,W336	V1,E337	V1,H338	V1,N339	V1,P340	V1,A341	V1,S342	V1,T343	V1,I344	V1,O345	V1,U346	V1,L347	V1,C348	V1,F349	V1,V350	V1,B351	V1,N352	V1,D353	V1,X354	V1,Z355	V1,M356	V1,J357	V1,Q358	V1,W359	V1,E360	V1,H361	V1,N362	V1,P363	V1,A364	V1,S365	V1,T366	V1,I367	V1,O368	V1,U369	V1,L370	V1,C371	V1,F372	V1,V373	V1,B374	V1,N375	V1,D376	V1,X377	V1,Z378	V1,M379	V1,J380	V1,Q381	V1,W382	V1,E383	V1,H384	V1,N385	V1,P386	V1,A387	V1,S388	V1,T389	V1,I390	V1,O391	V1,U392	V1,L393	V1,C394	V1,F395	V1,V396	V1,B397	V1,N398	V1,D399	V1,X400	V1,Z401	V1,M402	V1,J403	V1,Q404	V1,W405	V1,E406	V1,H407	V1,N408	V1,P409	V1,A410	V1,S411	V1,T412	V1,I413	V1,O414	V1,U415	V1,L416	V1,C417	V1,F418	V1,V419	V1,B420	V1,N421	V1,D422	V1,X423	V1,Z424	V1,M425	V1,J426	V1,Q427	V1,W428	V1,E429	V1,H430	V1,N431	V1,P432	V1,A433	V1,S434	V1,T435	V1,I436	V1,O437	V1,U438	V1,L439	V1,C440	V1,F441	V1,V442	V1,B443	V1,N444	V1,D445	V1,X446	V1,Z447	V1,M448	V1,J449	V1,Q450	V1,W451	V1,E452	V1,H453	V1,N454	V1,P455	V1,A456	V1,S457	V1,T458	V1,I459	V1,O460	V1,U461	V1,L462	V1,C463	V1,F464	V1,V465	V1,B466	V1,N467	V1,D468	V1,X469	V1,Z470	V1,M471	V1,J472	V1,Q473	V1,W474	V1,E475	V1,H476	V1,N477	V1,P478	V1,A479	V1,S480	V1,T481	V1,I482	V1,O483	V1,U484	V1,L485	V1,C486	V1,F487	V1,V488	V1,B489	V1,N490	V1,D491	V1,X492	V1,Z493	V1,M494	V1,J495	V1,Q496	V1,W497	V1,E498	V1,H499	V1,N500	V1,P501	V1,A502	V1,S503	V1,T504	V1,I505	V1,O506	V1,U507	V1,L508	V1,C509	V1,F510	V1,V511	V1,B512	V1,N513	V1,D514	V1,X515	V1,Z516	V1,M517	V1,J518	V1,Q519	V1,W520	V1,E521	V1,H522	V1,N523	V1,P524	V1,A525	V1,S526	V1,T527	V1,I528	V1,O529	V1,U530	V1,L531	V1,C532	V1,F533	V1,V534	V1,B535	V1,N536	V1,D537	V1,X538	V1,Z539	V1,M540	V1,J541	V1,Q542	V1,W543	V1,E544	V1,H545	V1,N546	V1,P547	V1,A548	V1,S549	V1,T550	V1,I551	V1,O552	V1,U553	V1,L554	V1,C555	V1,F556	V1,V557	V1,B558	V1,N559	V1,D560	V1,X561	V1,Z562	V1,M563	V1,J564	V1,Q565	V1,W566	V1,E567	V1,H568	V1,N569	V1,P570	V1,A571	V1,S572	V1,T573	V1,I574	V1,O575	V1,U576	V1,L577	V1,C578	V1,F579	V1,V580	V1,B581	V1,N582	V1,D583	V1,X584	V1,Z585	V1,M586	V1,J587	V1,Q588	V1,W589	V1,E590	V1,H591	V1,N592	V1,P593	V1,A594	V1,S595	V1,T596	V1,I597	V1,O598	V1,U599	V1,L600	V1,C601	V1,F602	V1,V603	V1,B604	V1,N605	V1,D606	V1,X607	V1,Z608	V1,M609	V1,J610	V1,Q611	V1,W612	V1,E613	V1,H614	V1,N615	V1,P616	V1,A617	V1,S618	V1,T619	V1,I620	V1,O621	V1,U622	V1,L623	V1,C624	V1,F625	V1,V626	V1,B627	V1,N628	V1,D629	V1,X630	V1,Z631	V1,M632	V1,J633	V1,Q634	V1,W635	V1,E636	V1,H637	V1,N638	V1,P639	V1,A640	V1,S641	V1,T642	V1,I643	V1,O644	V1,U645	V1,L646	V1,C647	V1,F648	V1,V649	V1,B650	V1,N651	V1,D652	V1,X653	V1,Z654	V1,M655	V1,J656	V1,Q657	V1,W658	V1,E659	V1,H660	V1,N661	V1,P662	V1,A663	V1,S664	V1,T665	V1,I666	V1,O667	V1,U668	V1,L669	V1,C670	V1,F671	V1,V672	V1,B673	V1,N674	V1,D675	V1,X676	V1,Z677	V1,M678	V1,J679	V1,Q680	V1,W681	V1,E682	V1,H683	V1,N684	V1,P685	V1,A686	V1,S687	V1,T688	V1,I689	V1,O690	V1,U691	V1,L692	V1,C693	V1,F694	V1,V695	V1,B696	V1,N697	V1,D698	V1,X699	V1,Z700	V1,M701	V1,J702	V1,Q703	V1,W704	V1,E705	V1,H706	V1,N707	V1,P708	V1,A709	V1,S710	V1,T711	V1,I712	V1,O713	V1,U714	V1,L715	V1,C716	V1,F717	V1,V718	V1,B719	V1,N720	V1,D721	V1,X722	V1,Z723	V1,M724	V1,J725	V1,Q726	V1,W727	V1,E728	V1,H729	V1,N730	V1,P731	V1,A732	V1,S733	V1,T734	V1,I735	V1,O736	V1,U737	V1,L738	V1,C739	V1,F740	V1,V741	V1,B742	V1,N743	V1,D744	V1,X745	V1,Z746	V1,M747	V1,J748	V1,Q749	V1,W750	V1,E751	V1,H752	V1,N753	V1,P754	V1,A755	V1,S756	V1,T757	V1,I758	V1,O759	V1,U760	V1,L761	V1,C762	V1,F763	V1,V764	V1,B765	V1,N766	V1,D767	V1,X768	V1,Z769	V1,M770	V1,J771	V1,Q772	V1,W773	V1,E774	V1,H775	V1,N776	V1,P777	V1,A778	V1,S779	V1,T780	V1,I781	V1,O782	V1,U783	V1,L784	V1,C785	V1,F786	V1,V787	V1,B788	V1,N789	V1,D790	V1,X791	V1,Z792	V1,M793	V1,J794	V1,Q795	V1,W796	V1,E797	V1,H798	V1,N799	V1,P800	V1,A801	V1,S802	V1,T803	V1,I804	V1,O805	V1,U806	V1,L807	V1,C808	V1,F809	V1,V810	V1,B811	V1,N812	V1,D813	V1,X814	V1,Z815	V1,M816	V1,J817	V1,Q818	V1,W819	V1,E820	V1,H821	V1,N822	V1,P823	V1,A824	V1,S825	V1,T826	V1,I827	V1,O828	V1,U829	V1,L830	V1,C831	V1,F832	V1,V833	V1,B834	V1,N835	V1,D836	V1,X837	V1,Z838	V1,M839	V1,J840	V1,Q841	V1,W842	V1,E843	V1,H844	V1,N845	V1,P846	V1,A847	V1,S848	V1,T849	V1,I850	V1,O851	V1,U852	V1,L853	V1,C854	V1,F855	V1,V856	V1,B857	V1,N858	V1,D859	V1,X860	V1,Z861	V1,M862	V1,J863	V1,Q864	V1,W865	V1,E866	V1,H867	V1,N868	V1,P869	V1,A870	V1,S871	V1,T872	V1,I873	V1,O874	V1,U875	V1,L876	V1,C877	V1,F878	V1,V879	V1,B880	V1,N881	V1,D882	V1,X883	V1,Z884	V1,M885	V1,J886	V1,Q887	V1,W888	V1,E889	V1,H890	V1,N891	V1,P892	V1,A893	V1,S894	V1,T895	V1,I896	V1,O897	V1,U898	V1,L899	V1,C900	V1,F901	V1,V902	V1,B903	V1,N904	V1,D905	V1,X906	V1,Z907	V1,M908	V1,J909	V1,Q910	V1,W911	V1,E912	V1,H913	V1,N914	V1,P915	V1,A916	V1,S917	V1,T918	V1,I919	V1,O920	V1,U921	V1,L922	V1,C923	V1,F924	V1,V925	V1,B926	V1,N927	V1,D928	V1,X929	V1,Z930	V1,M931	V1,J932	V1,Q933	V1,W934	V1,E935	V1,H936	V1,N937	V1,P938	V1,A939	V1,S940	V1,T941	V1,I942	V1,O943	V1,U944	V1,L945	V1,C946	V1,F947	V1,V948	V1,B949	V1,N950	V1,D951	V1,X952	V1,Z953	V1,M954	V1,J955	V1,Q956	V1,W957	V1,E958	V1,H959	V1,N960	V1,P961	V1,A962	V1,S963	V1,T964	V1,I965	V1,O966	V1,U967	V1,L968	V1,C969	V1,F970	V1,V971	V1,B972	V1,N973	V1,D974	V1,X975	V1,Z976	V1,M977	V1,J978	V1,Q979	V1,W980	V1,E981	V1,H982	V1,N983	V1,P984	V1,A985	V1,S986	V1,T987	V1,I988	V1,O989	V1,U990	V1,L991	V1,C992	V1,F993	V1,V994	V1,B995	V1,N996	V1,D997	V1,X998	V1,Z999	V1,M1000	V1,J1001	V1,Q1002	V1,W1003	V1,E1004	V1,H1005	V1,N1006	V1,P1007	V1,A1008	V1,S1009	V1,T1010	V1,I1011	V1,O1012	V1,U1013	V1,L1014	V1,C1015	V1,F1016	V1,V1017	V1,B1018	V1,N1019	V1,D1020	V1,X1021	V1,Z1022	V1,M1023	V1,J1024	V1,Q1025	V1,W1026	V1,E1027	V1,H1028	V1,N1029	V1,P1030	V1,A1031	V1,S1032	V1,T1033	V1,I1034	V1,O1035	V1,U1036	V1,L1037	V1,C1038	V1,F1039	V1,V1040	V1,B1041	V1,N1042	V1,D1043	V1,X1044	V1,Z1045	V1,M1046	V1,J1047	V1,Q1048	V1,W1049	V1,E1050	V1,H1051	V1,N1052	V1,P1053	V1,A1054	V1,S1055	V1,T1056	V1,I1057	V1,O1058	V1,U1059	V1,L1060	V1,C1061	V1,F1062	V1,V1063	V1,B1064	V1,N1065	V1,D1066	V1,X1067	V1,Z1068	V1,M1069	V1,J1070	V1,Q1071	V1,W1072	V1,E1073	V1,H1074	V1,N1075	V1,P1076	V1,A1077	V1,S1078	V1,T1079	V1,I1080	V1,O1081	V1,U1082	V1,L1083	V1,C1084	V1,F1085	V1,V1086	V1,B1087	V1,N1088	V1,D1089	V1,X1090	V1,Z1091	V1,M1092	V1,J1093	V1,Q1094	V1,W1095	V1,E1096	V1,H1097	V1,N1098	V1,P1099	V1,A1100	V1,S1101	V1,T1102	V1,I1103	V1,O1104	V1,U1105	V1,L1106	V1,C1107	V1,F1108	V1,V1109	V1,B1110	V1,N1111	V1,D1112	V1,X1113	V1,Z1114	V1,M1115	V1,J1116	V1,Q1117	V1,W1118	V1,E1119	V1,H1120	V1,N1121	V1,P1122	V1,A1123	V1,S1124	V1,T1125	V1,I1126	V1,O1127	V1,U1128	V1,L1129	V1,C1130	V1,F1131	V1,V1132	V1,B1133	V1,N1134	V1,D1135	V1,X1136	V1,Z1137	V1,M1138	V1,J1139	V1,Q1140	V1,W1141	V1,E1142	V1,H1143	V1,N1144	V1,P1145	V1,A1146	V1,S1147	V1,T1148	V1,I1149	V1,O1150	V1,U1151	V1,L1152	V1,C1153	V1,F1154	V1,V1155	V1,B1156	V1,N1157	V1,D1158	V1,X1159	V1,Z1160	V1,M1161	V1,J1162	V1,Q1163	V1,W1164	V1,E1165	V1,H1166	V1,N1167	V1,P1168	V1,A1169	V1,S1170	V1,T1171	V1,I1172	V1,O1173	V1,U1174	V1,L1175	V1,C1176	V1,F1177	V1,V1178	V1,B1179	V1,N1180	V1,D1181	V1,X1182	V1,Z1183	V1,M1184	V1,J1185	V1,Q1186	V1,W1187	V1,E1188	V1,H1189	V1,N1190	V1,P1191	V1,A1192	V1,S1193	V1,T1194	V1,I1195	V1,O1196	V1,U1197	V1,L1198	V1,C1199	V1,F1200	V1,V1201	V1,B1202	V1,N1203	V1,D1204	V1,X1205	V1,Z1206

* Omit for M.M.54C922/AM.74C922

typical performance characteristics



กิตติกรรมประกาศ

ในการทำปริญญานิพนธ์ ต้องขอขอบคุณ อาจารย์ สุพรรณ กุลพาณิชย์ และ พี่ สุรเชษฐ์ ผู้ให้คำปรึกษา และแนวความคิดทำให้ปริญญานิพนธ์นี้สำเร็จลงอย่างสมบูรณ์



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดลอกเผยแพร่ และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

1. ยืน ภู่วรวรรณ , ทฤษฎี และการประยุกต์ไมโครโปรเซสเซอร์ Z 80, กรุงเทพฯ บริษัท ซีเอ็ด ยูเคชั่น จำกัด (มหาชน) 2533
2. ประเมษฐ์ ประยานันท์, ปิยพงษ์ เผ่าวนิช, คู่มือและการประยุกต์ใช้ไมโครคอนโทรลเลอร์ MCS-51 บริษัท ซีเอ็ด ยูเคชั่น จำกัด (มหาชน) 2536
3. มนต์วี พจนากุลวณิช , การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอโมซี , บริษัท ซีเอ็ด ยูเคชั่น จำกัด (มหาชน) 2535
4. รศ. มณฑนา ปราการสมุทร, การเขียนชุดคำสั่งภาษาซี, บริษัท ดวงกมลสมัย 2534
5. คู่มือการใช้ - BOARD CP-SB 31, ETT
- LCD
- DEBUGGER