แอดวานส์ซิงเกิลบอร์ด

Advance Single Board

โดย

นางสาว ชุลีกร กิตติหิรัญวัฒน์ 38013269

นายวินัย ประดับมุข 38013290

นายเอก ฟองเมตตาจิต 38013308

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2540

แอดวานส์ซิงเกิลบอร์ด

Advance Single Board

โดย

นางสาว ชุลีกร กิตติหิรัญวัฒน์ 38013269

นายวินัย ประดับมุข 38013290

นายเอก ฟองเมตตาจิต 38013308

อาจารย์ที่ปรึกษา

ผศ. สมศักดิ์ มิตะถา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2540

ปริญญานิพนธ์ปีการศึกษา 2540

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง แอดวานส์ซิงเกิลบอร์ด

ผู้จัดทำ นางสาว ชุลีกร กิตติหิรัญวัฒน์ 38013269

นายวินัย ประดับมุข 38013290

นายเอก ฟองเมตตาจิต 38013308

..... อาจารย์ที่ปรึกษา

(ผศ. สมศักดิ์ มิตะถา)

แอดวานส์ซิงเกิลบอร์ด

นางสาว ชุลีกร กิตติหิรัญวัฒน์ 38013269
นายวินัย ประดับมุข 38013290
นายเอก ฟ่องเมตตาจิต 38013308
ผศ. สมศักดิ์ มิตะถา อาจารย์ที่ปรึกษา
ปีการศึกษา 2540

บทคัดย่อ

ในปัจจุบันได้มีบริษัทต่างๆพัฒนาซิงเกิลบอร์ด (Single Board) รุ่นต่างๆออกมาสู่ท้องตลาดมากมาย ซึ่งแต่ละรุ่น แต่ละบริษัท ต่างก็มีจุดเด่นจุดด้อยต่างกันไป โครงงานนี้เป็นโครงงานที่พัฒนาต่อจากรุ่นที่แล้ว ที่ได้พัฒนาและออกแบบซิงเกิลบอร์ดรุ่นใหม่ให้มีความสามารถ ,คุณลักษณะที่ผู้ใช้สามารถนำไปใช้ประโยชน์ตามต้องการได้โดยสะดวกมากยิ่งขึ้น ซึ่งโครงงานนี้มีจุดประสงค์สำคัญคือ เป็นการตรวจสอบการทำงานทั้งหมดของซิงเกิลบอร์ดที่ได้ออกแบบและสร้างไว้แล้ว โดยในการตรวจสอบการทำงานของซิงเกิลบอร์ด ได้จัดทำใบการทดลองขึ้นมาเพื่อตรวจสอบการทำงานทั้งด้านฮาร์ดแวร์และซอฟแวร์ และได้จัดทำระบบช่วยสอนด้วยคอมพิวเตอร์(CAI)ซึ่งมีเนื้อหาเกี่ยวกับไมโครคอนโทรลเลอร์ตระกูล MCS-51 โดยใช้โปรแกรม Authorware ซึ่ง CAI นี้เหมาะสำหรับผู้ที่ต้องการเรียนรู้ไมโครคอนโทรลเลอร์ด้วยตนเองและสามารถศึกษาได้ง่าย

Advance Single Board

Chuleekorn Kittihiranwat

Vinai Pradapmook

Ake Fongmettajit

Somsak Mittatha Advisor

1997

Abstract

At the present time,There are a lot of firms which develop many generation of single-board . However each generation of each company has individual strong point and weak point . This project is developing the last year project which developted and designed new model of single-board which has wanted capability and a user can use it friendly. The important purpose of this project is testing all function for working of advance single board. For testing the function of advance singleboard we were implemented the laboratory for test into hardware and software. And implement the computer assisted Instruction(CAI) which have detail about Microcontroller MCS-51 which use Authorware Program. This CAI will usefull for persons who have intersesting and easy to study with themself.

กิตติกรรมประกาศ

การที่โครงงานนี้สำเร็จลุล่วงไปได้ด้วยดีนั้น เป็นผลสืบเนื่องมาจากความรู้พื้นฐานหลายๆแขนงที่ได้เรียนมา คำแนะนำต่างๆที่ได้รับจากอาจารย์ในภาควิชาคอมพิวเตอร์ทุกๆท่าน โดยเฉพาะอาจารย์ที่ปรึกษาคือ อาจารย์สมศักดิ์ มิตะถา ที่คอยให้คำปรึกษาด้วยดีมาตลอด และคอยชี้แนะสิ่งที่เป็นประโยชน์ต่อโครงงานนี้เป็นอย่างมาก และขอขอบคุณเพื่อนๆ ที่ได้ช่วยเหลือแนะนำสิ่งต่างๆที่ติดขัด และคอยให้กำลังใจจนงานสำเร็จลุล่วงไปได้

และขอขอบพระคุณอาจารย์ผู้คุมสอบทุกท่านที่ได้สละเวลาอันเป็นประโยชน์ของท่านมาในการสอบโปรเจ็คให้ผ่านลุล่วงไปด้วยดีครั้งนี้

สารบัญ

สารบัญรูปภาพ

สารบัญตาราง

ตารางที่

หน้า

บทที่ 1

บทนำ

1.1 จุดประสงค์ของโครงงาน

เนื่องจากความก้าวหน้าทางเทคโนโลยีในปัจจุบัน เป็นผลให้มีการนำไมโครโปรเซสเซอร์ (Microprocessor) มาใช้ในชีวิตประจำวันมากขึ้น ไม่ว่าจะเป็นเครื่องใช้ไฟฟ้าทั่วไป ,ระบบควบคุม ,ระบบสื่อสารข้อมูล ,เครื่องมือวัดทางไฟฟ้า ,ของเด็กเล่น ,ตลอดจนใช้ในการควบคุมแต่ละจุดการทำงาน ในกระบวนการผลิตของโรงงานอุตสาหกรรม ล้วนแล้วแต่มีไมโครโปรเซสเซอร์เป็นตัวควบคุมการทำงานทั้งสิ้น

แต่จากการพัฒนาที่มีมากขึ้นเป็นลำดับ จึงได้มีการนำวงจรพื้นฐานที่จำเป็นรวมเข้าไปในไมโครโปรเซสเซอร์ ซึ่งเรียกรวมกันว่าไมโครคอนโทรลเลอร์ (Microcontroller) ทำให้ไมโครคอนโทรลเลอร์มีความสามารถมากขึ้น ในขณะที่ขนาดของวงจรและจำนวนชิพ (Chip) ที่ใช้ใน วงจรมีน้อยลง

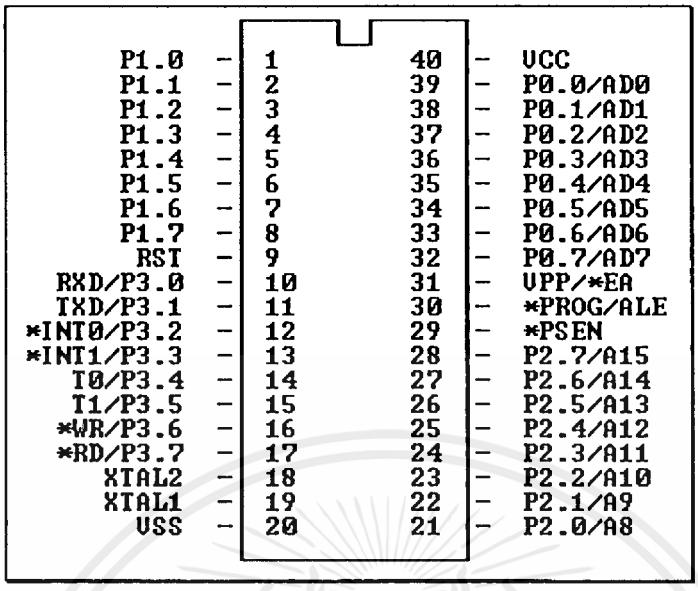
MCS-51 เป็นไมโครคอนโทรลเลอร์แบบชิพเดี่ยว ที่ได้รับความสนใจมากตัวหนึ่งในปัจจุบัน ทั้งนี้เป็นเพราะชิปในตระกูล MCS-51 มีข้อดีเมื่อเทียบกับไมโครโปรเซสเซอร์ขนาด 8 บิตในตระกูลอื่นดังต่อไปนี้

1. มีแรมบรรจุไว้ภายใน 128-256 ไบต์ (Bytes)
2. มีวงจรตั้งเวลา(Timer)/วงจรนับ(Counter) ขนาด 16 บิต (Bits) 2 ตัวอยู่ภายใน
3. มีวงจรรับส่งข้อมูลอนุกรมได้ 2 ทิศทาง โดยสามารถกำหนดอัตราเร็วในการรับและส่งข้อมูล (baud rate) ได้ตั้งแต่ 300 ถึง 375 กิโลบิตต่อวินาที
4. มีสัญญาณนาฬิกาภายในตัว
5. มีพอร์ต (Port) ที่สามารถรับหรือส่งข้อมูลได้ทั้ง 2 ทิศทาง จำนวน 4 พอร์ตๆละ 8 บิต หรือสามารถใช้งานเป็นพอร์ตขนาด 1 บิตแยกจากกัน ทำให้เสมือนมีพอร์ตขนาด 1 บิตใช้งานรวมทั้งสิ้น 32 พอร์ต
6. เป็นชิปที่ได้รับความนิยมสูง จึงมีราคาถูก และหาง่าย

นอกจากนี้ MCS-51 ยังมีคุณสมบัติอื่นๆ ที่น่าสนใจคือ

- ต้องการแหล่งจ่ายไฟ 5 โวลต์ (Volts) เพียงชุดเดียว
- มีหน่วยความจำสำหรับเก็บโปรแกรมควบคุมการทำงานอยู่ภายในชิปอยู่ด้วย ซึ่งอาจเป็นรอม (ROM) หรืออีพรอม (Eprom) ขึ้นอยู่กับรุ่นของชิป เช่น 8XC51 มีขนาด 4 กิโลไบต์ , 80C52 มีขนาด 8 กิโลไบต์ , 8XC51FB มีขนาด 16 กิโลไบต์ , 8XC51FC มีขนาด 32 กิโลไบต์ เป็นต้น (เบอร์ 8031,8032 ไม่มีหน่วยความจำส่วนนี้)
- สามารถใช้หน่วยความจำ สำหรับโปรแกรมและข้อมูลที่อยู่ภายนอกชิปได้อย่างละ 64 กิโลไบต์

- มีคำสั่งคูณและหารเลขขนาด 8 บิตในตัวเอง



รูปที่ 1.1 แสดงการจัดวางขาต่างๆ ของ MCS-51

ซึ่งจุดประสงค์สำคัญของโครงงานนี้คือ ตรวจสอบและทดสอบการทำงานทั้งหมดทางด้านฮาร์ดแวร์และซอร์ฟแวร์(ทดสอบชุดคำสั่งทั้งหมด)ของแอดวานส์ซิงเกิลบอร์ดที่ได้ออกแบบและสร้างไว้แล้ว โดยได้มีการจัดทำใบการทดลองขึ้นมาเพื่อใช้ในการทดสอบการทำงานทั้งหมด และศึกษาความรู้ทั่วไปและการใช้งานของ MCS-51 ซึ่งเป็นซีพียูของแอดวานส์ซิงเกิลบอร์ดนี้ เพื่อใช้ในการออกแบบและสร้างระบบช่วยสอนด้วยคอมพิวเตอร์(CAI) ซึ่งมีเนื้อหาเกี่ยวกับไมโครคอนโทรลเลอร์ตระกูล MCS-51 ทั้งหมด เพื่อใช้เป็นประโยชน์สำหรับผู้ที่มีความสนใจและต้องการศึกษาการใช้งานของ MCS-51 เพื่อนำไปใช้และพัฒนางานที่เป็นประโยชน์ต่อไป

1.2 ขอบเขตของโครงงาน

1. ศึกษาและตรวจสอบการทำงานทั้งหมดของ แอดวานส์ซิงเกิลบอร์ดโดยจัดทำใบงานการทดลองทั้งทางด้านฮาร์ดแวร์และซอร์ฟแวร์เพื่อทดลอบชุดคำสั่งและการติดต่อกับอุปกรณ์ภายนอก

2.จัดทำระบบช่วยสอนด้วยคอมพิวเตอร์(CAI) ซึ่งมีเนื้อหาทั้งหมดเกี่ยวกับ ไมโครตอนโทรลเลอร์ตระกูล MCS-51 เพื่อเป็นการง่ายและเป็นประโยชน์แก่ผู้ที่สนใจและต้องการศึกษาเพื่อนำ MCS-51 ไปใช้งานและพัฒนาต่อไป

1.3 วิธีดำเนินงาน

โครงงานนี้มี จุดประสงค์เพื่อที่จะศึกษา และ ทดสอบการทำงานทั้งหมดของเครื่อง แอวานส์

ซิงเกิลบอร์ด ซึ่งได้มีการออกแบบและสร้างไว้แล้ว ว่าสามารถนำไปใช้งานได้ตามวัตถุประสงค์ที่ตั้งไว้ โดยทำการทดสอบการทำงานทางด้านฮาร์ดแวร์และซอร์ฟแวร์(ทดสอบชุดคำสั่งทั้งหมด) โดยได้มีการจัดทำใบการทดลองขึ้นมาเพื่อใช้ในการทดสอบการทำงานทั้งหมดของแอดวานส์ซิงเกิลบอร์ด ดังแสดงในตารางเป็นการเปรียบเทียบลักษณะความแตกต่างของซิงเกิลบอร์ดต่างๆในท้องตลาด

	JAZZ-31	**ET 4.0**	**MPF-1**	**ANA-2.0**
หน่วยประมวลผลกลาง	MCS51	Z80	Z80	Z80
การแสดงผล	เซเว่นเซกเมนต์ (7-Segment)	LCD แบบ Dot Matrix 16 ตัวอักษรX1บรรทัด	ฟลูออเรสเซนต์ (Fluorescent)	เซเว่นเซกเมนต์
การป้อนโปรแกรม	เลขฐาน 16	Mnemonics	Mnemonics	เลขฐาน 16
แอสเซมเบลอร์ในตัว	ไม่มี	มี(ทีละบรรทัด)	มี	ไม่มี
รูปแบบคีย์บอร์ด	คีย์เลขฐาน 16 จำนวน 16 คีย์ + 12 ฟังก์ชันคีย์	32 คีย์ ตัวอักษร + ตัวเลข	คล้ายเครื่องคอมพิวส์เตอร์ส่วนบุคคล ตัวอักษร, ตัวเลข	คีย์เลขฐาน 16 จำนวน 16 คีย์ + 20 ฟังก์ชันคีย์
อื่นๆ	นิยมใช้ทั่วไป	นิยมใช้ทั่วไป	หายาก	นิยมใช้ทั่วไป

ตารางที่ 1 ตารางเปรียบเทียบลักษณะของซิงเกิลบอร์ดในท้องตลาด

โดยลักษณะที่สำคัญ (specification) ของซิงเกิลบอร์ดในโครงงานมีดังต่อไปนี้

ฮาร์ดแวร์ (HARDWARE)

- ใช้ชิปในตระกูล MCS-51 เป็นหน่วยประมวลผลกลาง
- แสดงผลโดยใช้อุปกรณ์แสดงผลแบบผลึกเหลว LCD (Liquid crystal display) แบบอักขระ (Character LCD Module) ขนาด 4 บรรทัด 20 ตัวอักษร
- มีคีย์บอร์ดภาษาอังกฤษแบบเคอตี้ (Qwerty) ในตัวดังภาพที่ 12
- มีบัสและพอร์ตขยายระบบที่เข้ากันได้กับ JAZZ 31

ซอฟแวร์ (SOFTWARE)

- ป้อนโปรแกรมด้วย Mnemonics ได้

- มีโปรแกรมสร้าง/แก้ไขข้อความ (Editor) และ โปรแกรมแปลรหัสคำสั่ง (Asscmblcr) ในตัว สามารถทำการโปรแกรมและทดสอบโปรแกรมได้ในตัว ทำให้ใช้งานได้โดยลำพัง (Stand Alone) ได้อย่างคล่องตัวขึ้น
- มีโปรแกรมใช้งานทั่วไป (Software Utility) เช่น เครื่องคิดเลข , นาฬิกา เป็นต้น
- เปลี่ยนไปใช้ภาษาเบสิคควบคุมไมโครคอนโทรเลอร์ได้
- สามารถส่งข้อมูล (Upload), รับข้อมูล (Download) และติดต่อกับเครื่องคอมพิวเตอร์ส่วนบุคคลเป็นเทอร์มินอล (Teminal) หรือ Remote Access ผ่านทางพอร์ตอนุกรม RS-232 ของเครื่องคอมพิวเตอร์ส่วนบุคคลได้

1.3.1 การทดสอบการทำงานของเครื่องซิงเกิลบอร์ด

โดยจัดทำใบการทดลองประกอบการทดสอบการทำงานของเครื่อง โดยเน้นทางด้านคำสั่งต่างๆที่ใช้งาน ว่าสามารถใช้งานได้ถูกต้องหรือไม่ โดยมีทั้งหมด 11 การทดลองดังนี้คือ

การทดลองที่ 1 เรื่องคำสั่งการโอนย้ายข้อมูล
การทดลองที่ 2 คำสั่งทางคณิตศาสตร์และลอจิก
การทดลองที่ 3 คำสั่งการกระโดด
การทดลองที่ 4 การบวกเลขฐานสอง
การทดลองที่ 5 คำสั่งเกี่ยวกับบิท
การทดลองที่ 6 การเลื่อนและการหมุนข้อมูล
การทดลองที่ 7 แสตกและรูทีน
การทดลองที่ 8 การคูณเลขฐานสอง
การทดลองที่ 9 การหารเลขฐานสอง
การทดลองที่ 10 การเปลี่ยนเลขฐานสองเป็นรหัส BCD
การทดลองที่ 11 การเปลี่ยนรหัส BCD เป็นเลขฐานสอง

1.3.2 จัดทำระบบช่วยสอนด้วยคอมพิวเตอร์(CAI)

จัดทำระบบช่วยสอนด้วยคอมพิวเตอร์(CAI) ซึ่งมีเนื้อหาทั้งหมดเกี่ยวกับ ไมโครตอนโทรลเลอร์ตระกูล MCS-51 เพื่อเป็นการง่ายและเป็นประโยชน์แก่ผู้ที่สนใจและต้องการศึกษาเพื่อนำ MCS-51 ไปใช้งานและพัฒนาต่อไป ซึ่งมีขั้นตอนดังนี้คือ

1. กำหนดเป้าหมายที่จะนำมาใช้ในการสอน

โดยศึกษาโครงสร้างและการทำงานทั้งหมดของ MCS-51 เพื่อให้ทราบขอบเขตและเป้าหมายของเนื้อหาที่จะนำมาสร้างเป็นบทเรียน CAI

2 รวบรวมข้อมูลที่จะนำมาใช้ในการสอน

เมื่อเราได้เป้าหมายในการสอนว่าจะสอนเรื่องอะไร หัวข้ออะไรบ้าง แล้วนั้น ต่อมาก็จะเป็นการรวบรวมข้อมูลต่างๆ ซึ่งเกี่ยวข้องกับ MCS-51 ทั้งหมด ตั้งแต่บล็อกการทำงานของ MCS-51 การทำงาน

ในภาคต่างๆ ของ MCS-51 ทั้งเรื่องหน่วยความจำ, พอร์ทอินพุท/ พอร์ทเอาท์พุท, วงจรนับ/ จับเวลา, การสื่อสารข้อมูลอนุกรม และระบบการอินเตอร์รัพท์ และชุดคำสั่งการทำงานทั้งหมดของ MCS-51

3 จัดลำดับของเนื้อหาที่จะนำมาสอน

เมื่อรวบรวมข้อมูลได้แล้วก็จะมาจัดลำดับเนื้อหาที่จะนำมาสอน โดยจะเริ่มจาก

→ คุณสมบัติของ MCS-51

→ โครงสร้างของ MCS-51 โดยในส่วนของโครงสร้างก็จะแยกย่อยเข้าไปเป็นส่วนประกอบย่อยต่างๆ

→ ชุดคำสั่งของ MCS-51 โดยในส่วนของชุดคำสั่งก็จะแบ่งออกเป็น 5 กลุ่มย่อย

4. จัดวางลำดับเนื้อหาการนำเสนอในแต่ละหน้าของ CAI

นำเนื้อหาที่จัดเรียงไว้มาจัดลำดับลงเป็นหน้าใน CAI ว่าในแต่ละหน้าใน CAI จะมีเนื้อหาอะไรบ้างและมีรูปแบบอย่างไร

5. ทำการสร้างแต่ละหน้าของ CAI ที่จะใช้ในการสอน

เขียนหรือ วาดรูปส่วนประกอบต่างๆของเนื้อหาที่จะจัดลง CAI โดยใช้ Program Photoshop หรือโปรแกรมอื่นๆที่จำเป็น เช่น ต้องการใช้บล็อกไดอะแกรม ก็อาจใช้ Program Visio หรือโปรแกรอื่นที่สามารถสร้าง ภาพที่เราต้องการได้ หลังจากนั้นนำส่วนประกอบเหล่านั้นรวมเข้าวางปืน 1 หน้าของ CAI จัดความสวยงามและจัดลักษณะการโชว์เนื้อหาในแต่ละหน้าด้วย Program Authorware

6. จัดทำความต่อเนื่องในแต่ละหน้าเข้าด้วยกัน

หลังจากที่ได้ทุกหน้าของเนื้อหาครบแล้ว ก็จะนำแต่ละหน้ามา link เข้าด้วยกันเพื่อให้ CAI สามารถทำงานได้อย่างสมบูรณ์ โดยมีหลักการคือ ดูว่าในแต่ละหัวข้อมีหัวข้อแยกย่อยเข้าไปอีกหรือไม่ โดยจากหัวข้อใหญ่สามารถเข้าสู่หัวข้อย่อยเข้าไปเรื่อยๆได้ และจากหัวข้อย่อยก็สามารถกลับไปที่หัวข้อใหญ่ได้ตามลำดับเรื่อยๆได้ ซึ่งก็คือสามารถเรียนรู้เนื้อหาโดยเลือกหัวข้อที่สนใจได้โดยไม่จำเป็นต้องศึกษาทั้งหมดตั้งแต่เริ่มแรกจนจบ

สรุปผลการทำงาน

จากการทดสอบการใช้งานของซิงเกิลบอร์ดทั้งทางฮาร์ดแวร์ ,และซอฟแวร์ ที่ผ่านมาแล้วนั้น ทำให้ผู้ใช้งานสามารถนำซิงเกิลบอร์ดในโครงงานนี้ไปใช้ประโยชน์ได้โดยไม่ต้องอาศัยเครื่องมืออื่นๆมากนัก เพราะนอกจากจะมีเครื่องมือหรือทูลที่จำเป็นเหมือนซิงเกิลบอร์ดทั่วไปในท้องตลาดแล้ว ยังมีเครื่องมือช่วยในการเขียนโปรแกรมภาษาแอสเซมบลีด้วยในตัว ทั้งเอดิเตอร์, แอสเซมเบลอ อีกทั้งมีการจัดวางคีย์ให้คล้ายคีย์บอร์ดบนคอมพิวเตอร์ส่วนบุคคลทั่วไปให้มากที่สุด เพื่อเพิ่มความสะดวกในการใช้งานและค้นหาตำแหน่งของคีย์ ทำให้การใช้งานแบบ Stand alone มีประสิทธิภาพมากยิ่งขึ้น

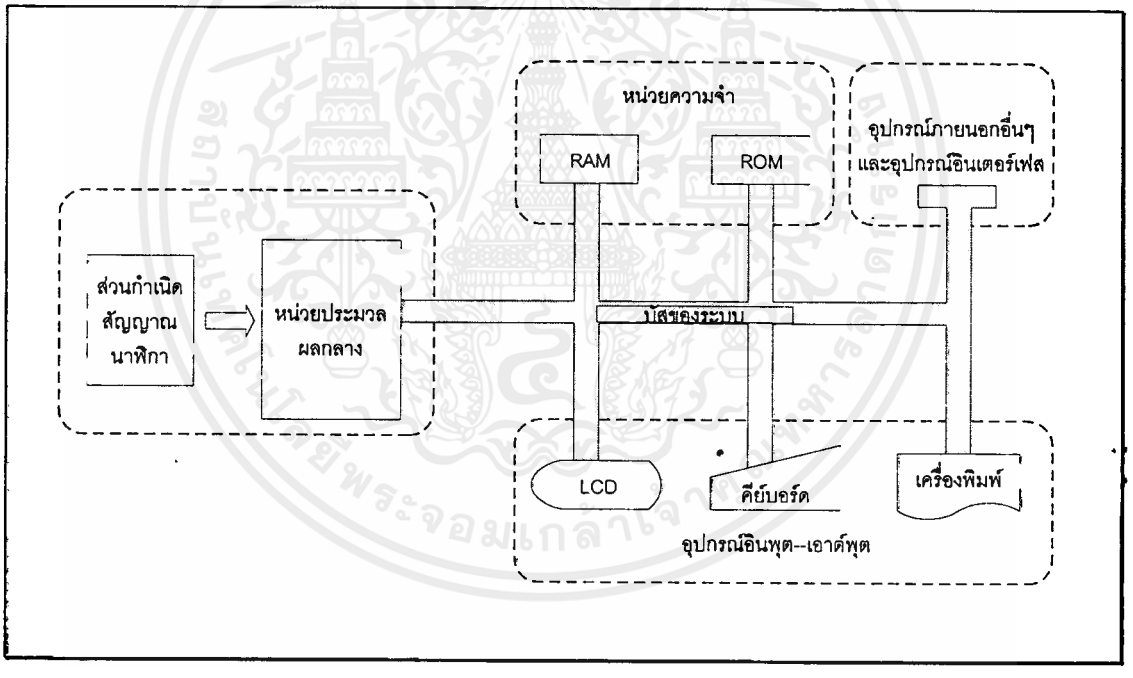
ส่วนการสร้าง CAI ก็เหมาะและเป็นประโยชน์สำหรับผู้ที่มี PC แต่ไม่มีความรู้ทางด้าน MCS-51 ก็สามารถเรียนด้วยตนเองได้

อย่างไรก็ตามเนื่องจากเวลาในการทำโครงงานนี้มีจำกัด ในขณะที่จำนวนเนื้องานที่ต้องทำก็มีมาก ทั้งการศึกษาข้อมูลเกี่ยวกับ MCS-51 ทั้งหมดเพื่อจัดลงในเนื้อหาของ CAI, การคิดหาวิธีและรูปแบบในการสอนของเนื้อหาใน CAI แต่ละหน้าซึ่งมีจำนวนมากพอสมควร และขั้นตอนในการสร้าง CAI จัด และตกแต่งแต่ละหน้าต้องใช้เวลามาก รวมทั้งในขั้นตอนการนำแต่ละหน้ามาเชื่อมเข้าด้วยกันจะมีความยุ่งยากและปัญหามาก จึงเป็นไปได้ที่อาจจะมีข้อผิดพลาดบางประการซึ่งทางผู้จัดทำต้องขออภัยล่วงหน้าและจะได้หาโอกาสดำเนินการแก้ไขต่อไป

บทที่ 2
โครงสร้างและส่วนประกอบของแอดวานส์ซิงเกิลบอร์ด

2.1 ส่วนประกอบทางด้านฮาร์ดแวร์ของซิงเกิลบอร์ด

โครงสร้างโดยทั่วไปของซิงเกิลบอร์ดดังภาพที่ 2 จะแสดงดังภาพที่ 3.1 ซึ่งมีความคล้ายคลึงกับระบบไมโครคอมพิวเตอร์ทั่วไป ซึ่งไม่ได้มีแต่เพียงหน่วยประมวลผลกลางเพียงอย่างเดียวเท่านั้น แต่จะประกอบไปด้วย หน่วยความจำทั้งที่เป็น แรม RAM (Random Access Memmory) ,และ รอม ROM (Read Only Memory) และอุปกรณ์อินพุต (Input) เอาต์พุต (Output) ต่างๆ ที่ทำงานร่วมกับหน่วยประมวลผลกลางด้วย



ภาพที่ 2.1 แผนผังแสดงพื้นฐานของระบบในซิงเกิลบอร์ด

องค์ประกอบพื้นฐานของโครงสร้างของซิงเกิลบอร์ด แบ่งออกเป็นส่วนต่างๆได้ดังนี้

- หน่วยประมวลผลกลาง (Central Processing Unit) เป็นส่วนสำคัญที่ทำหน้าที่ควบคุมการทำงานของส่วนต่างๆของระบบให้เป็นไปตามโปรแกรมที่กำหนดไว้ ซึ่งเก็บไว้ในหน่วยความจำ

- วงจรกำเนิดสัญญาณนาฬิกา (Clock Oscillator) วงจรกำเนิดสัญญาณนาฬิกาเป็นพื้นฐานของระบบ ที่ทำให้แต่ละส่วนของระบบทำงานได้อย่างสัมพันธ์กัน
- หน่วยความจำ (Memory) คือส่วนที่ใช้สำหรับเก็บข้อมูลและโปรแกรม หน่วยความจำแบ่งออกได้เป็น 2 ชนิดคือ รอม และ แรม
    1. รอม (Read Only Memory) คือหน่วยความจำที่ใช้สำหรับเก็บโปรแกรมที่ควบคุมการทำงานของระบบ ถึงแม้จะปิดไฟเลี้ยงแล้วก็ตาม ข้อมูลที่อยู่ในหน่วยความจำรอมนี้ก็จะไม่หายไปด้วย
    2. แรม (Random Access Memory) คือหน่วยความจำในระบบคอมพิวเตอร์ที่ใช้สำหรับเก็บข้อมูลหรือตัวแปรแต่ละชนิดแบบชั่วคราว แต่มีข้อเสียคือเมื่อตัดไฟออกแล้วจะทำให้ข้อมูลหรือสิ่งที่อยู่ในหน่วยความจำนี้หายไป
- อุปกรณ์อินพุต - เอาต์พุต ระบบของคอมพิวเตอร์มีการทำงานตามโปรแกรมที่ได้ตั้งไว้และจำเป็นต้องมีขบวนการในการส่งผลลัพธ์ออกไปแสดงผลภายนอกโดยผ่านพอร์ตเอาต์พุต เช่น จอแสดงผลแบบผลึกเหลว ,หรือในการปฏิบัติการบางอย่างมีความจำเป็นต้องอ่านข้อมูลจากภายนอกโดยผ่านพอร์ตอินพุต เช่น คีย์บอร์ด เป็นต้น

อื่นๆ ได้แก่ภาคจ่ายไฟ , ระบบไฟสำรอง , ระบบตรวจสอบแรงดัน , ส่วนถอดรหัสตำแหน่งแอดเดรส (Address Decoder) เป็นต้น

2.2 ส่วนประกอบทางด้านซอร์ฟแวร์ของซิงเกิลบอร์ด

ซอฟแวร์เป็นส่วนสำคัญในการทำงานของซิงเกิลบอร์ด จะดูแลการทำงานทั้งหมด เป็นตัวที่กำหนดขั้นตอนการทำงานให้กับซิงเกิลบอร์ด เพื่อให้ได้ผลลัพธ์ตามจุดประสงค์ของผู้ใช้งาน มีการทำงานในส่วนของ Editor , Assembler และ Debuger เพื่อให้สะดวกในการใช้งาน จึงสามารถทำงานได้อย่างอิสระด้วยตัวเองได้ สำหรับซอฟแวร์สามารถแบ่งเป็นส่วนย่อยๆได้ 2 ส่วนคือ

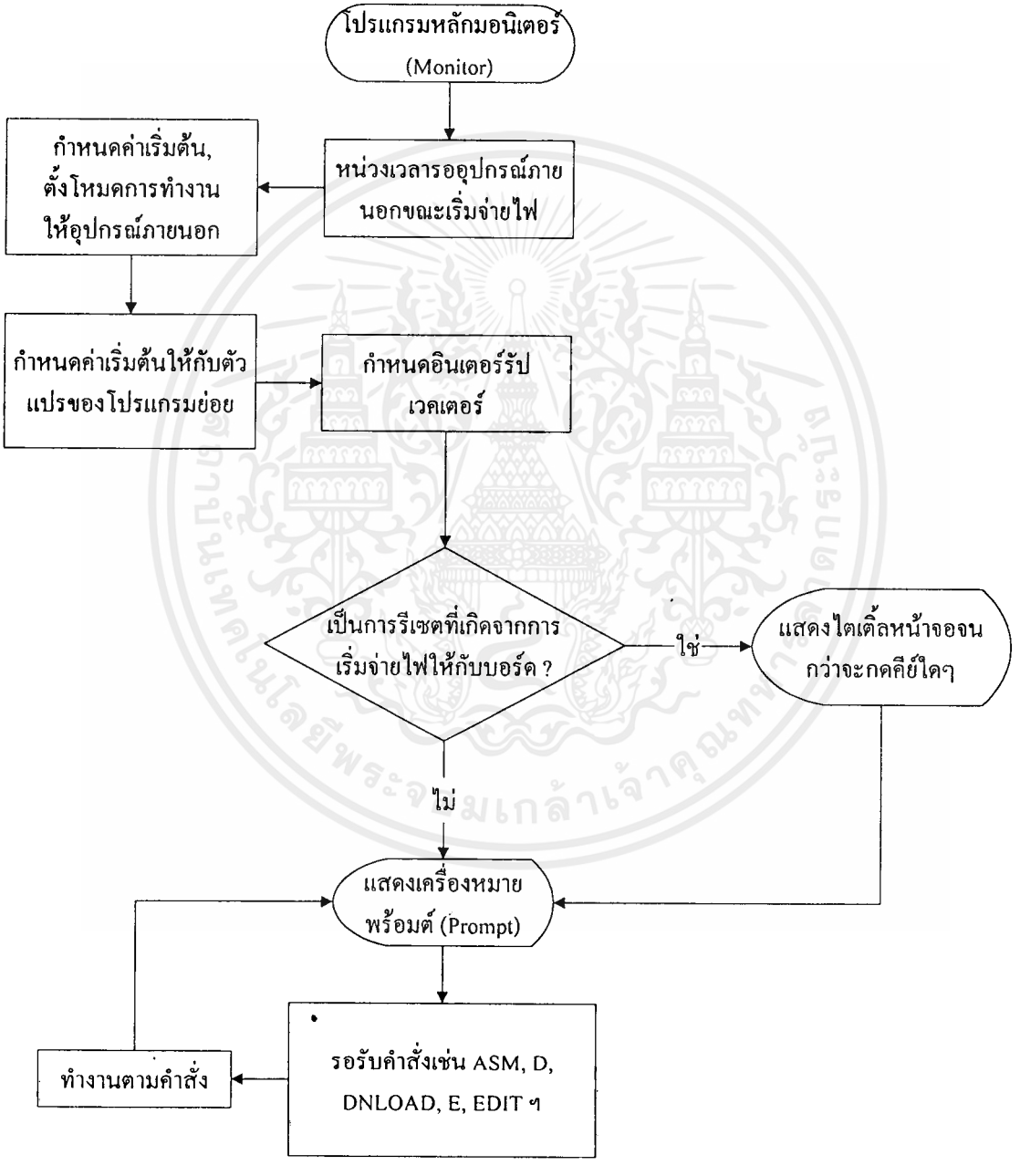
2.2.1 มอนิเตอร์

จากภาพที่2.2 มอนิเตอร์เป็นโปรแกรมส่วนแรกที่เริ่มทำงานในขณะเริ่มจ่ายไฟให้กับบอร์ด เพื่อทำหน้าที่เซตอัพ (Setup) และควบคุม, ดูแลการใช้งานของผู้ใช้ดังนี้

- กำหนดโหมด (Mode) การทำงานเริ่มต้นให้กับอุปกรณ์อินพุตเอาต์พุต (ฮาร์ดแวร์) ดังนี้
- กำหนดค่าเริ่มต้นให้กับหน่วยความจำบางส่วนที่สงวนไว้สำหรับระบบ (ซอฟแวร์) เพื่อเก็บสถานะการทำงาน , ตาราง , พอยเตอร์ (Pointer) , ตำแหน่งอินเตอร์รัปเวกเตอร์ (Interrupt Vector) , บัฟเฟอร์ (Buffer) ฯ ที่จำเป็นต่อระบบและโปรแกรมย่อยของระบบ โดยหน่วยความจำที่สงวนไว้มีดังนี้คือ

- หน่วยความจำชนิดข้อมูลตำแหน่ง 0-3FFFh ใน U4
- รีจิสเตอร์ภายในของหน่วยประมวลผลกลางตำแหน่ง 10h-21h

- ทำหน้าที่คล้ายเชลล์ (Shell) กล่าวคือรอรับคำสั่งจากพร้อมท์ (Prompt) และกระโดดไปปฏิบัติคำสั่งนั้นๆดังตารางที่ 2.1



ภาพที่ 2.2 ภาพแสดงแผนผังการทำงานของมอนิเตอร์

คำสั่ง	พารามิเตอร์ (Parameter)	หมายเหตุ
ASM	-	เรียกโปรแกรมแอสเซมเบลอ (Assembler) เพื่อแปลข้อความของผู้ใช้ในหน่วยความจำที่เป็นภาษาแอสเซมบลี้ (Assembly) เป็นรหัสภาษาเครื่อง (Machine Code) ให้พร้อมที่จะทำงาน
BAUD	[12 | 24 | 48 | 96 | 19 | 1200 | 2400 | 4800 | 9600 | 19200]	แสดง/กำหนดความเร็วในการรับส่งข้อมูลแบบอนุกรม (บิตต่อวินาที)
C	[p | i | d] <address,address> [p | i | d] <address>	เปรียบเทียบค่าในหน่วยความจำ / รีจิสเตอร์ภายใน
CTTY	-	สลับโหมดการแสดงผลระหว่างเทอร์มินอลและอุปกรณ์แสดงผลแบบผลึกเหลว
D	[address ,[address]]	แสดงข้อมูลในหน่วยความจำชนิดข้อมูล
DATE	[dd/mm/yy]	แสดง , ตั้ง วัน/เดือน/ปี ปัจจุบัน
DNLOAD	<address> <-a | -h | -b>	ดาวน์โหลด (Download) ข้อมูลจากคอมพิวเตอร์ภายนอก เป็น แอสกี (Ascii) , ไบนารี (Binary) , หรือ Hex ผ่านพอร์ตอนุกรม
E	<address> [list]	ป้อนเลขฐาน 16 ลงในหน่วยความจำชนิดข้อมูล
EDIT	-	เข้าสู่เอดิเตอร์ (Editor)
EVAL	<expression>	ตีความจากชุดข้อความเป็นตัวเลขมีเครื่องหมายขนาด 2 ไบต์
F	<address,address> <list>	เติมเลขฐาน 16 ลงในหน่วยความจำข้อมูลในช่วงที่กำหนด
G	[address][,address]	กระโดดไปทำงานจากตำแหน่ง หรือในช่วงที่กำหนด
H	[command]	แสดงคำอธิบายถึงคำสั่งต่างๆ
HELP	[command]	แสดงคำอธิบายถึงคำสั่งต่างๆ
I	<port>	แสดงค่าจากพอร์ตหมายเลขนั้นๆ

ตารางที่ 2.1 ตารางแสดงคำสั่งภายในของมอนิเตอร์

คำสั่ง	พารามิเตอร์ (Parameter)	หมายเหตุ
M	[p | i | d] <address,address> [i | d]<address>	สำเนาข้อมูลในหน่วยความจำ/รีจิสเตอร์ภายในเป็นช่วง ไปยังหน่วยความจำข้อมูล/รีจิสเตอร์ภายใน ตำแหน่งอื่น
O	<port> <byte>	ส่งค่าขนาด 1 ไบต์ออกไปยังพอร์ตนั้นๆ
P	[address]	ไปทำงานในโปรแกรมของผู้ใช้ที่ตำแหน่งที่เก็บในพอยเตอร์ หรือที่ผู้ใช้กำหนดไว้ ครั้งละ 1 คำสั่ง ยกเว้นคำสั่งของผู้ใช้เป็นคำสั่ง ACALL และ LCALL ก็จะทำงานจนกว่าจะจบโปรแกรมย่อยที่เรียกไปด้วย ACALL หรือ LCALL
Q	-	จบการทำงาน (Non Interruptable)
RESET	[*]	รีเซตบางส่วน หรือทั้งหมด
R	[number] [= byte]	แสดง /กำหนดค่าในรีจิสเตอร์
S	<address,address> <list>	ค้นหาข้อความ, ตัวเลขฐานใดๆ ในช่วงหน่วยความจำข้อมูลที่กำหนด
STOP-WATCH	-	นาฬิกาจับเวลา
T	[address]	ทำงานที่ตำแหน่งที่เก็บในพอยเตอร์ หรือที่ผู้ใช้กำหนดไว้ ครั้งละ 1 คำสั่ง
TIME	[hh:mm[:ss]]	แสดง / ตั้งเวลาปัจจุบัน
U	[address][,address]	แปลคำสั่งภาษาเครื่องในหน่วยความจำชนิดโปรแกรมเป็นภาษาแอสเซมบลี (Unassemble)
UPLOAD	<address,address> <-a | -b | -h>	อัพโหลด (Download) ข้อมูลไปคอมพิวเตอร์ภายนอก เป็น แอสกี (Ascii) , ไบนารี , Hex ผ่านพอร์ตอนุกรม
VEC	<number> [= address]	เป็นคำสั่งที่แสดง, หรือกำหนด ตำแหน่งอินเตอร์รัปเวกเตอร์ใหม่
?	[command]	แสดงข้อความอธิบายคำสั่งต่างๆ

ตารางที่ 2.1 (ต่อ)

หมายเหตุ

[xxx]	- จะมีหรือไม่ก็ได้
<xxx>	- จำเป็นต้องมี
number,byte	- ตัวเลขขนาด 1 ไบต์ เช่น 0, 0ah (ต้องขึ้นด้วยตัวเลขเสมอ)
port	- ตัวเลขขนาด 2 ไบต์แทนหมายเลขของพอร์ต เช่น 4000h, 0a000h ฯ (ต้องขึ้นด้วยตัวเลขเสมอ)
address	- ตำแหน่งของหน่วยความจำขนาด 2 ไบต์ (ต้องขึ้นด้วยตัวเลขเสมอ)
[p | i | d] <address>	- ชนิดของหน่วยความจำ (พารามิเตอร์ปกติคือ d) เมื่อ
i	= รีจิสเตอร์ภายในหน่วยประมวลผลกลาง เช่น 'i 0E0h' หมายถึง ACC
p	= หน่วยความจำที่เก็บคำสั่ง (อ้างด้วย "MOVC") เช่น 'p 8000h'
d	= หน่วยความจำที่เก็บข้อมูล (อ้างด้วย "MOVX") เช่น 'd 4000h'
list	- ลำดับรายการของ ไบต์|ข้อความ [[,] ไบต์|ข้อความ [[,]...]] เช่น 'Hello',20h,0Dh,0Ah,'New Line'
command	- คำสั่งในมอนิเตอร์เช่น ASM, I, BAUD เป็นต้น
expression	- [จำนวน(ไบต์)|ข้อความ [[,]จำนวน(ไบต์)|ข้อความ [...]]] เช่น 'String 1',0Dh,0Ah,'String 2',26

2.2.2 คำสั่งในมอนิเตอร์

ดังแสดงไปแล้วในตารางที่ 2.1 จะเห็นว่าคำสั่งที่เรียกใช้ได้จากมอนิเตอร์พร้อมท์ (Monitor Prompt) มีอยู่มากมาย ซึ่งแต่ละคำสั่งมีรายละเอียดดังนี้

- ASM

คำสั่ง 'ASM' เป็นคำสั่งที่นำโปรแกรมของผู้ใช้ที่ได้รับการสร้าง, แก้ไข (ในหน่วยความจำตำแหน่ง 4000h-7FFFh) ซึ่งเป็นภาษาแอสเซมบลีของ MCS-51 ที่ยังอยู่ในรูปของรหัสแอสกีมาตรวจสอบไวยากรณ์ , ตีความ, และสร้างรหัสภาษาเครื่องออกมาและเก็บในหน่วยความจำโปรแกรมของผู้ใช้ (8000h-FFEFh) โดยมีความสามารถเช่นเดียวกับ SXA51 ซึ่งเป็นแอสเซมเบลอสำหรับ MCS-51 ซึ่งทำงานบน PC ที่ได้รับความนิยมในกรใช้งานปัจจุบันตัวหนึ่ง โดยมีขั้นตอนการทำงานแบ่งเป็น 2 ช่วงที่สำคัญคือ

- เฟส(Phase) ที่ 1
  - ◊ จะทำการหาอ่านตัวอักษรเข้ามาและตีความเป็นโทเคน (Token)

- ◊ ดูว่าเป็นโทเคนในแต่ละบรรทัดที่ประกอบกันนั้นเป็นชุดคำสั่งใด ถูกต้องตามไวยากรณ์หรือไม่ มีขนาดเท่าใด หากไม่มีข้อผิดพลาดในโปรแกรมของผู้ใช้ จึงเพิ่มค่าของพอยเตอร์ (Pointer) เท่ากับขนาดคำสั่ง เพื่อจองที่ให้กับคำสั่งดังกล่าวไว้
- ◊ หากพบคำสั่งเทียม "ORG" ก็จะนำค่าที่ตามหลังมาไปเก็บในพอยน์เตอร์
- ◊ หากพบคำสั่งเทียม "DB" , "DW" , "DS" ก็จะเพิ่มค่าของพอยน์เตอร์ไปเท่ากับจำนวนตัวเลข/ตัวอักษร ที่ตามหลัง , จำนวนตัวเลข/ตัวอักษร ที่ตามหลัง * 2 , หรือที่กำหนดไว้ ตามลำดับ เช่นเดียวกับ SXA51
- ◊ หากโทเคนดังกล่าวเป็นชื่อเลเบล (Label) และไม่มีความผิดพลาดเกิดขึ้นจากการตั้งชื่อ จะนำค่าในพอยน์เตอร์ดังกล่าวซึ่งเป็นตำแหน่งจริงๆของเลเบลนั้นๆพร้อมชื่อไปเก็บไว้ในตารางชั่วคราวตารางตัวแปร (Variable Table)
- ◊ หากพบคำสั่งเทียม "EQU" ก็จะนำค่าที่ตามหลังมา พร้อมชื่อเลเบลไปเก็บในตารางตัวแปร

- เฟสที่ 2
  - ◊ ทำงานคล้ายกับในเฟสที่ 1 คืออ่านตัวอักษรเข้ามา และแทนด้วยเลขโทเคน ตรวจสอบโทเคนในแต่ละบรรทัดที่ประกอบกันว่าเป็นคำสั่งใด ถูกต้องตามไวยากรณ์หรือไม่ หากไม่มีปัญหาใดๆ ก็จะวางรหัสคำสั่งลงไปและเพิ่มค่าของพอยน์เตอร์ ตามขนาดคำสั่ง
  - ◊ หากโทเคนเป็นชนิดเลเบล ก็จะไปดูตำแหน่งจากตาราง Variable Table หากพบ และ
    - ◆ หากคำสั่งดังกล่าวอ้างอิงเลเบลแบบโดยตรง เช่นคำสั่ง "LCALL" ก็จะวางตำแหน่งของเลเบลนั้นลงไปเลย
    - ◆ หากมีการอ้างอิงแบบเป็นเพจ (Page) เช่น "ACALL" หรืออ้างอิงแบบสัมพัทธ์ (Relative) ก็จะต้องนำค่าในตารางกับค่าในพอยน์เตอร์ (แอดเดรสปัจจุบัน) ไปคำนวณเสียก่อน
  - ◊ หากพบคำสั่งเทียม "ORG" ก็จะนำค่าที่ตามหลังมาไปเก็บในพอยน์เตอร์
  - ◊ หากพบคำสั่งเทียม "DB" , "DW" , "DS" ก็จะใส่ตัวเลข/รหัสแอสกี ที่ตามหลังลงไปในหน่วยความจำโปรแกรม และเพิ่มค่าของพอยน์เตอร์ เหมือนกับโปรแกรม SXA51

◊ หากพบคำสั่งเทียม "EQU" ก็จะข้ามไปบรรทัดต่อไป

ต่อไปนี้เป็นตารางตัวอย่างแสดงโครงสร้างการเก็บไวยากรณ์ของแต่ละคำสั่ง

โทเคนคำสั่ง (Token)	โครงสร้างโอเปอแรนด์ (Operand Stucture)	ขนาด (ไบต์)	รหัสคำสั่ง (Opcode)	โหมดการคำนวณของโอเปอแรนด์
T_ACALL	'*',0	2	11h	AREL
T_CLR	T_A,0,	1	E4h	NOP_
T_XCH	T_A,',@!',0	1	C6h	REG

หมายเหตุ

* - ข้อความ (expression) ใดๆ เช่น (500*2)+Label1

? - รีจิสเตอร์ R0..R7

! - รีจิสเตอร์ R0 or R1

T_xxx - หมายเลขประจำโทเคนซึ่งกำหนดไว้ก่อนหน้านี้เช่น T_ACALL EQU 80h

0 - เป็นเครื่องหมายบอกจบของโครงสร้างโอเปอแรนด์

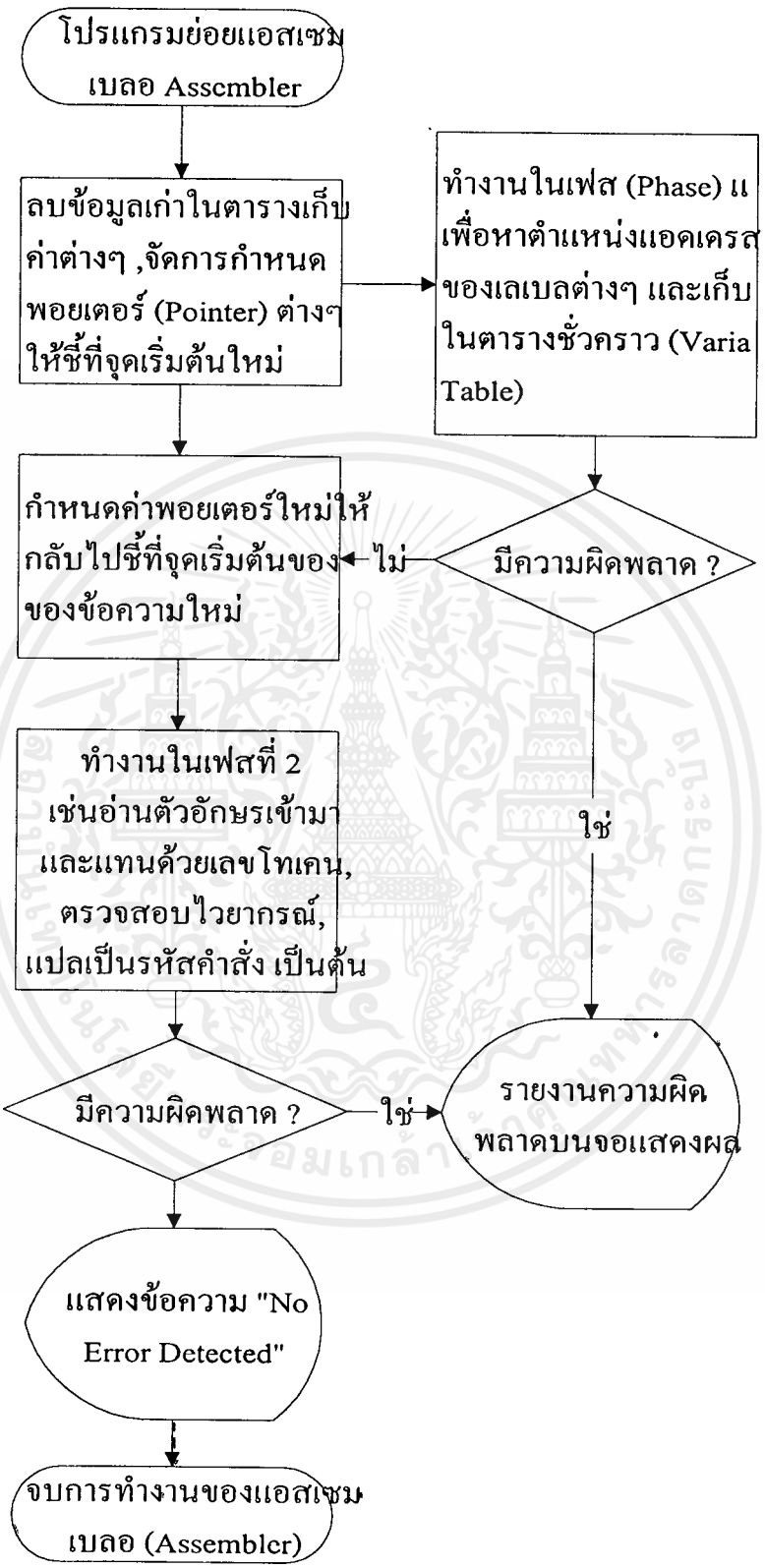
NOP_ - ไม่ต้องทำสิ่งใดหลังจากใส่รหัสภาษาเครื่องลงไป

REG - คำนวนโอเปอแรนด์ตามแบบรีจิสเตอร์เช่น "INC Rx" -> 0000xxxx

"INC R0" -> 00001000

AREL - คำนวณแบบเพจ (1 เพจ ขนาด 2 กิโลไบต์)

ตารางที่ 2.2 ตารางตัวอย่างแสดงโครงสร้างการเก็บไวยากรณ์ของแต่ละคำสั่ง



ภาพที่ 2.3 ภาพแสดงแผนผังแสดงขั้นตอนการทำงานของแอสเซมเบลอ

สำหรับส่วนที่ใช้ในการแปลข้อความ (expression) ให้เป็นตัวเลขที่เป็นค่าคงที่นั้น สามารถเขียนเป็นไวยากรณ์แบบ Attributed Translation Grammar อย่างคร่าวๆ ดังแสดงต่อไปนี้

$Expr_p \rightarrow Term_q\ Elist_{q,p}$

$Elist_{p,p} \rightarrow e$

$Elist_{p,q} \rightarrow + Term_r\{Add\}_{p,r,s}\ Elist_{s,q}$

$Elist_{p,q} \rightarrow - Term_r\{Sub\}_{p,r,s}\ Elist_{s,q}$

$Elist_{p,q} \rightarrow and\ Term_r\{Add\}_{p,r,s}\ Elist_{s,q}$

$Elist_{p,q} \rightarrow shr\ Term_r\{Shr\}_{p,r,s}\ Elist_{s,q}$

$Elist_{p,q} \rightarrow shl\ Term_r\{Shl\}_{p,r,s}\ Elist_{s,q}$

$Elist_{p,q} \rightarrow xor\ Term_r\{Xor\}_{p,r,s}\ Elist_{s,q}$

$Elist_{p,q} \rightarrow mod\ Term_r\{Mod\}_{p,r,s}\ Elist_{s,q}$

$Elist_{p,q} \rightarrow or\ Term_r\{Or\}_{p,r,s}\ Elist_{s,q}$

$Term_p \rightarrow STerm_q\ Tlist_{q,p}$

$Tlist_{p,p} \rightarrow e$

$Tlist_{p,q} \rightarrow * STerm_r\{Mul\}_{p,r,s}\ Tlist_{s,q}$

$Tlist_{p,q} \rightarrow / STerm_r\{Div\}_{p,r,s}\ Tlist_{s,q}$

$STerm_p \rightarrow Factor_q\ Stlist_{q,p}$

$STlist_{p,p} \rightarrow e$

$STlist_{p,q} \rightarrow \wedge Factor_r\{Times\}_{p,r,s}\ Stlist_{s,q}$

$Factor_p \rightarrow SFactor_q\ Flist_{q,p}$

$Flist_{p,p} \rightarrow e$

$Flist_{p,q} \rightarrow . SFactor_r\{OpenTable\}_{p,r,s}\ Flist_{s,q}$

$Sfactor_p \rightarrow (Expr_p)$

$Sfactor_p \rightarrow ident_p$

$Sfactor_p \rightarrow + (Expr_p)$

$Sfactor_p \rightarrow + ident_p$

$Sfactor_p \rightarrow - (Expr_s)\ \{2'S\}_{s,p}$

$Sfactor_p \rightarrow - ident_s\ \{2'S\}_{s,p}$

$Sfactor_p \rightarrow high\ (Expr_s)\ \{Hi\}_{s,p}$

$Sfactor_p \rightarrow$ high $(Expr_s)$ $\{Hi\}_{s,p}$

$Sfactor_p \rightarrow$ high $ident_s$ $\{Hi\}_{s,p}$

$Sfactor_p \rightarrow$ low $(Expr_s)$ $\{Lo\}_{s,p}$

$Sfactor_p \rightarrow$ low $ident_s$ $\{Lo\}_{s,p}$

โดยไวยากรณ์ดังกล่าวนี้จะทำการแปลเป็นตัวเลขขนาด 2 ไบต์แบบมีเครื่องหมายเช่นเดียวกับโปรแกรม SXA51 เช่น

(200 + 200) / 200 + 1 -1 → 1

ACC.0 + 1 → 0E1h

- BAUD

```
BAUD [12 | 24 | 48 | 96 | 19 | 1200 |
      2400 | 4800 | 9600 | 19200]
```

เป็นคำสั่งที่แสดง /กำหนดความเร็วในการรับส่งข้อมูลผ่านพอร์ตอนุกรม TXD และ RXD ในหน่วยเป็น บิตต่อวินาที โดยสามารถกำหนดได้ 5 ระดับคือ 1200, 2400, 4800, 9600, 19200 บิตต่อวินาที โดยกำหนดผ่านรีจิสเตอร์ฟังก์ชันพิเศษ (Special Function Register) คือ PCON, SCON, TMOD, TH1, TL1

เช่น BAUD 9600 - 9600 บิตต่อวินาที

BAUD 19 - 19200 บิตต่อวินาที

BAUD - แสดงความเร็วในการรับส่งข้อมูล

- C

```
C [p | i | d] <address,address>
  [p | i | d] <address>
```

เป็นคำสั่งเปรียบเทียบ (Compare) ค่าในหน่วยความจำ / รีจิสเตอร์ภายใน ที่เป็นช่วง กับหน่วยความจำ / รีจิสเตอร์ อีกช่วงหนึ่ง เช่น

c p 8000h,8010h d 4000h - เปรียบเทียบค่าในหน่วยความจำโปรแกรมจากตำแหน่ง 8000h-8010h กับหน่วยความจำชนิดข้อมูลตำแหน่ง 4000h-4010h

- CTTY

เป็นคำสั่งสลับโหมดการแสดงผลและควบคุมบอร์ดไปมาระหว่าง จากคอมพิวเตอร์ที่เป็นเทอร์มินอลและจากตัวบอร์ด (Local) เอง (Change Terminal Type) โดยการสลับบิตบางบิตไปมา (Toggle) ในหน่วยความจำของระบบ โดยโปรแกรมย่อยสำหรับการแสดงผลและการรับคำสั่งจะตรวจสอบจากบิตดังกล่าวเอง

- D

D [address [,address]]

เป็นคำสั่งแสดงข้อมูล (Dump) ในหน่วยความจำชนิดข้อมูล โดยจะแสดงในช่วงที่กำหนด เช่น

d 1000h ,1010h - แสดงข้อมูลในหน่วยความจำชนิดข้อมูลในตำแหน่ง 1000h-1010h (เฉพาะเทอร์มินอลโหมด)

d - แสดงข้อมูลต่อจากตำแหน่งเดิมก่อนหน้า

d 1000h - แสดงข้อมูลในหน่วยความจำชนิดข้อมูลเริ่มจากตำแหน่ง 1000h

<u>หมายเหตุ</u> สำหรับโหมดการแสดงผลที่เป็น LCD หลังจากที่เรียกคำสั่งดังกล่าวแล้ว จะยังไม่จบการทำงานทันทีเหมือนเทอร์มินอลโหมด แต่จะมีคีย์ควบคุมการทำงานดังนี้

SPACE - ดูข้อมูลในไบต์ถัดไป

'-' - ดูข้อมูลในไบต์ก่อนหน้านี้

MON - จบการทำงาน

- DATE

DATE [dd/mm/yy]

เป็นคำสั่งที่แสดง , เปลี่ยน วัน/เดือน/ปี ปัจจุบัน โดยอ่านเขียนข้อมูล, คำสั่ง ลงบนรีจิสเตอร์ของชิปฐานเวลาจริงดังแสดงในตารางที่ 3.3 เช่น

DATE - แสดงวันที่ปัจจุบัน

DATE 25/3/97 - ตั้งวันที่ปัจจุบันเป็น 25/3/97

รีจิสเตอร์	ตำแหน่งแอดเดรส	หมายเหตุ
RTC0	FF10h	วินาทีหลักหน่วย
RTC1	FF11h	วินาทีหลักสิบ
RTC2	FF12h	นาทีหลักหน่วย
RTC3	FF13h	นาทีหลักสิบ
RTC4	FF14h	ชั่วโมงหลักหน่วย
RTC5	FF15h	ชั่วโมงหลักสิบ
RTC6	FF16h	วันที่หลักหน่วย
RTC7	FF17h	วันที่หลักสิบ
RTC8	FF18h	เดือนหลักหน่วย
RTC9	FF19h	เดือนหลักสิบ
RTCA	FF1Ah	ปีหลักหน่วย
RTCB	FF1Bh	ปีหลักสิบ
RTCC	FF1Ch	เลขประจำสัปดาห์
RTCD	FF1Dh	30SEC,ADJ,IRQ,BUSY,HOLD
RTCE	FF1Eh	T1,T0,ITRPT/STND,MASK
RTCF	FF1Fh	TEST,24/12,STOP,REST

ตารางที่ 2.3 ตารางแสดงตำแหน่งรีจิสเตอร์ของชิปฐานเวลาจริง

- DNLOAD

```
DNLOAD [address] <-a-h-b>
```

เป็นคำสั่งที่อ่านข้อมูลจากคอมพิวเตอร์ภายนอกผ่านพอร์ตอนุกรม ในรูปของรหัส แอสกี , ไบนารี , หรือ Hex และเปลี่ยนอยู่ในรูปของไบนารีเพื่อนำไปไว้ยังหน่วยความจำชนิดข้อมูลในส่วนของผู้ใช้ตามที่ผู้ใช้ต้องการ

สำหรับการโหลดโปรแกรมจะแสดงข้อความ "Waiting Data" รอจนกว่าจะอ่านพบอักษรตัวแรกจากพอร์ตอนุกรม จากนั้นจะเริ่มอ่านข้อมูลจากพอร์ตอนุกรมเข้ามาเรื่อยๆ หากไม่พบก็จะวนกลับมาอ่านใหม่ และจะหยุดอ่านก็ต่อเมื่อวนกลับมาถึง 100*100 รอบแล้ว หากยังไม่มีข้อมูลเข้ามา (ประมาณไม่ถึง 1 วินาที) ก็จะจบการทำงาน เช่น

DNLOAD 4000h -a - รับข้อมูลจากพอร์ตอนุกรมในรูปของรหัสแอสกีและเริ่มเก็บจากตำแหน่ง 4000h

DNLOAD 8000h -h - รับข้อมูลจากพอร์ตอนุกรมในรูปของ Intel Hex Format และเปลี่ยนเป็นไบนารี เพื่อนำไปเก็บเริ่มจากตำแหน่ง 8000h

<u>หมายเหตุ</u> สำหรับการโหลดแบบแอสกี จะตัดตัวอักษร 0Dh (Line Feed) ออกไปจาก สัญญลักษณ์จบบรรทัด (End of Line) เพื่อประหยัดเนื้อที่สำหรับเก็บข้อความ เนื่องจากเอดิเตอร์ใช้เพียงตัวอักษร 0Ah ตัวเดียวแทนสัญญลักษณ์จบบรรทัดก็เพียงพอแล้ว

- E

E <address> [list]

เป็นคำสั่งที่ใช้ในการป้อนตัวเลขฐาน 16 ด้วยค่าที่ตามหลังตำแหน่งแอดเดรสปลายทางลงไปในหน่วยความจำชนิดข้อมูล ที่ต้องการ เช่น

e 8000h 12,'ab' -> หน่วยความจำ 8000h = 0Ch

8001h = 61h

8002h = 62h

แต่หากไม่ตัวเลข/ตัวอักษรตามหลังตำแหน่งแอดเดรส เช่น 'e 8000h' โปรแกรมก็จะแสดงค่าปัจจุบันก่อนและรอให้ผู้ใช้ป้อนค่าเข้าไปใหม่ หากผู้ใช้กดคีย์ Space ค่าใหม่ก็จะถูกบันทึกและเลื่อนไปตำแหน่งถัดไปจนถึงตำแหน่งสูงสุด จากนั้นก็จะวนไปตำแหน่งต่ำสุดอีกครั้ง จนกว่าจะกดปุ่ม Enter ซึ่งค่าใหม่ก็จะถูกบันทึกเช่นกันแต่จากนั้นก็จะจบการทำงาน

- EDIT

เป็นคำสั่งที่เรียกโปรแกรมเอดิเตอร์เพื่อเข้าไปทำการแก้ไขข้อความของผู้ใช้ในหน่วยความจำชนิดข้อมูลที่ตำแหน่ง 4000h-7FFFh

คีย์ที่ใช้งาน	ความหมาย
CTRL-QS	เลื่อนเคอร์เซอร์ไปต้นบรรทัด
CTRL-QD	เลื่อนเคอร์เซอร์ไปท้ายบรรทัด
CTRL-QF	ค้นหาคำที่ต้องการ (เลือก Case Sensitive On/Off ได้) [Find]
CTRL-QX	จบการทำงานของเอดิเตอร์ (Quit)
CTRL-L	ค้นหาคำถัดไป [Find Next]
CTRL-J	เปลี่ยนบรรทัด [Goto Line]
CTRL-C	เลื่อนหน้าจอลง [Page Down]
CTRL-R	เลื่อนหน้าจอขึ้น [Page UP]
CTRL-V	พิมพ์แทรก/พิมพ์ทับ [Insert Mode On/Off]
CTRL-H	ลบตัวอักษร [Back Space]
CTRL-G	ลบตัวอักษร [Delete]
CTRL-Y	ลบทั้งบรรทัด [Delete Line]
CTRL-A	เลื่อนเคอร์เซอร์ไปทางซ้าย 1 คำ [Word Left]
CTRL-F	เลื่อนเคอร์เซอร์ไปทางขวา 1 คำ [Word Right]
CTRL-S	เลื่อนเคอร์เซอร์ไปทางซ้าย 1 ตัวอักษร [Character Left]
CTRL-D	เลื่อนเคอร์เซอร์ไปทางขวา 1 ตัวอักษร [Character Right]
CTRL-E	เลื่อนเคอร์เซอร์ขึ้น 1 บรรทัด [Line Up]
CTRL-X	เลื่อนเคอร์เซอร์ลง 1 บรรทัด [Line Down]
CTRL-KR	อ่านข้อความจากพอร์ตอนุกรม (Down Load) มายังหน่วยความจำข้อมูลที่เป็นบัฟเฟอร์ (Buffer) ของเอดิเตอร์
CTRL-KW	ส่งข้อความที่แก้ไขจากหน่วยความข้อมูลจำผ่านพอร์ตอนุกรม (Upload)
CTRL-N	เปลี่ยนโหมดการแสดงผลเทอร์มินอล <-> อุปกรณ์แสดงผลบนบอร์ด

ตารางที่ 2.4 ตารางแสดงชุดคำสั่งพิเศษในโปรแกรมเอดิเตอร์

โดยการทำงานหลังจากกำหนดค่าเริ่มต้นต่างๆ (Configuration) แล้วก็จะรอรับอินพุตจากผู้ใช้ (พอร์ตอนุกรม หรือ คีย์บนบอร์ด)

- หากเป็นคีย์พิเศษในตารางที่ 3.4 ก็จะทำหน้าที่ไปตามจุดประสงค์ของคีย์นั้นๆ

- หากเป็นคีย์ Enter ก็จะตรวจสอบว่าบัฟเฟอร์เต็มหรือยัง ถ้ายังมีที่ว่างก็จะแทรกตัวอักษร 0Dh ลงไปเพียงตัวเดียวเพื่อแทนสัญญลักษณ์จบบรรทัด และแสดงผลบนหน้าจอใหม่เท่าที่จำเป็น
- หากเป็นคีย์ DEL , Backspace ก็จะลบตัวอักษรครั้งละ 1 ตัวเช่นเดียวกับคอมพิวเตอร์ทั่วไป
- หากเป็นคีย์ TAB และยังมีที่เหลือในหน่วยความจำก็จะแทรกตัวอักษร 20h ลงไป 8 ตัว
- คีย์ที่เหลือหากยังมีที่ว่างอยู่ ก็จะแทรกรหัสแอสกีของคีย์นั้นลงไป

- EVAL

EVAL <expression>

เป็นคำสั่งซึ่งเรียกโปรย่อยของแอสเซมเบลอเพื่อตีความเป็นเลขจำนวนเต็มมีเครื่องหมายขนาด 2 ไบต์และแสดงผลออกมา เช่น

EVAL (2+2-2)*2 -> 4

- F

F <address,address> <list>

เป็นคำสั่งเติมเลขฐาน 16 ที่ลงในหน่วยความจำชนิดข้อมูลในช่วงตำแหน่งที่กำหนดด้วยตัวเลข/รหัสแอสกี ที่ตามหลังช่วงของตำแหน่ง เช่น

f 8000h 8003h 01,02 -> หน่วยความจำข้อมูล

8000h = 01h

8001h = 02h

8002h = 01h

8002h = 02h

- G

G [address][,address]

เป็นคำสั่งที่จะให้หน่วยประมวลผลกลางกระโดดทำงานที่ตำแหน่งใดๆ หรือทำงานเป็นช่วงที่กำหนดเช่น

G 8000h - เริ่มทำงานจากตำแหน่ง 8000h

G 8000h,9000h - เริ่มทำงานจากตำแหน่ง 8000h และจะหยุดจนกว่าจะถึง

ตำแหน่ง9000h

G ,8003h - เริ่มทำงานต่อจากปัจจุบันและหยุดจนกว่าจะถึงตำแหน่ง . 8003

G - เริ่มทำงานต่อจากปัจจุบัน

- H,HELP,?

H [command]
HELP [command]
? [command]

เป็นคำสั่งเพื่อแสดงข้อความอธิบายคำสั่งต่างๆคำสั่งใดคำสั่งหนึ่ง หรือทั้งหมดที่มีอยู่ในมอนิเตอร์โปรแกรม เช่น H EDIT -> อธิบายคำสั่ง EDIT

? -> แสดงคำสั่งทั้งหมด

- I

I <port>

เป็นคำสั่งแสดงค่าจากพอร์ตหมายเลขนั้นๆ บนจอเช่น

I 0FFFFh - อ่านข้อมูลจากพอร์ตหมายเลข 0FFFFh

<u>หมายเหตุ</u> หากค่าของพอร์ตอยู่ในช่วง 0000h-FEFFh จะเป็นการอ่านข้อมูลจากหน่วยความจำชนิดข้อมูลแทน

- M

M [p | I | d] <address,address>
[i | d]<address>

เป็นคำสั่งเพื่อทำสำเนาข้อมูลในหน่วยความจำใดๆเป็นช่วง ไปยังหน่วยความจำที่เป็นชนิดข้อมูล หรือรีจิสเตอร์ภายในอีกตำแหน่งหนึ่งเช่น

m p 1000h ,1001h i 00h -> (i00) = (p1000h)

(i01) = (p1001h)

m p 1000h ,1001h d 8000h -> (d8000h) = (p1000h)

(d8001h) = (p1001h)

- O

O <port> <byte>

เป็นคำสั่งส่งตัวเลขฐาน 16 จำนวน 1 ไบต์ออกไปยังพอร์ตหมายเลขนั้นๆ เช่น O 0FFFFh 20h -> ส่งค่า 20h ไปยังพอร์ตหมายเลข 0FFFFh

<u>หมายเหตุ</u> หากค่าของพอร์ตอยู่ในช่วง 0000h-FEFFh จะเป็นการส่งข้อมูลไปยังหน่วยความจำชนิดข้อมูลแทน

- P

P [address]

เป็นคำสั่งให้หน่วยประมวลผลกลางทำงานที่ตำแหน่งปัจจุบัน หรือกำหนดไว้ ทีละคำสั่ง และกลับมาที่มอนิเตอร์และแสดงค่าในรีจิสเตอร์บนจอ ยกเว้นเป็นคำสั่ง ACALL และ LCALL จะทำงานจนกว่าจะจบโปรแกรมย่อยที่ ACALL หรือ LCALL เรียกไป เช่น

P 8000h -> กระโดดไปทำงานที่ตำแหน่ง 8000h

- Q

เป็นคำสั่งที่สั่งให้หน่วยประมวลผลหยุดทำงานใดๆ (Sleep Mode) และจะไม่รับสัญญาณอินเตอร์รัปใดๆอีกต่อไป

- R

R [number] [= byte]

เป็นคำสั่งแสดง /กำหนดข้อมูลในรีจิสเตอร์ต่างๆ ตัวอย่างเช่น

R -> แสดงข้อมูลในรีจิสเตอร์ทั่วไปทั้งหมด (สำหรับโหมด LCD จะใช้คีย์ลูกศร บน, ล่าง ในการเลื่อนจอ และคีย์ MON ในการจบการทำงาน)

R 0 -> แสดงข้อมูลเริ่มจากรีจิสเตอร์ 0 (เฉพาะ LCD)

R ACC -> แสดงข้อมูลในรีจิสเตอร์ A

R DPH = 0 -> กำหนดค่าในรีจิสเตอร์ DPH ให้เป็น 0

- RESET

RESET [*]

เป็นคำสั่งรีเซตเพื่อสั่งให้หน่วยประมวลกลางเริ่มทำงานใหม่ , เคลียร์ (Clear) ข้อมูลบางส่วนเช่น ข้อมูลในรีจิสเตอร์ ฯ และยังเก็บข้อมูลในบัฟเฟอร์หรือหน่วยความจำบางส่วน (RESET) หรือรีเซตการทำงานและเคลียร์ข้อมูลใหม่ทั้งหมดและแสดงไตเติ้ลใหม่ (RESET *)

- S

S <address,address> <list>

เป็นคำสั่งค้นหาข้อความ, ตัวเลขฐานใดๆ ในช่วงหน่วยความจำชนิดข้อมูลที่กำหนด หากพบก็จะแสดงตำแหน่งที่พบออกมาด้วย เช่น

S 0A000h ,0B000h 'abcd' -> ค้นหาชุดของเลขฐาน 16 ซึ่งตรงกับรสัศแอสกีของ 'abcd' ในช่วงหน่วยความจำชนิดข้อมูลตำแหน่ง A000h-B000h

- STOPWATCH

STOPWATCH

เป็นคำสั่งที่ใช้ชิงเกิลบอร์ดเป็นนาฬิกาจับเวลาโดยมีคีย์ควบคุมการทำงานดังนี้

C - เริ่มจับเวลา

R - รีเซตเวลาใหม่

S - หยุดเวลา

Q - ออกจากโปรแกรม

โดยที่โปรแกรมดังกล่าวจะทำงานได้เฉพาะโหมด LCD เท่านั้น

- T

T [address]

เป็นคำสั่งทำงานเช่นเดียวกับคำสั่ง P คือให้หน่วยประมวลผลกลางกระโดดไปทำงานของผู้ใช้ทีละคำสั่ง แต่ถ้าคำสั่งของผู้ใช้ที่พบเป็น "ACALL" หรือ "LCALL" ก็จะเข้าไปในโปรแกรมย่อยของผู้ใช้เลย จะไม่รอให้กลับออกมาก่อนแล้วค่อยเข้ามอนิเตอร์เหมือนคำสั่ง P

- TIME

TIME [hh:mm[:ss]]

เป็นคำสั่งที่แสดง , เปลี่ยน เวลาในปัจจุบัน โดยอ่านเขียนข้อมูล, คำสั่ง ลงบนรีจิสเตอร์ของชิปฐานเวลาจริงดังแสดงในตารางที่ 16 เช่น

TIME - แสดงเวลาปัจจุบัน

TIME 12:00 - ตั้งเวลาปัจจุบันเป็น 12:00 น.

- U

U [address][,address]

เป็นคำสั่งแปลคำสั่งภาษาเครื่องเป็นภาษาแอสเซมบลี (UNASSEMBLE) ในช่วงของหน่วยความจำชนิดโปรแกรมที่กำหนดเช่น

U - แปลคำสั่งต่อจากเดิม

U 8000h,8100h - แปลคำสั่งจากตำแหน่ง 8000h-8100h (เฉพาะเทอร์มินอลโหมด)

U ,8100h - แปลคำสั่งต่อจากเดิมจนถึงตำแหน่ง 8100h (เฉพาะเทอร์มินอลโหมด)

หมายเหตุ สำหรับโหมดการแสดงผลที่เป็น LCD หลังจากที่เรียกคำสั่งดังกล่าวแล้ว จะยังไม่จบการทำงานทันทีเหมือนเทอร์มินอลโหมด แต่จะมีคีย์ควบคุมการทำงานดังนี้

SPACE - แปลคำสั่งถัดไป

MON - จบการทำงาน

- UPLOAD

UPLOAD <address,address> <-a | -b | -h>

เป็นคำสั่งที่ส่งข้อมูลจากหน่วยความจำชนิดข้อมูลไปยังคอมพิวเตอร์ภายนอก ผ่านพอร์ตอนุกรมในรูปของรหัส แอสกี ,ไบนารี , และ HEX เช่น

upload 4000h,7ffffh -a -> ส่งข้อมูลในหน่วยความจำชนิดข้อมูลในตำแหน่ง 4000h-7ffffh ออกไปในรูปของรหัสแอสกี

upload 8000h,9000h -h -> ส่งข้อมูลในหน่วยความจำชนิดข้อมูลใน ตำแหน่ง 8000h-9000h ออกไปในรูปของ Intel Hex Format

upload 8000h,9000h -b -> ส่งข้อมูลในหน่วยความจำชนิดข้อมูลใน ตำแหน่ง 8000h-9000h ออกไปในรูปของ ไบนารี

หมายเหตุ สำหรับการอัพโหลดแบบแอสกี หากพบตัวอักษร 0Dh (Line Feed) ซึ่งแทนสัญญลักษณ์จบบรรทัดในเอดิเตอร์และไม่มีตัวอักษร 0Ah ตามมา ก็จะส่งตัวอักษร 0Ah ออกไปด้วย

- VEC

VEC <number> [= address]

เป็นคำสั่งที่แสดง, หรือกำหนด ตำแหน่งอินเตอร์รัปเวกเตอร์ใหม่โดยความหมายของอินเตอร์รัปเวกเตอร์แต่ละหมายเลขเป็นไปตามตารางที่ 18 เช่น

vec 0 -> แสดงอินเตอร์รัปเวกเตอร์ของการเกิดอินเตอร์รัปภายนอก 0

vec 2 = 8000h -> กำหนดอินเตอร์รัปเวกเตอร์ให้กับการเกิดอินเตอร์รัปภายนอกที่ส่งสัญญาณผ่านขา INT1

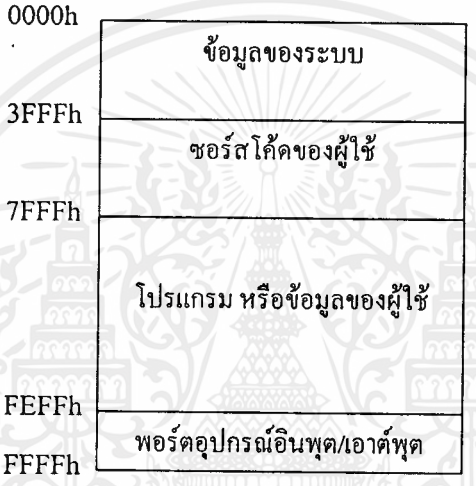
หมายเลข	สัญญาณ	ตำแหน่งเริ่มต้น	ความหมาย
0	IE0	8003h	อินเตอร์รัปภายนอก 0
1	TF0	800Bh	วงจรนับ/จับเวลา 0
2	IE1	xxxx	อินเตอร์รัปภายนอก 1 (ปกติระบบสงวนไว้ใช้ในการหยุดการทำงาน Break))
3	TF1	801Bh	วงจรนับ/จับเวลา 1
4	RI หรือ TI	8023h	วงจรรับ/ส่งข้อมูลอนุกรม
5	TF2	802bh	วงจรนับ/จับเวลา 2

ตารางที่ 2.5 ตารางแสดงความหมายของหมายเลขอินเตอร์รัปเวกเตอร์

2.2.3 การใช้งานหน่วยความจำ

ในกรใช้งานซอฟแวร์ ผู้ใช้จะต้องระวังการเข้าไปแก้ไข หรือเปลี่ยนแปลง ข้อมูลในหน่วยความจำที่ระบบสงวนเอาไว้ มิฉะนั้นการทำงานของโปรแกรมบนบอร์ดอาจไม่ถูกต้องได้ ผู้ใช้สามารถจะนำซอร์สโค้ด (Source Code) ของผู้ใช้ที่เป็นแอสกีไปเก็บไว้ในตำแหน่ง 4000h-7FFFh และเก็บข้อมูลหรือโปรแกรมของตัวเองที่เป็นไบนารี (Binary) ไว้ในตำแหน่ง 8000h-FEFFh

และจากภาพที่ 2.5 ในระหว่างที่โปรแกรมของผู้ใช้ทำงานอยู่ ก็ไม่ควรเข้าไปใช้รีจิสเตอร์ภายในของหน่วยประมวลผลกลางในตำแหน่ง 2Fh-46h เช่นกัน



ภาพที่ 2.4 ภาพแสดงการใช้งานหน่วยความจำชนิดข้อมูล

ภาพที่ 2.5 ภาพแสดงการใช้งานรีจิสเตอร์ภายในของหน่วยประมวลผลกลาง

สรุป

Advance Single Board มีส่วนประกอบทางด้านฮาร์ดแวร์และซอร์ฟแวร์ที่เหมาะสม โดยนำคุณสมบัติเด่นของ Single Board ในท้องตลาดมารวมกัน

บทที่ 3
การทดสอบการทำงานของเครื่องซิงเกิลบอร์ด

การตรวจสอบและทดสอบการทำงานทั้งหมดทางด้านฮาร์ดแวร์และซอร์ฟแวร์(ทดสอบชุดคำสั่งทั้งหมด)ของแอดวานส์ซิงเกิลบอร์ดที่ได้ออกแบบและสร้างไว้แล้ว โดยได้มีการทดสอบการทำงานในแต่ละส่วน และจัดทำใบการทดลองขึ้นมาเพื่อใช้ในการทดสอบการทำงานทั้งหมด

3.1 การทดสอบทางด้านฮาร์ดแวร์

3.1.1 วงจรควบคุมหลัก

ขั้นตอนการทดลอง

- เขียนโปรแกรมส่งค่า #055h ออกที่พอร์ต 1 ของหน่วยประมวลผลกลาง และสังเกตผลโดยใช้ลอจิกโพรบวัดสัญญาณที่ขา 1-7 ของ U1 (8031)
- ทดลองส่งค่าอื่นๆออกมาที่พอร์ต 1

ผลการทดลอง

- สัญญาณที่พอร์ต 1 เป็นไปตามค่าที่ส่งออกมา

สรุปผลการทดลอง

- หน่วยประมวลผลกลางบนซิงเกิลบอร์ดสามารถทำงานตามโปรแกรมของผู้ใช้ได้อย่างถูกต้อง

3.1.2 ส่วนแสดงผล

ขั้นตอนการทดลอง

- ทดลองเขียนโปรแกรมส่งคำสั่งต่อไปนี้ไปยังอุปกรณ์แสดงผลแบบผลึกเหลว LCD ที่พอร์ต FF0Ch เพื่อกำหนดโหมดการทำงานให้ LCD
    - #00111000b ;function set
    - #00000110b ;entry mode set
    - #00001111b ;display on,cursor on,blink on
    - #00000001b ;clear screen
    - #00000010b ;cursor goto home

- ให้โปรแกรมส่งรหัสแอสกีของ 'a' ไปที่พอร์ต FF0Fh เพื่อส่งข้อมูลที่จะแสดงบนจอไปที่ LCD
- ให้โปรแกรมลองส่งตัวอักษรตัวอื่นตามไปบ้าง

ผลการทดลอง

- บนจอ LCD สามารถแสดงตัวอักษรตามรหัสแอสกีที่ส่งไปให้ได้

สรุปผลการทดลอง

- LCD บนบอร์ดสามารถทำงานได้อย่างถูกต้อง

3.1.3 พอร์ตใช้งานทั่วไป 8255

ขั้นตอนการทดลอง

- ทดลองเขียนโปรแกรมส่งค่า #82h ไปยังพอร์ต D ของ U6 ที่ตำแหน่ง FF03h เพื่อให้พอร์ต A เป็นเอาต์พุตพอร์ต
- ส่งค่า #055h ไปยังพอร์ต A ของ U6 ที่ตำแหน่ง FF00h
- ใช้ลอจิกโพรบวัดสัญญาณที่พอร์ต A
- ทดลองกับพอร์ต A ของ U7 ในทำนองเดียวกัน

ผลการทดลอง

- พอร์ต A สามารถส่งค่าออกมาได้ตามที่โปรแกรมต้องการ

สรุปผลการทดลอง

- 8255 บนบอร์ดสามารถทำงานได้อย่างถูกต้อง

3.1.4 ส่วนคีย์บอร์ด

ขั้นตอนการทดลอง

- ทดลองเขียนโปรแกรมหน่วงเวลา 1/10 วินาที แล้วส่งคำสั่ง #82h ไปให้ U6 ที่ตำแหน่ง FF03h เพื่อให้พอร์ต B เป็นอินพุต
- ส่งค่า #0 ไปที่พอร์ต FF08h เพื่อเลือกคีย์ในกลุ่มที่ 1
- ให้โปรแกรมอ่านค่าจากพอร์ต B ของ U6 และกำจัด (Mask) 2 บิตบนทิ้งไป จากนั้นนำค่าดังกล่าวไปแสดงที่จอ LCD ตลอดเวลาในรูปของเลขฐาน 2
- ทดลองกดคีย์ในกลุ่มที่ 1 และสังเกตผลบน LCD
- ทดลองกดคีย์ในกลุ่มอื่นๆ และสังเกตผล

ผลการทดลอง

- หากกดคีย์ในกลุ่มที่ 1 ตั้งแต่ 1 คีย์ขึ้นไปจะมีบิตบางบิตที่ตำแหน่งตรงกันกลายเป็น 0 แต่คีย์ในกลุ่มอื่นจะไม่มีผล

สรุปผลการทดลอง

- หน่วยประมวลผลกลางบนซิงเกิลบอร์ดใช้งานวงจรแสกนคีย์บอร์ดได้ถูกต้อง

3.1.5 ส่วนวงจรสร้างฐานเวลาจริง RTC

ขั้นตอนการทดลอง

- ทดลองเขียนโปรแกรมส่งค่าต่างๆไปยังพอร์ตต่อไปนี้
  - #00000100b -> พอร์ต FF1Fh ;24 ชั่วโมง นาฬิกาเดิน
  - #00000100b -> พอร์ต FF1Eh ;อินเตอร์ทุกๆ 1 วินาที
- ใช้ลอจิกโพรบวัดที่ขา 1 ของชิปดังกล่าวและสังเกตผล
- อ่านค่าจากพอร์ต FF10h และแสดงผลบนจอ LCD ตลอดเวลา และสังเกตผล

ผลการทดลอง

- มีสัญญาณพัลส์เกิดขึ้นทุกๆ 1 วินาที
- บนจอ LCD จะแสดงค่าตัวเลขเพิ่มขึ้นทุกๆ 1 วินาที

สรุปผลการทดลอง

- หน่วยประมวลผลกลางบนซิงเกิลบอร์ดใช้งานชิปฐานเวลาจริงได้อย่างถูกต้อง

3.1.6 ส่วนวงจรอินเตอร์เฟสอาร์เอส 232 (RS-232)

ขั้นตอนการทดลอง

- ต่อ J5 เข้ากับพอร์ตอนุกรมของคอมพิวเตอร์ภายนอกเขียนโปรแกรมกำหนดความเร็วในการรับส่งข้อมูลของซิงเกิลบอร์ดตามโปรแกรมย่อยต่อไปนี้

```
;initialize for serial port
;*****************************************************
  anl   pcon,#01111111b  ;Serial port operation clock is normal
  mov  scon,#52h         ;Serial in Mode 1 ,and enable REN,and
                         ;Set ti for the first sent character
  mov  tmod,#20h         ;Timer 1 in Mode 2
  mov  th1,#0fdh         ;Set baud rate = 9600 bps
  mov  tl1,#0fdh
  setb  tr1              ;Timer 1 run
```

- ทดลองส่งรหัสแอสกีของ 'B' โดยผ่านรีจิสเตอร์ sbuf ดังโปรแกรมย่อยต่อไปนี้ และสังเกตผลบนจอคอมพิวเตอร์

```
  mov  a,#'B'
  jnb  ti,$          ;wait sending is success
  clr  ti
  mov  sbuf,a        ;begin send data
```

- ทดลองให้โปรแกรมบนบอร์ดอ่านค่าจากพอร์ตอนุกรมด้วยโปรแกรมย่อยต่อไปนี้ และแสดงผลบน LCD เมื่อได้รับข้อมูลตลอดเวลา จากนั้นลองกดคีย์บอร์ดของคอมพิวเตอร์ด้วยคีย์ตัวอักษร และสังเกตผลบน LCD

```
ser_get:
;*************************************************************
;This routine will receievc character through serial port
;if no input,then return #0
;output  : acc
;*************************************************************
  clr  a              ;                (1)
  jnb  ri,$+7         ;if not recieve any character
                      ;then return #0       (3)
  clr  ri             ;clear receiver flag   (2)
  mov  a,sbuf         ;read character        (2)
  ret
```

- ทำการทดลองการทำงานแบบ remote โดยสั่ง CTTY จากนั้นทำการสั่งงานจากเครื่องคอมพิวเตอร์ใหม่อีกครั้ง

ผลการทดลอง

- ซิงเกิลบอร์ดสามารถส่งข้อมูลให้ปรากฏบนจอคอมพิวเตอร์ได้
- ซิงเกิลบอร์ดสามารถรับข้อมูลที่ส่งมาจากคอมพิวเตอร์ได้
- ซิงเกิลบอร์ดสามารถทำงานได้ถูกต้อง

สรุปผลการทดลอง

- หน่วยประมวลผลกลางบนซิงเกิลบอร์ดสามารถติดต่อกับคอมพิวเตอร์ภายนอกได้
- สามารถควบคุมการทำงานจากเครื่องคอมพิวเตอร์ได้

3.1.7 ความคงทน

เขียนโปรแกรมแสดงเวลาของเครื่องให้สามารถทำงานได้และแสดงผลที่ LED ตลอดเวลา

ผลการทดลอง

- ซิงเกิลบอร์ดสามารถทำงานได้ตลอด 3 วัน อย่างถูกต้อง

สรุปผลการทดลอง

- ซิงเกิลบอร์ดสามารถทำงานได้ถูกต้องตลอดเวลาแม้จะทำงานเป็นเวลานาน

3.1.8 ระบบเสียง

ขั้นตอนการทดลอง

- จากวงจรในภาพที่ 18 และตารางที่ 10 ให้กำหนดจัมเปอร์เพื่อใช้ลำโพง
- เขียนโปรแกรมกำหนดโหมดให้พอร์ต A ของ U6 เป็นเอาต์พุต
- ส่งค่า '0' และ '1' สลับกันไปผ่านพอร์ต A บิตที่ 7 ลองฟังเสียงจากลำโพง
- ทดลองเปลี่ยนเวลาที่ใช้หน่วงและลองฟังเสียงใหม่อีกครั้ง

ผลการทดลอง

- ซิงเกิลบอร์ดสามารถสร้างเสียงที่ความถี่ต่างๆได้

สรุปผลการทดลอง

- หน่วยประมวลผลกลางบนซิงเกิลบอร์ดสามารถสร้างเสียงที่ความถี่ต่างๆได้

3.1.9 Source Code

- การทดลองโปรแกรมขนาด 500 บรรทัด ลงในเครื่องซิงเกิลบอร์ด
- ทำการแปลงโปรแกรมเพื่อทำการตรวจสอบความถูกต้อง

ผลการทดลอง

- ซิงเกิลบอร์ดสามารถแปลงได้ถูกต้องและทำงานได้

สรุปผลการทดลอง

- ซิงเกิลบอร์ดนี้สามรถทำการแก้ไขโปรแกรมได้สะดวก

3.1.10 Program Code

- การทดลองโดยนำโปรแกรม monitor มา compile ให้เริ่มที่ตำแหน่ง 8000h จากเครื่องคอมพิวเตอร์
- ทำการ download ลงเครื่องซิงเกิลบอร์ดโดยสั่ง

  ซิงเกิลบอร์ดคือ DOWNLOAD 8000h

  เครื่องคอมพิวเตอร์ คือ copy xxx.hex แล้วเรียกคำสั่ง B 8000h

ผลการทดลอง

- เหมือนการทำงานของ monitor ทุกอย่างแต่ไมสามารถใช้ compile ได้ เนื่องจากแอดเดรส 8000h ถูกใช้งานแล้ว เมื่อ compile จะมาซ้อนกัน ทำให้ทำงานไม่ถูกต้อง

สรุปผลการทดลอง

- ซิงเกิลบอร์ดสามารถทำโปรแกรมขนาดใหญ่มาทดลองได้

3.2 การทดสอบการทำงานทางด้านซอฟแวร์

ต่อไปนี้เป็นตัวอย่างไฟล์ที่ใช้ในการทดสอบเอดิเตอร์และแอสเซมเบลอ

```
;TEST.ASM
;It is downloaded to Single Board
 ORG  0A000H

 port_wi equ 0FF0Ch      ;write instruction
 port_wd equ 0FF0Dh      ;write data
 port_ri equ 0FF0Eh      ;read  instruction
 port_rd equ 0FF0Fh      ;read  data


 r0_   EQU   00          ; address of R0 to R7
 r1_   EQU   01
 r2_   EQU   02
 r3_   EQU   03
 r4_   EQU   04
 r5_   EQU   05
 r6_   EQU   06
 r7_   EQU   07


 lcall delay             ;delay time
 lcall lcdclr            ;clear screen
```

```
  mov  dptr,#text1          ;get text address for display
  lcall lcdwritest          ;display the text on LCD
  sjmp  $                   ;stop running

text1: db 'Advance Single Board',0

delay:   push r1_
         push r2_
         mov r1,#100
         mov r2,#10
         djnz r2,$
         djnz r1,$-4    ;to "mov r2,#10" command
         pop  r2_
         pop  r1_
         ret

lcdwi:   push dph              ;write instuction in acc to LCD
         push dpl
         push acc
         mov  dptr,#port_ri
         movx a,@dptr
         jb  acc.7,$-1
         pop acc
         mov dptr,#port_wi
         movx @dptr,a
         pop dpl
         pop dph
         ret
lcdwd:   push dph              ;write data in acc to LCD
        push dpl
```

```
        push acc
        mov  dptr,#port_ri
        movx a,@dptr
        jb  acc.7,$-1
        pop acc
        mov dptr,#port_wd
        movx @dptr,a
        pop dpl
        pop dph
        ret

blinkon:                ;display on ,blink on
  mov a,#00001111b
  acall lcdwi
  ret

lcdclr:  mov a,#01h     ;clear screen (LCD)
        acall lcdwi
        ret

lcdhome:  mov a,#02h    ;move cursor to home
        acall lcdwi
        ret
lcdwritest:             ;display string pointed by dptr on LCD
nextch:  clr a
        movc a,@a+dptr
        jz printst1
        push dph
        push dpl
        acall lcdwd
```

```
        pop  dpl
        pop  dph
        inc  dptr
        sjmp nextch
printst1: ret

  END                ;End of User Program
```

3.2.1 เอดิเตอร์

ขั้นตอนการทดลอง

- หลังจากจ่ายไฟให้กับบอร์ด และเข้าสู่มอนิเตอร์พร้อมท์แล้ว (ปรากฏเครื่องหมาย '-') เรียกคำสั่ง "baud 9600"
- เรียกคำสั่ง "edit"
- ลองใช้คำสั่ง CTRL-KR เพื่อรับไฟล์ข้อมูลจากคอมพิวเตอร์ภายนอก จากนั้นบนจอ LCD จะแสดงข้อความ "Ploase Download ..."
- ต่อคอนเนคเตอร์ J5 ตามรูปที่ 13, 14, 15
- กำหนดความเร็วในการรับส่งข้อมูลของคอมพิวเตอร์ผ่านพอร์ตที่ x ด้วยคำสั่งต่อไปนี้ "mode comx:9600,n,8,1"
- ทดลองส่งไฟล์ "test.asm" ไปให้กับบอร์ดด้วยคำสั่ง "copy test.asm comx"
- เมื่อมีข้อความของไฟล์ "test.asm" ปรากฏบนจอ LCD จึงทดลองใช้คำสั่งในตารางที่ 17 ในการแก้ไขข้อความ

ผลการทดลอง

- เอดิเตอร์บนซิงเกิลบอร์ดสามารถทำงานได้อย่างถูกต้อง

สรุปผลการทดลอง

- ผู้ใช้สามารถใช้ซิงเกิลบอร์ดเพื่อแก้ไขโปรแกรมขนาดเล็กได้โดยไม่ต้องอาศัยคอมพิวเตอร์ภายนอก

3.2.2 แอสเซมเบลอ

ขั้นตอนการทดลอง

- ทำการทดลองเช่นเดียวกับข้อ 4.2.1

- ออกจากเอดิเตอร์ด้วยคีย์ CTRL-QX
- เมื่ออยู่ในมอนิเตอร์พร้อมท์เรียกคำสั่ง "asm" จะปรากฏข้อความแสดงขั้นตอนการแอสเซมเบิล (Assemble) โปรแกรม
- หากไม่มีความผิดพลาดเกิดขึ้น ให้ทดลองรันโปรแกรมที่ได้แปลเป็นรหัสภาษาเครื่องแล้วด้วยคำสั่ง "g 8000h" จากมอนิเตอร์พร้อมท์

ผลการทดลอง

- ซิงเกิลบอร์ดสามารถทำงานโปรแกรมที่ได้มาจากแปลด้วยแอสเซมเบลอ บนซิงเกิลบอร์ด ได้อย่างถูกต้อง คือสามารถแสดงข้อความออก่จอ LCD คือ "Advance Single Board" ได้ถูกต้อง

สรุปผลการทดลอง

- ผู้ใช้สามารถใช้ซิงเกิลบอร์ดเพื่อแก้ไขโปรแกรมขนาดเล็กและแปลเป็นรหัสภาษาเครื่องได้ โดยไม่ต้องอาศัยคอมพิวเตอร์ภายนอกได้ (Stand Alone)

3.2.3 ชุดคำสั่งทั้งหมดของ MCS-51

ได้จัดทำใบการทดลองประกอบการทดสอบการทำงานของเครื่อง โดยเน้นทางด้านคำสั่งต่างๆที่ใช้งาน ว่าสามารถใช้งานได้ถูกต้องหรือไม่ โดยมีทั้งหมด 19 การทดลองดังนี้คือ

การทดลองที่ 1 เรื่องคำสั่งการโอนย้ายข้อมูล
การทดลองที่ 2 คำสั่งทางคณิตศาสตร์และลอจิก
การทดลองที่ 3 คำสั่งการกระโดด
การทดลองที่ 4 การบวกเลขฐานสอง
การทดลองที่ 5 คำสั่งเกี่ยวกับบิท
การทดลองที่ 6 การเลื่อนและการหมุนข้อมูล
การทดลองที่ 7 แสตกและรูทีน
การทดลองที่ 8 การคูณเลขฐานสอง
การทดลองที่ 9 การหารเลขฐานสอง
การทดลองที่ 10 การเปลี่ยนเลขฐานสองเป็นรหัส BCD
การทดลองที่ 11 การเปลี่ยนรหัส BCD เป็นเลขฐานสอง
การทดลองที่ 12 การต่อไมโครโปรเซสเซอร์กับอุปกรณ์เอาต์พุท
การทดลองที่ 13 การแสดงผลข้อมูลด้วย LED 7 SEGMENT
การทดลองที่ 14 การรับข้อมูลจากอุปกรณ์ภายนอก
การทดลองที่ 15 การอินเทอร์รัพท์

การทดลองที่ 16 การรับส่งข้อมูลแบบอนุกรม

การทดลองที่ 17 การเปลี่ยนสัญญาณดิจิตอลเป็นอนาลอก

การทดลองที่ 18 การเปลี่ยนสัญญาณอนาลอกเป็นดิจิตอล

การทดลองที่ 19 การเชื่อมต่อกับเครื่องพิมพ์

สรุปผลการทดสอบ

จากการทดสอบการทำงานของเครื่อง Advance Single Board ทั้งทางด้านฮาร์ดแวร์และซอร์ฟแวร์ เครื่อง Advance Single Board สามารถทำงานได้ถูกต้องตามคุณสมบัติที่มีอยู่ในท้องตลาด

บทที่ 4
ความรู้เกี่ยวกับโปรแกรม Authorware

4.1 Macromedia Authorware

Authorware นับเป็นวิวัฒนาการอีกชั้นหนึ่งของโปรแกรมประเภท Authoring Systerm ที่ใช้สำหรับการสร้างแอพพลิเคชั่นในระบบมัลติมิเดียด้วยการออกแบบการทำงานในลักษณะแผนภูมิที่ทำให้แม้แต่ผู้ที่ไม่ได้เป็นโปรแกรมเมอร์ก็สามารถที่จะสร้างงานขึ้นมาได้ โดยไม่ต้องกังวลเกี่ยวกับภาษาโปรแกรม Authorware มีคุณสมบัติสามประการที่สนับสนุนงาน สร้าง ออกแบบแอพพลิเคชั่น รวมทั้งการกระจาย ไปยังผู้ใช้ได้แก่

Object Authoring ด้วยการออกแบบโปรแกรมด้วยเทคนิค Object Authoring ทำให้ผู้ใช้ที่ไม่คุ้นเคยกับการออกแบบโปรแกรม หรือผู้ที่มีประสบการณ์มาแล้วก็ตามสามารถทุ่มเท ความสนใจไปยังรายละเอียดของเนื้อหา และวิธีการโต้ตอบของผู้ใช้โดยไม่ต้องกังวลเกี่ยวกับการเขียนโปรแกรม การใช้สัญลักษณ์ (Icon) แทนคำสั่งทำให้ผู้ใช้สามารถสร้างโปรแกรมที่มีคุณภาพสูงได้อย่างง่ายดาย

Multimedia Tools ในโปรแกรม Authorware ประกอบด้วยเครื่องไม้เครื่องมือด้านมัลติมิเดียอย่างพร้อมมูล ทำให้ผู้ใช้สามารถสร้างแอพพลิเคชั่น ที่ประกอบด้วย ข้อความ รูปภาพ เสียง ภาพเคลื่อนไหว และวิดีโอ เข้าด้วยกัน ทำให้เป็นแอพพลิเคชั่นที่มีประสิทธิภาพที่จะใช้ในการเรียนการสอน การอ้างอิง จำลองการทำงานในการนำเสนอสินค้า และการโฆษณา

การออกแบบโปรแกรมให้สามารถใช้ได้หลายระบบ ทำให้ผู้ใช้ไม่ว่าจะเป็นเครื่อง Macintosh หรือภายใต้ระบบ Microsoft Windows ที่อยู่บนเครื่อง PC มีการทำงานที่เหมือนกัน และสามารถที่จะติดต่อไปยังทรัพยากรภายนอกระบบไม่ว่าการใช้ระบบฐานข้อมูลหรือระบบคอมพิวเตอร์เครือข่าย คำสั่งในการทำงานต่างๆ ไม่ว่าเป็นเครื่อง Macintosh หรือเวอร์ชั่นที่ทำงานภายใต้ Windows ไม่ได้มีความแตกต่างกันมากนัก ยกเว้นในส่วนของมัลติมิเดีย และการทำงานของโปรแกรมในสภาพแวดล้อมที่ต่างกัน

4.2 Object Authoring™

กล่าวได้ว่าส่วนหนึ่งที่ทำให้โปรแกรม Authorware เป็นโปรแกรมที่ใช้ง่าย ก็คือการที่ออกแบบคำสั่งต่างๆ อยู่ในรูปของสัญลักษณ์ (Icon) การสร้างโปรแกรมทำได้ด้วยการวางไอคอน เรียงไว้บนเส้นโฟลว์งาน ด้วยวิธีนี้จึงไม่มีความจำเป็นต้องเรียนรู้การใช้คำสั่งเป็นลักษณะภาษาโปรแกรม

การทำงานด้วยการใช้สัญลักษณ์

คำสั่งใน Authorware ถูกออกแบบไว้ในลักษณะที่เป็นสัญลักษณ์จำนวนสิบเอ็ดตัว ซึ่งสัญลักษณ์แต่ละตัวจะใช้แทนคำสั่งในการพัฒนาแอพพลิเคชั่นได้อย่างสมบูรณ์ อีกทั้งมีความง่ายในการใช้งานเมื่อเลือกสัญลักษณ์ หรือ

คำสั่งใดคำสั่งหนึ่ง โปรแกรมจะอสดงรายละเอียดหรทอคำสั่งเพิ่มเติมที่จำเป็นในการทำงานของสัญลักษณ์นั้นๆ ให้เลือกไม่ว่าเป็นคำสั่งเกี่ยวข้องกับลอจิกของโปรแกรม หรือคำสั่งในที่ทำงานเป็นมัลติมิเดีย

4.3 วิธีการพัฒนาโปรแกรม

ลักษณะการทำงานประกอบด้วยไอคอน ที่จะเรียงลงบนเส้นโฟลว์งาน เป็นการกำหนดลอจิกในการทำงานโปรแกรมนอกจากนี้ยังมีคำสั่งที่เป็นเมนูเพื่อกำหนดรายละเอียดของการทำงาน สามารถกำหนดรายละเอียดของโปรแกรม เลือกลักษณะการทำงานของโปรแกรมว่าให้ทำต่อจากที่ค้างไว้ หรือเริ่มต้นใหม่ทุกครั้งที่เรียกรวมทั้งสามารถกำหนดชื่อของโปรแกรม

คำสั่ง "Try It" ทำให้ผู้ที่พัฒนาโปรแกรมสามารถทดสอบโปรแกรมได้โดยง่าย คำสั่ง Start Flag, Stop Flag ช่วยให้การทดสอบและแก้ไขโปรแกรมในส่วนต่างๆ ได้รวมทั้งการเลือกทดสอบโปรแกรมแต่ละส่วน

คำสั่ง Package ช่วยในการจัดเตรียมแอพพลิเคชั่นสำหรับ ผ้ใช้โดยไม่ต้องติดตัว System ไปด้วยทำให้การกระจาย แอพพลิเคชั่นเป็นไปอย่างสะดวก หรือในกรณีที่ต้องการลดขนาดของแอพพลิเคชั่นลงก็สามารถทำได้ แต่ในการเรียกใช้งานต้องเรียกผ่าน System ของ Authorware เอง

4.4 ลักษณะที่เอื้ออำนวยในการทำงานของโปรแกรม

1. สามารถทดสอบ และแก้ไขโปรแกรมได้ในเวลาเดียวกัน
2. ความสามารถในการแก้ไขเปลี่ยนแปลงลอจิกของโปรแกรมได้โดยตรง
3. สามารถกำหนดวิธีการโต้ตอบกับผู้ใช้ได้ถึงสิบวิธี
4. คุณสมบัติที่เอื้ออำนวยอื่นๆ ได้การผสมผสานสื่อต่างๆ เข้าด้วยกัน

4.5 Library สนับสนุนการทำงาน

1. มี Library อันได้แก่ ภาพเคลื่อนไหว ภาพกราฟฟิกส์ ภาพจากวิดีโอ เสียงและอื่นๆ
2. มีโฟล์โครงสร้างที่ผู้ใช้สามารถนำไปใช้งานได้ ประกอบด้วยตัวอย่างโปรแกรมอย่าง เช่นระบบ Pull Down Menu สมุดโน๊ต โปรแกรมบันทึกการทำงาน ขั้นตอนในการทำงาน ข้อแนะนำทางเทคนิค และยูติลิตี้
3. ผู้ใช้สามารถสร้างโมเดลการทำงานที่สามารถนำกลับไปใช้ได้

4.6 ตัวแปรและฟังก์ชัน

และฟังก์ชั้นสนับสนุนการทำงานมากกว่า 200 ตัว เป็นการเพิ่มความสามารถในการ เก็บค่า แก้ไข หรือแสดงข้อมูลต่างๆ รวมทั้งการควบคุมการทำงานของโปรแกรม ซึ่งมีข้อดีในการทำงานได้แก่

1. ความสามารถในการใช้ตัวแปรทำให้สามารถติดตามการใช้โปรแกรมและเรียกใช้ฟังก์ชั่นการทำงานที่เหมาะสมเพื่อตอบสนองการทำงานของผู้ใช้

2. มีคำสั่งสำหรับดูรายละเอียดของฟังก์ชั่นและตัวแปรรวมทั้งสามารถคัดลอกตัวแปรและฟังก์ชั่นไปยัง Calcuation Icon, Option Slot และ Presentation Windows

3. สามารถควบคุมฟอร์แมตการแสดงผลของตัวแปรได้ ช่วยให้สามารถทดสอบระดับความรู้พื้นฐานของผู้ใช้ได้

4.7 การทำเอกสารกำกับโปรแกรมโดยอัตโนมัติ

1. ทำดัชนีของโปรแกรมโดยมีไอคอนหรือไม่มีก็ได้
2. พิมพ์ Presentation Windows ที่มีอยู่ทั้งหมดออกมาได้
3. ทำตารางอ้างอิงการใช้ตัวแปร

4.8 Multimedia Tools

Authorware มีอุปกรณ์เครื่องไม้เครื่องมือในการที่จะสร้างแอพพลิเคชั่นที่เป็นมัลติมิเดียให้อย่างสมบูรณ์ รวมทั้งความสามารถในการเรียกใช้ และแก้ไข Media ที่สร้างมาจากโปรแกรมอื่น

ข้อความ

- สามารถใช้ตัวอักษรหลายแบบผสมกันได้ รวมทั้งสีและขนาด
- สามารถกำหนดตัวอักษรเป็น Outline, เงา, ตัวเอียง และขีดเส้นใต้
- ฟอร์มเมตข้อความให้มีการตัดคำ ตั้งแท็ปทั้งข้อความ และตัวเลข รวมทั้งกำหนดกรอบ
- จัดคำให้ชิดซ้าย, ขวา หรืออยู่กลางได้
- สามารถใช้ตัวอักษรมาตรฐานของวินโดว์

กราฟฟิกส์

มีคำสั่งในการวาดรูปวงกลม วงรี สี่เหลี่ยมและลากเส้น รวมทั้งแสดงเส้นตาราง

- คำสั่งลากเส้น สามารถลากเส้นตั้ง, เส้นนอน, เส้นเอียง 45 องศา รวมทั้งการใส่ลูกศร และกำหนดความหนาของเส้นได้ 5 ระดับ
- สามารถกำหนด Fill Pattem ได้ทั้งหมด 36 แบบ
- กำหนดการแสดงผลของภาพได้เป็นชั้น สามารถรวมภาพเข้าและแก้ไขภาพเป็นกลุ่มได้
- สามารถของดูภาพก่อนที่จะนำเข้ามาใช้ได้
- ไฟล์กราฟฟิกส์ที่จะนำมาใช้ทั้งที่เป็น TIF, PIC, PNT, WMF, EPS BMP, DIB, RLE, PCX, PICT และ Paint ของเครื่อง Macintosh รวมทั้ง Windows Meta File

เสียง

- ควบคุมการเล่นซ้ำ เริ่ม และหยุดได้
- สามารถเล่นไฟล์ PCM ของ Macintosh ไฟล์ WAV ของวินโดวส์ และเล่น MIDI โดยผ่าน Microsoft's Multimedia Extentions
- สามารถเรียกใช้ไฟล์เสียงของ Macintosh โดยผ่านโปรแกรม Sound Wave หรือ Macromedia's SoundEdit
- การใส่เสียงให้กับโปรแกรมต้องมี Sound Card ที่เล่นภายใต้วินโดว์ส

Animation

- กำหนดทิศทางในการเคลื่อนของวัตถุได้หลายแบบเป็น Scaled Path, Fixed Destination, Fixed Path, Linear Scale และ Scale X/Y
- กำหนดทิศทาง, เวลา และความเร็วได้
- ควบคุมจำนวนเฟรม, ความเร็ว และจำนวนรอบของการเล่น Movie File ได้
- กำหนดชั้นในการเคลื่อนที่ของวัตถุได้ในกรณีที่มีวัตถุมากกว่าหนึ่ง เคลื่อนที่มาอยู่ในตำแหน่งที่ซ้อนกัน

Video

- สามารถเล่นได้ทั้ง Still และ Motion Video
- แสดงผลวีดีโอเต็มจอได้
- สามารถเปลี่ยนขนาด ย้ายวินโดวส์ได้
- ควบคุมการเล่นและหยุดภาพได้
- เลือกเฟรมได้
- ปรับความเร็วในการเล่นได้
- ควบคุมสัญญาณออดิโอได้สองแชลแนลแยกจากช่องสัญญาณวีดีโอ
- ผู้ใช้สามารถควบคุมวีดีโอจากจอภาพได้ *ต้องมี Video Card ที่ทำงานภายใต้ Windows ได้

Effects

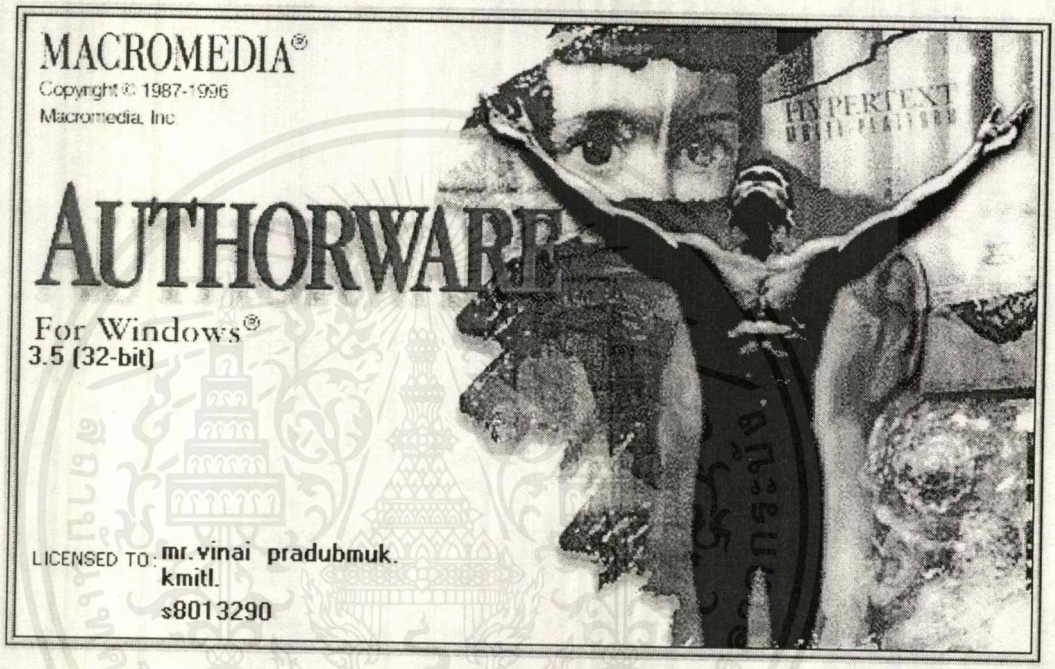
- ควบคุมการเล่นวีดีโอ เสียง และ Animation ได้เป็น Concurrent, perpetual และ Wait until done
- สามารถใช้สีเป็น 4 สี หรือ 8 บิทได้

- แสดงผลข้อความ และกราฟฟิกส์ได้เป็น Opaque, Transparent, Inverse, matted และ Erase มี Special Effects สำหรับแสดงผลหรือลบกราฟฟิกส์ได้หลายแบบ

4.9 การเริ่มเข้าโปรแกรม Authorware

1.เข้าโปรแกรม windows

2.ดับเบิลคลิกที่กลุ่มไอคอนของ Authorware



4.10 สัญลักษณ์ที่ใช้ในโปรแกรม Authorware

Display Icon

เป็นคำสั่งสำหรับทำงานกราฟฟิกส์ในคำสั่งนี้จะมีคำสั่งที่เกี่ยวข้องกับการวาดรูป การอ่านแฟ้มข้อมูลกราฟฟิกส์ จากภายนอกเข้ามา รวมทั้งการแสดงภาพ ข้อความ โดยมี Special effect ต่างๆ

Animation Icon

เป็นคำสั่งทำภาพเคลื่อนไหว ด้วยการกำหนดภาพที่จะเคลื่อนที่ปลายทาง ความเร็ว จำนวนรอบที่จะแสดงภาพ

Erase Icon

ใช้สำหรับลบภาพ หรือข้อความออกจากจอ โดยสามารถกำหนด Special effect ได้

Wait Icon

ใช้หยุดการทำงานของโปรแกรม อาจเป็นการหยุดโดยกำหนดระยะเวลา หรือหยุดจนกว่าผู้ใช้จะให้ทำงานต่อ

Decision Icon

ประกอบด้วยคำสั่งควบคุมการทำงาน ไม่ว่าเป็นการทำงานตามลำดับขั้น การทำงานแบบสุ่ม หรือการทำงานโดยการกำหนดขั้นตอนด้วยค่าของตัวแปร

Interaction Icon

เป็นคำสั่งที่ใช้ในการกำหนดวิธีการติดต่อกับผู้ใช้ ซึ่งอาจกำหนดเป็น ปุ่มกด, เป็น Pulldown Menu และอีกหลายแบบ รวมทั้งกำหนดทิศทางการทำงานของโฟลว์งาน

Calculation Icon

เมื่อต้องการที่จะใส่ตัวแปรเพื่อควบคุมการทำงานของโปรแกรม ใช้ฟังก์ชันพิเศษเรียกใช้โปรแกรมภายนอก หรือเรียกไปยังแอพพลิเคชั่นอื่น

Map Icon

เป็นคำสั่งควบคุมลอจิกการทำงานของโปรแกรม สามารถทำงานในลักษณะโครงสร้างที่ซับซ้อนมากกว่าหนึ่งระดับด้วยการทำงานร่วมกับคำสั่งอื่นๆ

Start Icon

เป็นไอคอนธงเริ่มต้น สำหรับต้องการ Run โปรแกรมเป็นส่วนๆ โดยใช้คำสั่ง Run from flag ในส่วนของ Try it บน Menu bar

Stop Icon

เป็นไอคอนธงหยุด สำหรับการ Run โปรแกรมเป็นบางส่วน โดยใช้ร่วมกับธง Start

Movie Icon

เป็นคำสั่งควบคุมการเรียกไฟล์ที่เป็น Animation จากภายนอกเข้ามาใช้

Sound Icon

ใช้เรียกแฟ้มข้อมูลเสียงเข้ามาเพื่อใช้งาน

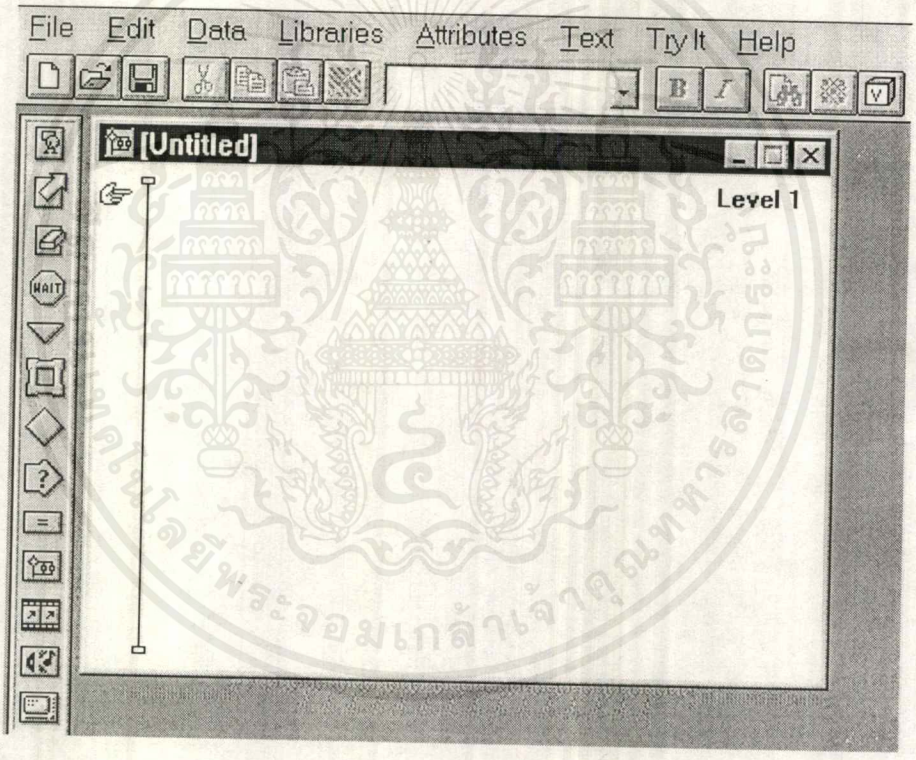
Video Icon

ควบคุมการเล่นวีดีโอ จากโปรแกรม

4.11 การสร้าง Application ใหม่

1.เลือกคำสั่ง New จาก Menu File

2.พิมพ์ชื่อ File ใหม่ที่ต้องการแล้วตอบ OK จะได้ดังภาพ



Display Icon

วัตถุประสงค์ของ Display

1.สร้างกราฟฟิกส์แสดงรายละเอียด อธิบายขั้นตอนสำคัญต่างๆ

2.นำเข้าข้อความหรือรูปภาพกราฟฟิกส์จากโปรแกรม หรืออุปกรณ์อื่นๆ

3.กำหนดตำแหน่งในการแสดงข้อมูลบนจอภาพ

4.แสดงข้อความคงที่หรือเปลี่ยนแปลงบนจอภาพ กำหนดแบบตัวอักษรชนิดของฟอนต์และขนาดต่างๆ

5.กำหนดมาตราส่วนของข้อความ และกราฟฟิกส์โโยอัตโนมัติ

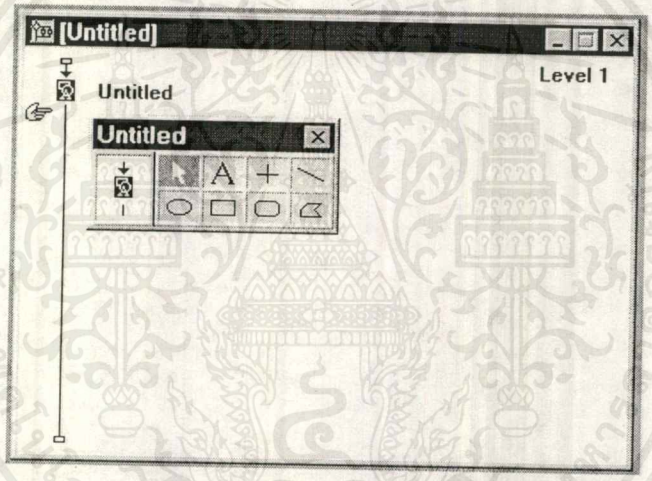
6.กำหนดกริดเพื่อสะดวกในการจัดข้อความและกราฟฟิกส์บนจอ

7.เคลื่อนย้ายข้อความ และกราฟฟิกส์ไปยังตำแหน่งที่ต้องการบนจอ

การเปิด Display Icon

1.ลาก Display Icon จาก Icon Palatte มาไว้บน Flowline ในตำแหน่งที่ต้องการ

2.ดับเบิลคลิกที่ Display Icon บน Flowline



Toolbox คือกล่องเครื่องมือที่ใช้สร้างและแก้ไขข้อความหรือกราฟฟิกส์

Icon Type คือ Icon ที่กำลังเปิดอยู่ในขณะนั้น

Title คือ ชื่อข้อความหรือรูปภาพที่กำลังอยู่ในขณะนั้น

Editing Handles คือ ตัวแสดงขอบเขตของภาพหรือข้อความที่ถูกเลือก

การแก้ไข Display Icon

1.ถ้า Presentation กำลังทำงานอยู่ให้ดับเบิลที่ภาพหรือข้อความที่ต้องการ หรือถ้า Presentation ไม่ได้กำลังทำงานให้คลิกที่ภาพหรือข้อความที่ต้องการ

2.จะปรากฏ Editing Handies

3.แก้ไขให้เหมาะสมตามต้องการ

การใช้ Multimedia Icon

การพัฒนา Authorware ให้สามารถควบคุมอุปกรณ์ Multimedia ได้ ทำให้ผลงานออกมามีประสิทธิภาพสูง การสร้างและนำไปใช้สามารถทำได้โดยง่ายซึ่งมี Icon หลักที่สำคัญสำหรับการทำงานในลักษณะของ Multimedia อยู่ 3 Icon ได้แก่ Sound, Movie และ Video

การใช้ Sound Icon

การออกแบบผลงานใดๆที่ต้องการให้มีเสียงประกอบ ก็สามารถทำได้โดยง่ายไม่ว่าจะเป็นเสียงพูด เสียงดนตรี ที่บันทึกไว้ในลักษณะของไฟล์สกุล WAV (Wave Sound File) สิ่งที่จำเป็นเมื่อใช้ Sound Icon

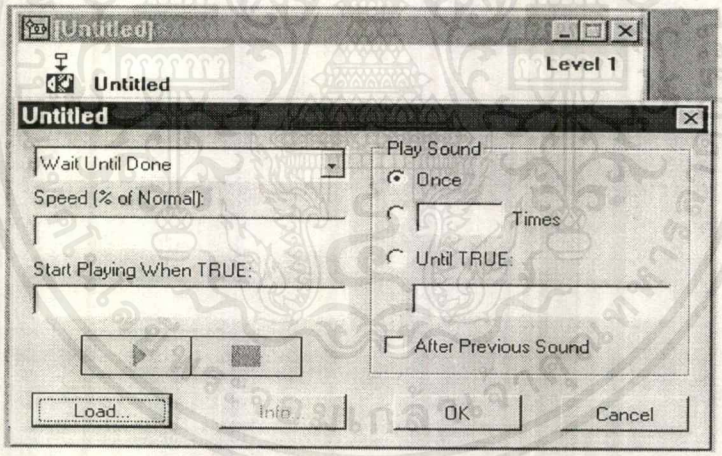
1.Sound Card เป็นอุปกรณ์ที่ต้องมีและจะต้องเป็น Card ที่สนับสนุนโปรแกรม Authorware ด้วย โดยทั่วไปถ้าอยู่ในมาตรฐาน MPC จะใช้ได้

2.Sound Driver ซึ่งจะติดตั้งเพื่อเชื่อมการทำงานไปยัง Sound Card

ขั้นตอนการใช้ Sound Icon

1.เปิด Sound Icon ที่ Flowline

2.ดับเบอลคลิกที่ Sound Icon จะมีรายละเอียดดังภาพ



3.เลือก File สกุล WAV ที่ต้องการ

4.กำหนดรายละเอียดต่างๆ ใน Dialog Box

- Playปุ่มเริ่มแสดงเสียงจากไฟล์ที่เปิด
- Stop ปุ่มหยุดหารแสดงเสียง
- Concurrencyทางเลือกในการแสดงว่าจะให้ต่อเนื่อง
- Speed กำหนดเปอร์เซ็นต์ของความเร็วเสียงปกติ
- Start Playing When True เริ่มแสดงเสียงตามเงื่อนไข

- Play Sound กำหนดจำนวนครั้ง
- Load ปุ่ม Load ให้เปิด File สกุล WAV ใหม่

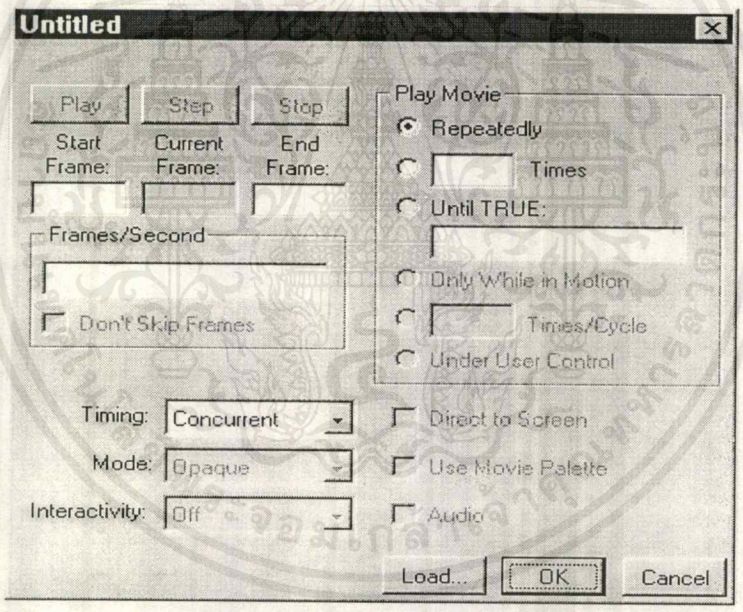
การใช้ Movies Icon

ใน Authorware ไม่สามารถสร้างไฟล์สกุล MOV หรือ Movie ได้ สามารถนำไฟล์ประเภทนี้จากโปรแกรมอื่นๆ มาใช้ได้เช่น Macintosh Authorware Movie Editor, Macromind Director, Studio/1 ซึ่งต้องศึกษาในรายละเอียดของโปรแกรมเหล่านั้น ในการทำ Movie ได้ บน Authorware Prefessional จะมีไฟล์ประเภท MOV มาให้เพื่อใช้เป็นตัวอย่างซึ่งอยู่ในไดเร็คตอรีของ Movies สามารถนำมาใช้ได้ทันที

ขั้นตอนการใช้ Movies Icon

1.เปิด Movies Icon ที่ Flowline

2.ดับเบิลคลิกที่ Movies Icon แล้วเปิด File สกุล MOV ในไดเร็คตอรี Movies ที่ต้องการจะมี Dialog Box Authorwareดังภาพ



- Play ปุ่มเริ่มทำงาน
- Step ปุ่มบอกบอกตำแหน่ง Frame ปัจจุบัน
- Stop หยุดการาแสดง
- Start Frame ช่อง Frame เริ่มต้น
- Current Frame ช่องบอก Frame ปัจจุบัน
- End Frame ช่องบอก Frame สุดท้าย
- Frame Per Second เวลาในการแสดงตามจำนวน Frame ที่กำหนด

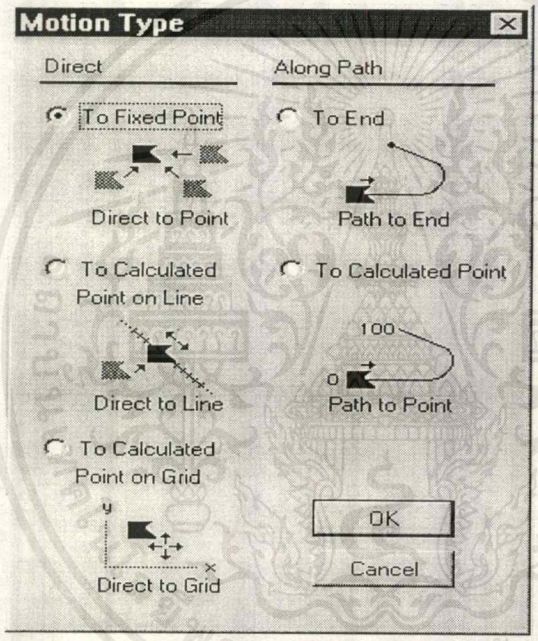
- Layer การซ้อนภาพหลายๆ ภาพตามลำดับของ Layer
- Play Movies จำนวนครั้งในการแสดง
- Load เปิดไฟล์ Movies ใหม่

3.กำหนดรายละเอียดที่ต้องการ

4.ตอบ OK เมื่อทุกอย่างเรียบร้อย

การใช้ Animation Icon

ในโปรแกรม Authorware Professional สามารถสร้างภาพเคลื่อนไหวโดยแบ่งออกเป็น 5 ประเภท ดังนี้



1.Fixed Destination

2.Fixed Path Animation

3.Scaled Path Animation

4.Linear Scale Animation

5.Scaled X-Y Animation

Animation ประเภทนี้จะง่ายและไม่ซับซ้อนเป็นการเคลื่อนย้ายวัตถุจากที่หนึ่งไปยังที่ใหม่ซึ่งเป็นเป้าหมาย

ขั้นตอนการสร้าง

1.เปิด Display Icon ที่ Flowline แล้วสร้างภาพ หรือวัตถุที่ต้องการให้เคลื่อนไหวแล้วลากไปไว้ในตำแหน่งที่ต้องการให้เป็นจุดเริ่มต้น

2.เปิด Animation Icon ที่ Flowline แล้วดับเบิลคลิกที่ Animation Icon จะปรากฏรายละเอียด

- Rate คือทางเลือกสำหรับกำหนดความเร็วในการเคลื่อนที่ ถ้าค่าน้อยจะเคลื่อนที่เร็วถ้าค่ามากจะเคลื่อนที่ช้า
- Replay เป็นปุ่มที่ใช้ทดสอบการเคลื่อนที่ของภาพ
- Change Setup ใช้เมื่อต้องการเปลี่ยนประเภทของ Animation
- Concurrency ทางเลือกในการทำงานต่อมี 2 ทางเลือกดังนี้
- Wait until done แสดงรายละเอียดที่ละ Icon
- Concurrent แสดงหลายๆ Icon พร้อมกัน
- Layer เป็นการกำหนดทางเลือกว่า จะมีอะไรเกิดขึ้นเมื่อวัตถุเคลื่อนที่ไปซ้อนกับวัตถุอื่นๆ

3. เลื่อนเม้าส์ไปที่ภาพที่ต้องการแล้วกดปุ่มเม้าส์ลากภาพไปไว้ในตำแหน่งเป้าหมายที่ต้องการ

4. ทดลองคลิกที่ปุ่ม Replay เพื่อทดสอบการเคลื่อนไหวของภาพจะเห็นว่าภาพจะเคลื่อนที่จากจุดเริ่มต้นเป็นเส้นตรงไปยังจุดสุดท้ายที่ต้องการ

5. กำหนด Layer, Concurrent และ Rate ตามต้องการ

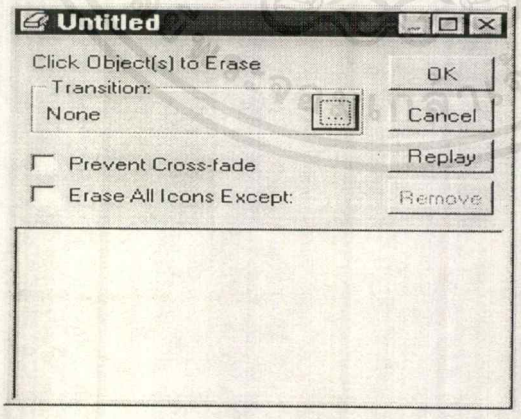
6. ตอบ OK

การใช้ Erase Icon

ใช้สำหรับลบภาพหรือวัตถุที่ไม่ต้องการให้แสดงค้างอยู่บนจน มีขั้นตอนการใช้ดังนี้

1.เปิด Erase Icon ที่ Flowline

2.ดับเบิลคลิกที่ Erase Icon จะปรากฏรายละเอียดดังภาพ



3.กำหนดผลในการลบตามแบบที่ต้องการ โดยเลือกจากรายการในช่อง Effect

4.คลิกที่รูปภาพที่ต้องการลบ สามารถลบรูปหลายๆรูปได้ตามต้องการ

5.เมื่อลบรูปตามต้องการแล้วตอง OK

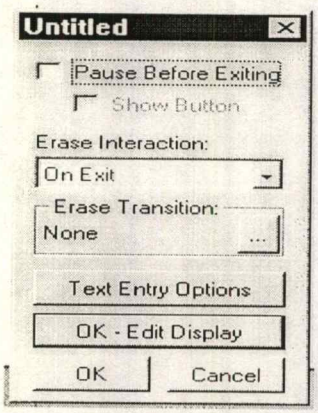
การใช้ Dicision Icon

เป็นคำสั่งควบคุมการทำงานของโปรแกรม โดยอาจให้มีการทำงานตามลำดับ (Sequence) ให้สุ่ม (Random) หรือควบคุมลำดับโดยใช้ตัวแปร

- Branching เป็นการกำหนดวิธีการหาเส้นทางในการทำงานซึ่งภายในแบ่งออกเป็น 4 วิธีคือ
  - -Sequential ให้เรียงลำดับไปจากซ้ายไปขวา
  - -Random w/o Replacement ให้ทำการสุ่มเส้นทางขึ้นมาโดยแต่ละครั้งจะไม่ซ้ำเส้นทางเดิม
  - -Random with Replacement ให้ทำการสุ่มเส้นทางขึ้นมาโดยแต่ละครั้งจะสามารถซ้ำเส้นทางเดิมได้
  - -Calculated Path เส้นทางที่เลือกขึ้นอยู่กับตัวแปรที่นำมาใส่ ถ้าตัวแปรนั้น เท่ากับ 1 จะไปเส้นทางแรก, 2 จะไปเส้นทางที่สอง, 3 จะไปเส้นทางที่สาม หากตัวแปรเป็นลบ หรือเกินขอบเขตเส้นทางที่มีอยู่จริงก็จะผ่านไป
- Reset Paths on Entry เป็นการรีเซ็ตเส้นทางที่เคยผ่านมาแล้ว
- Repeat เป็นการกำหนดวิธีในการทำงานซ้ำใน Dicision Icon
  - -Time จำนวนครั้งที่ต้องการซ้ำ
  - -Until All Selected ซ้ำจนครบทุกเส้นทาง
  - -Until Click / Keypress ซ้ำจนมีการกดคีย์ใดๆ หรือมีการคลิกเม้าส์
  - -Until TRUE ซ้ำจนเงื่อนไขเป็นจริง
- Don't Repet ไม่ต้องซ้ำ
- Time Limit ซ้ำจนครบเวลาที่กำหนด หน่วยเป็นวินาที
- Show Time Remaining แสดงนาฬิกาของเวลาที่ตั้งโดย Time Limit

การใช้ Interaction Icon

รูปร่างของ Interaction Icon มีลักษณะคล้ายลูกศร ซึ่งก็คือผลรวมของ Display Icon และ Decision Icon



ใช้ในการแสดงภาพเพื่อติดต่อกับผู้ใช้ โดยสามารถกำหนดเส้นทางการทำงานโดยตัวผู้ใช้เอง ในการติดต่อกับผู้ใช้นั้นทำได้ 10 วิธีคือ

ดับเบิลคลิกที่ Interaction Icon จะปรากฏลายละเอียดดังนี้

- Pause Before Exiting หยุดรอการกดคีย์ หรือปุ่ม ก่อนออก
- Show Prompt ให้แสดงปุ่มกด
- Erase Interaction ใช้ในการลบภาพใน Interaction Icon
  - Upon Exit ลบตอนออกจาก Interaction Icon
  - After Each Emtry ลบก่อนที่เข้าไปในเส้นทางที่เลือก
  - Don't Erase ไม่ต้องลบ เราสามารถลบได้โดยใช้ Erase Icon
  - Erase Effect กำหนดรูปแบบในการลบภาพ
- Text Entry Option ใช้กำหนดรูปแบบการรับตัวอักษร มีผลกับรูปแบบการติดต่อกับ ผู้ใช้โดยการรับตัวอักษรทางคียบอร์ดเท่านั้น
  - Character Limit จำนวนตัวอักษรที่กำหนด
  - Action key(s) จะทำงานเมื่อมีการกดคีย์ที่ระบุเท่านั้น
  - Auto Entry เมื่อเติมตัวอักษรครบตามกำหนดจะทำงานทันที
  - Ignore Null Entries ห้ามเว้นว่าง ต้องป้อนเสมอ
  - Entry Area Position & Size กำหนดตำแหน่งที่จะรับตัวอักษร
- Show Entry maker แสดงเครื่องหมาย ณ ตำแหน่งที่จะรับตัวอักษร
- Erase Entry On Exit ลบอักษรที่ป้อนก่อนออก
- OK-Edit display ใช้ในการเข้าไปแก้ไขภาพใน Interaction Icon

4.12 ฟังก์ชันใน Authorware

1. Random จะส่งค่าระหว่าง min และ max คูณกับ units

   รูปแบบ Number := Ramdom (min,max,unit)

2. Suจะส่งอักษรตั้งแต่ตำแหน่งที่ first ถึง last ของข้อความ String

   รูปแบบ String := Substr (string,first,last)

3. Go to ใช้เมื่อกระโดดไปยัง Icon ที่อ้างโดย Icon title

รูปแบบ Goto(IconID@* Icon title*)

4. JumpFileReturn ใช้ในการเรียก file ของ Authoware มาทำงานพร้อมทั้งส่งค่าตัวแปรไปทำงานใน ไฟล์ที่เรียกได้ด้วย และเมื่อต้องการกลับสู่ไฟล์เดิม ใช้ Quit(0)

5. JumpOutReturn ใช้ในการเรียก Application อื่นๆ และเมื่อทำงานเสร็จก็จะกลับสู่ไฟล์เดิม

6. exit ใช้ในการออกจากโปรแกรมที่กำลังทำงาน

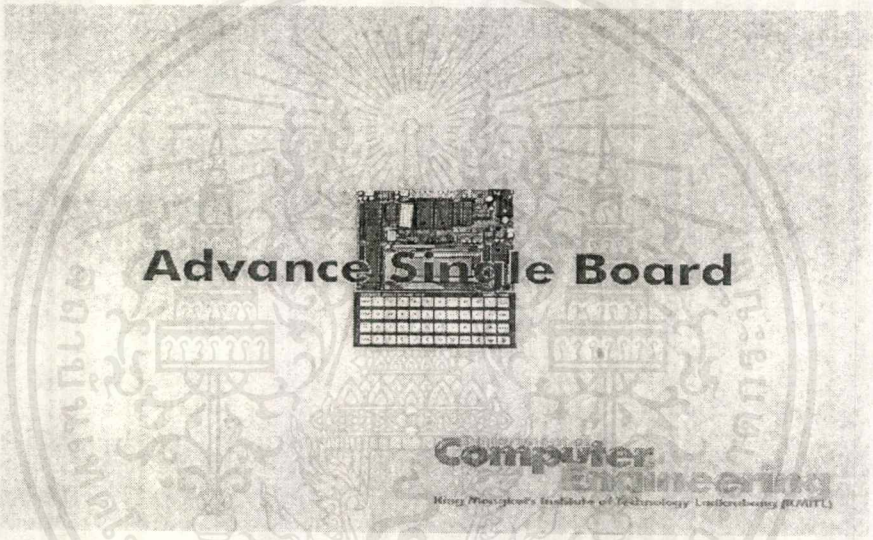
สรุป

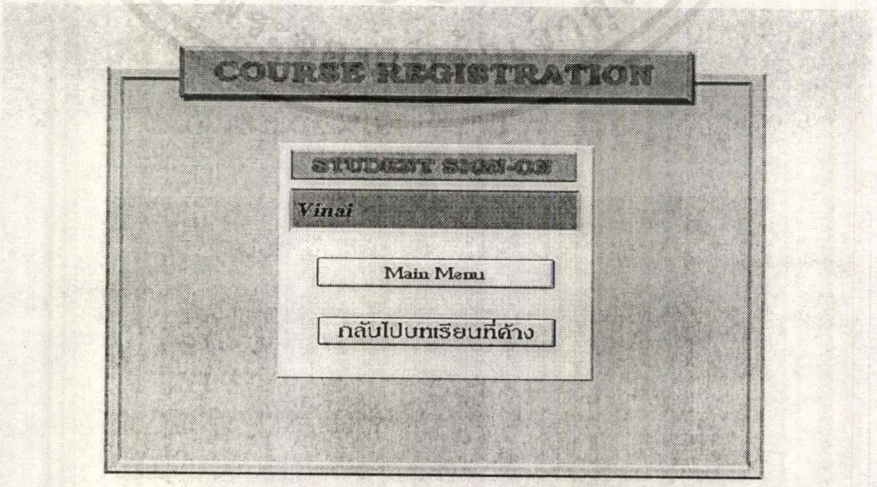
เนื่องจากโปรแกรม Authorware เป็นโปรแกรมที่สามารถสร้างระบบช่วยสอนได้ง่ายและมี Icon ให้เลือกใช้ได้สะดวกทำให้ไม่ต้องจำรูปแบบของคำสั่ง สามารถนำเสนอข้อมูลได้รวดเร็วยิ่งขึ้น โครงงานนี้จึงเลือกโปรแกรมนี้ในการทำระบบคอมพิวเตอร์ช่วยสอน

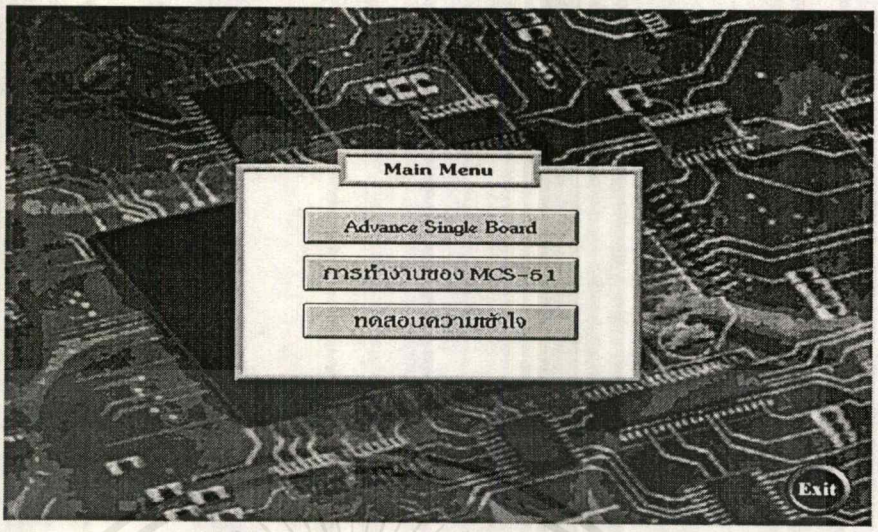
บทที่ 5

รายละเอียดภายใน ระบบคอมพิวเตอร์ช่วยสอน

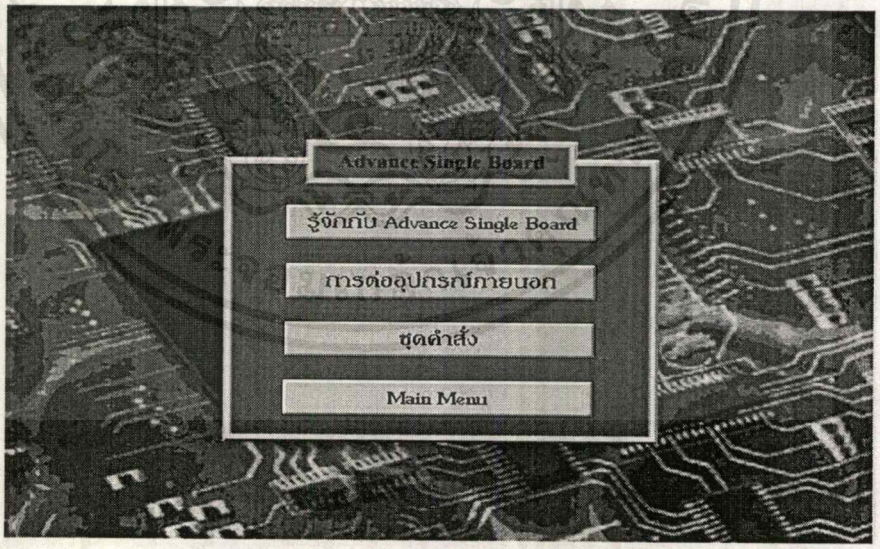
ในบทนี้จะเป็นเนื้อหาเกี่ยวกับรายละเอียดภายในของระบบคอมพิวเตอร์ช่วยสอน ประกอบด้วย การแนะนำและสอนการใช้งานของเครื่อง Advance Single Board จำนวน 14 pages และนำโครงสร้างและการทำงานของ MCS-51 จำนวน 80 pages และส่วนสุดท้ายจะเป็นแบบทดสอบความเข้าใจเมื่อเรียนจบจำนวน 50 ข้อ โดยมีรายลัเอียดดังนี้

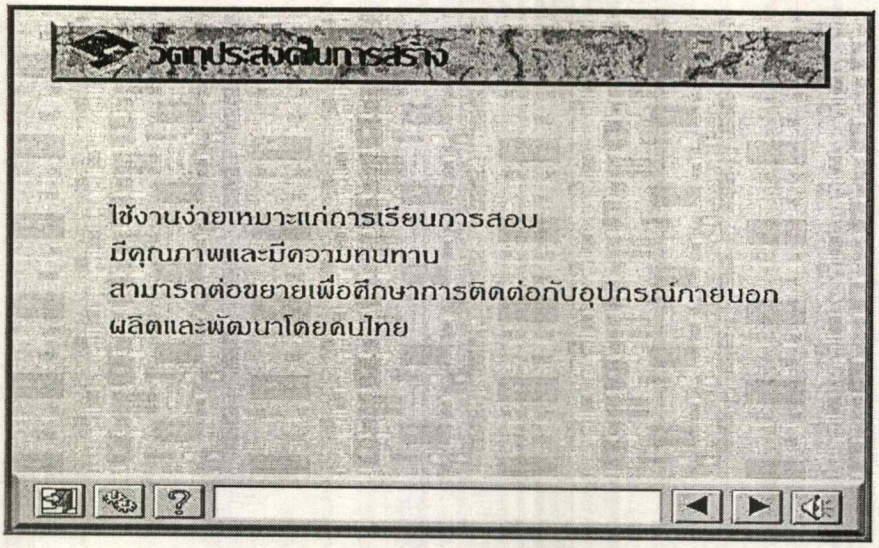


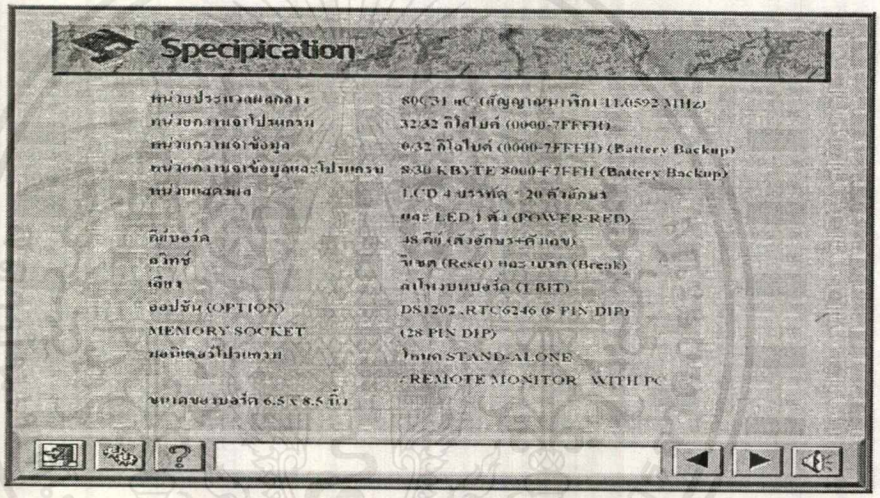


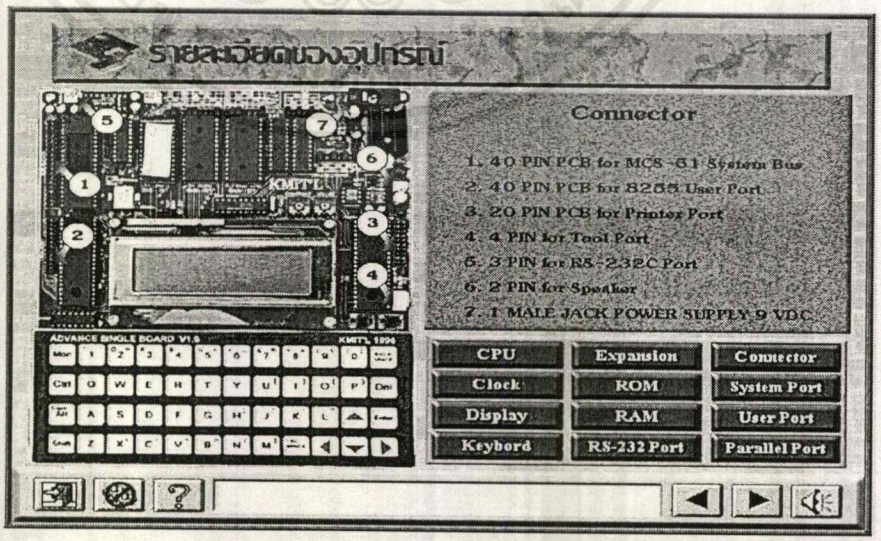


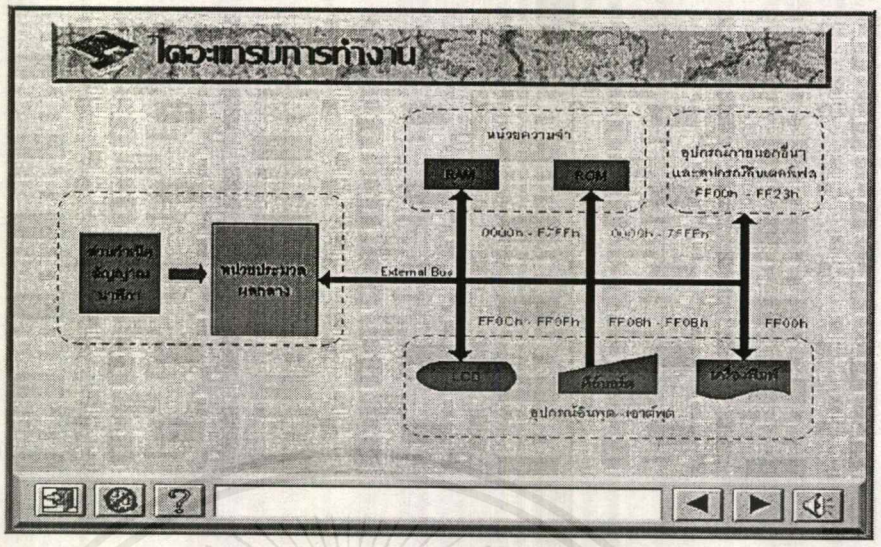
Advance Single Board

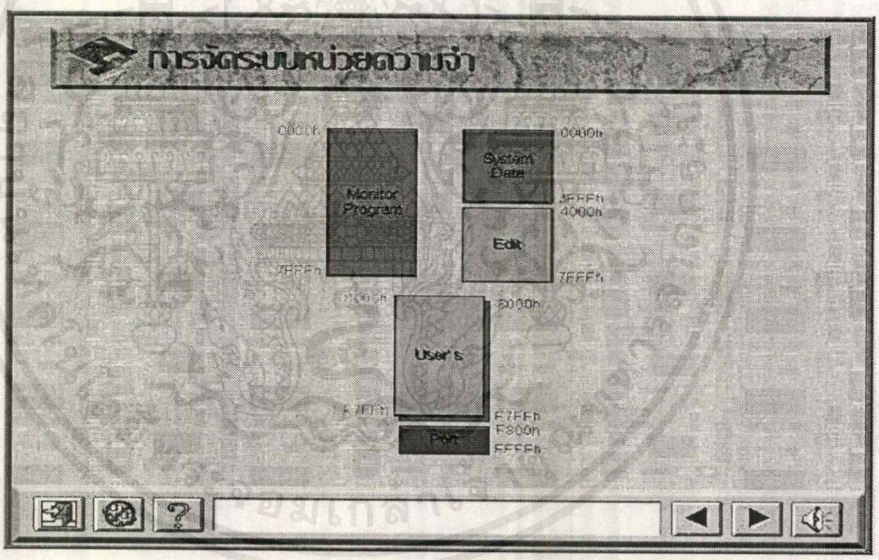




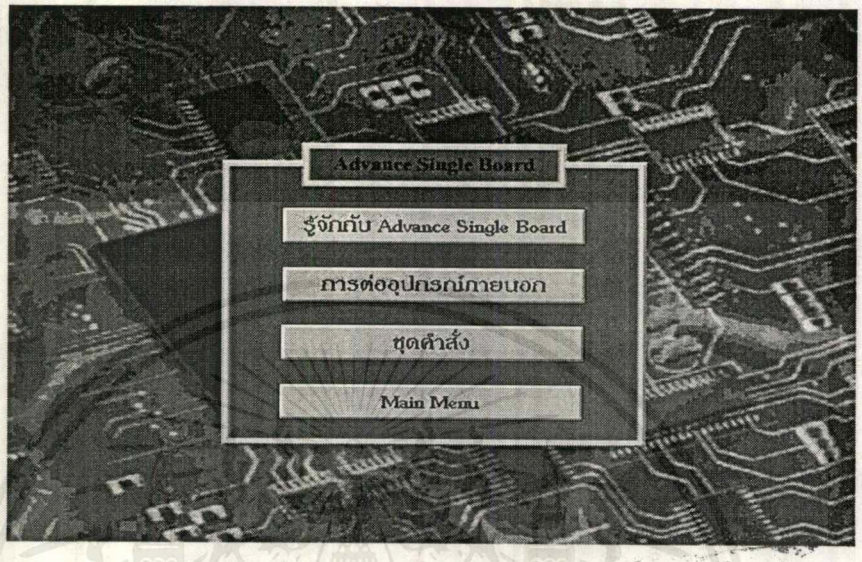


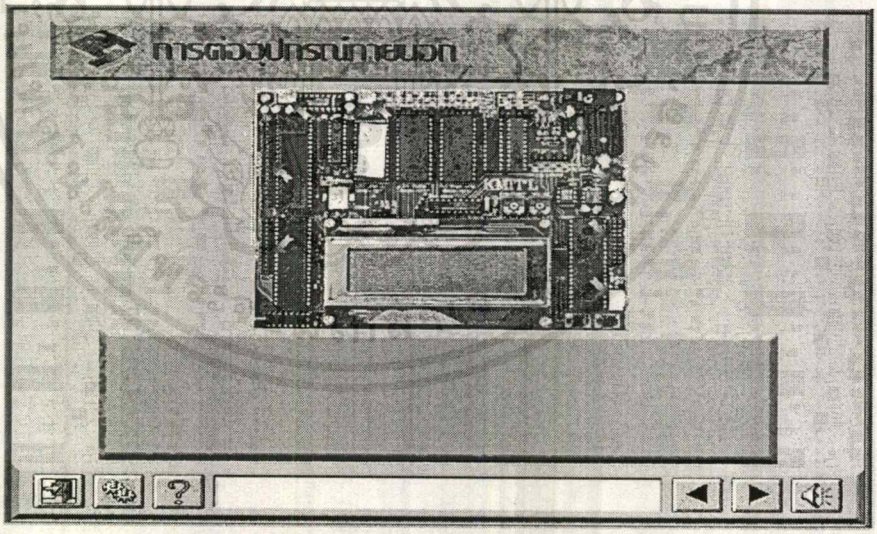


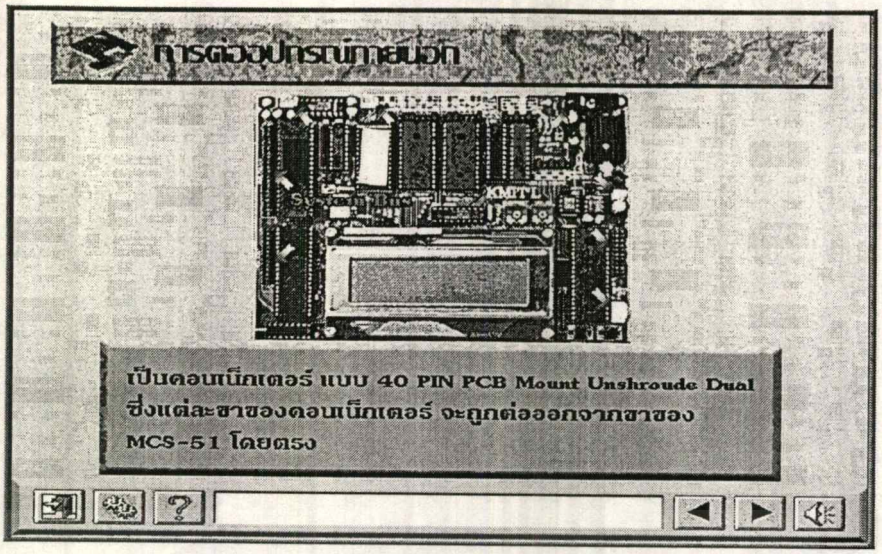


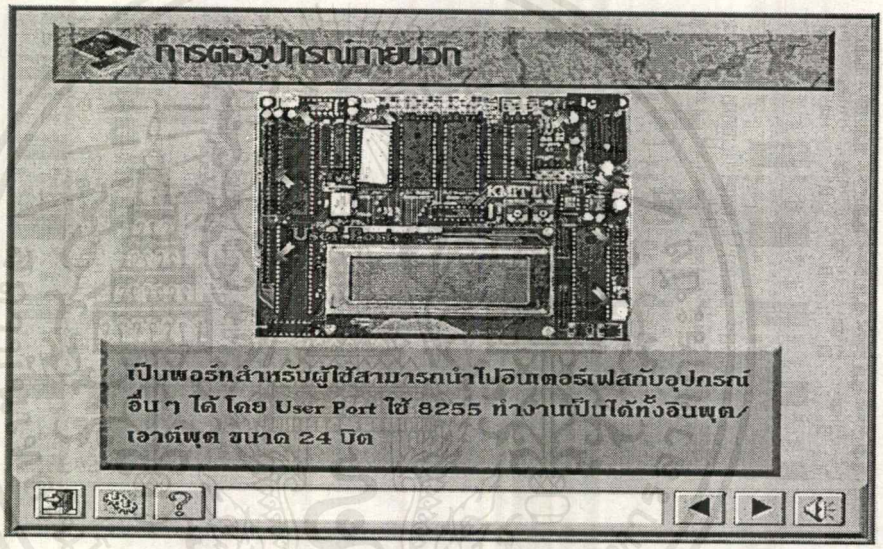


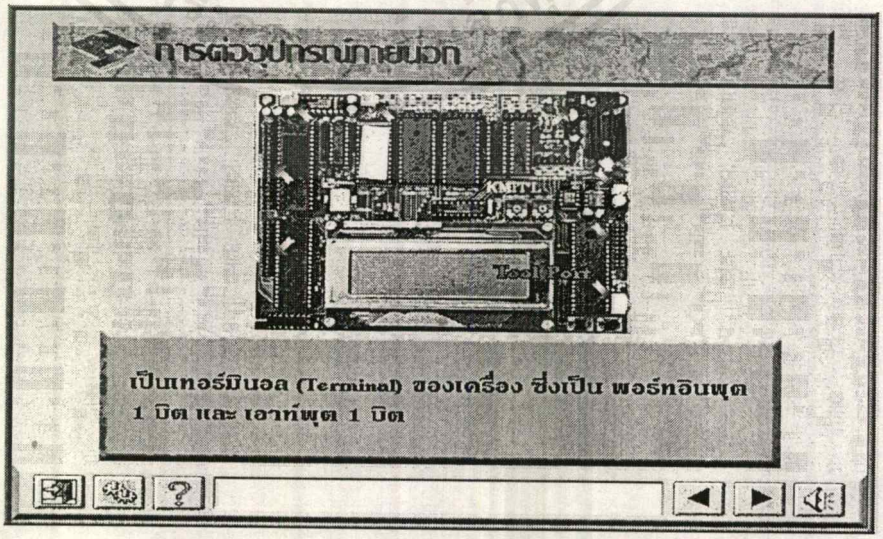
การต่ออุปกรณ์ภายนอก

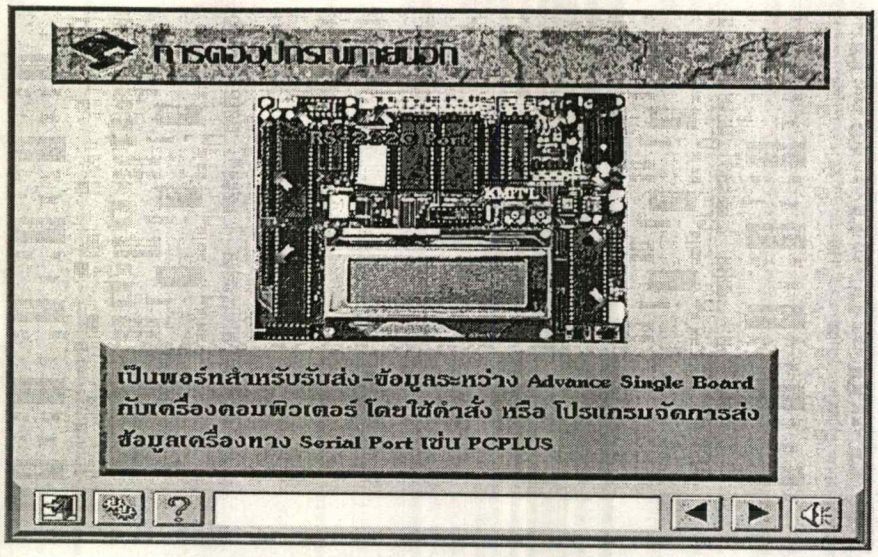


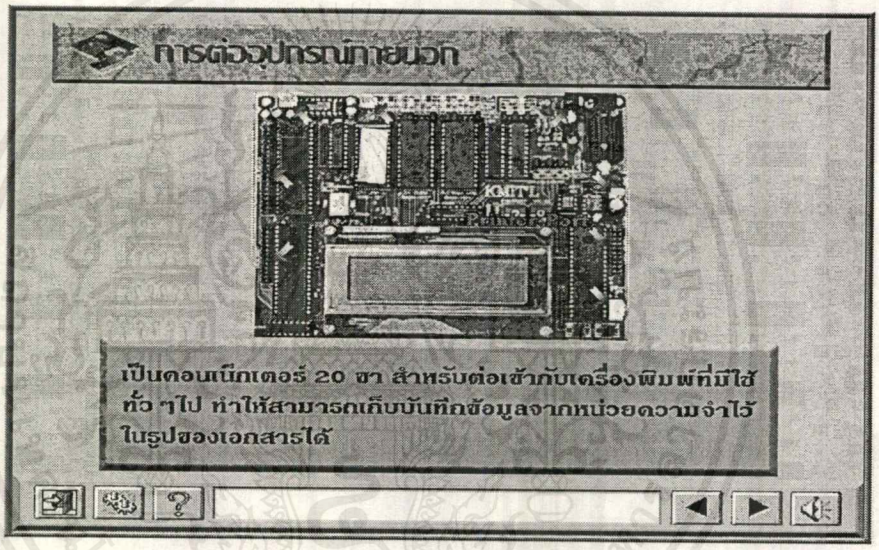


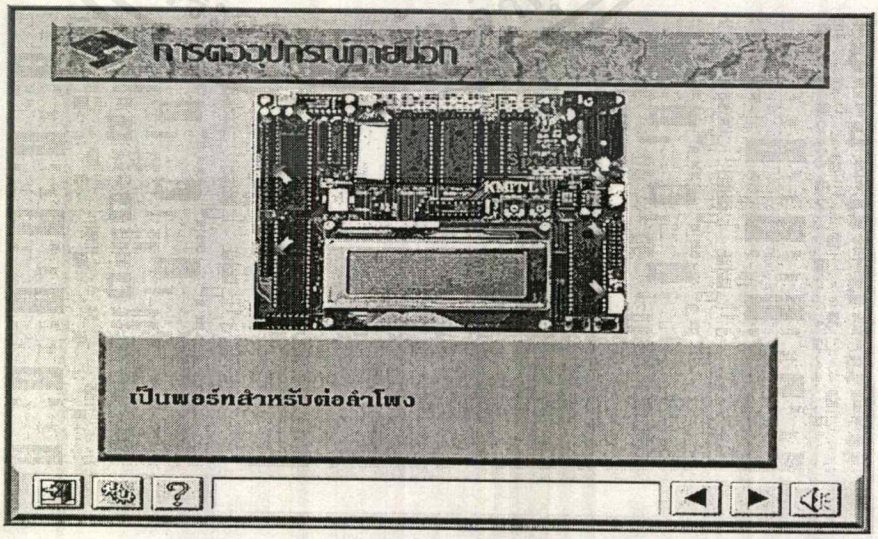


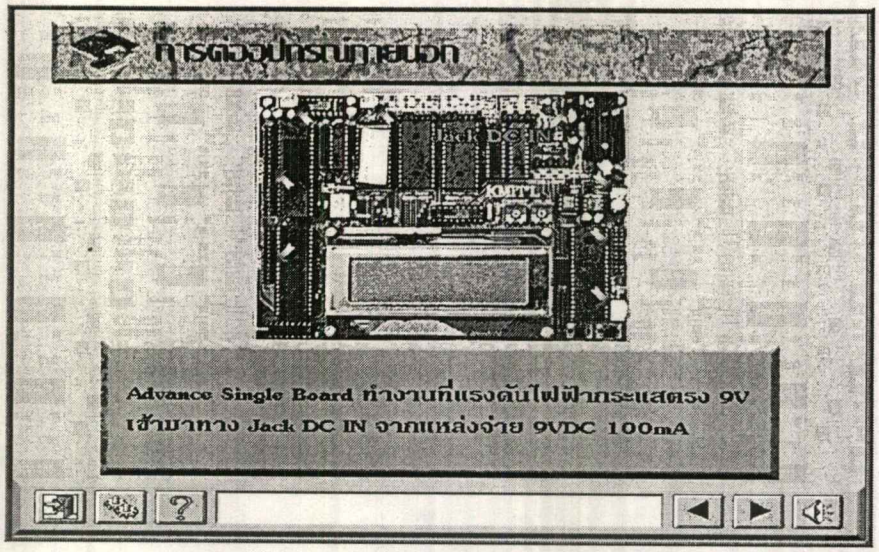




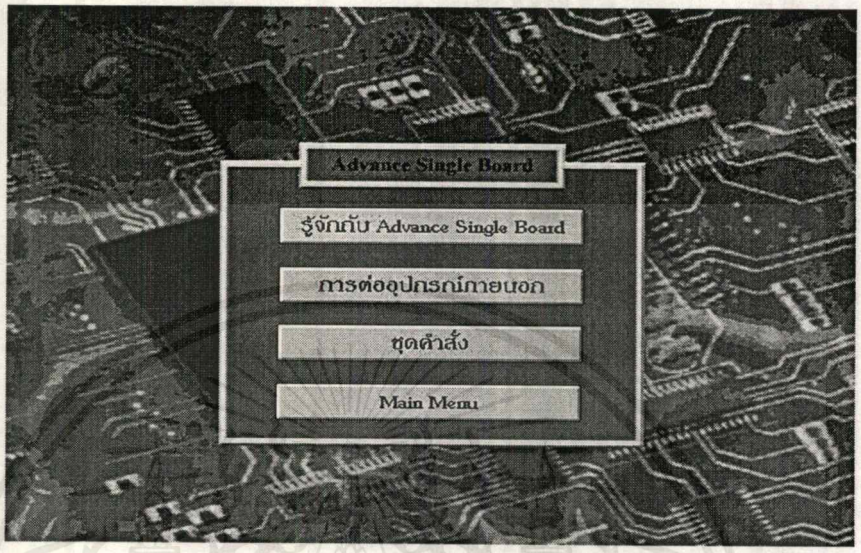


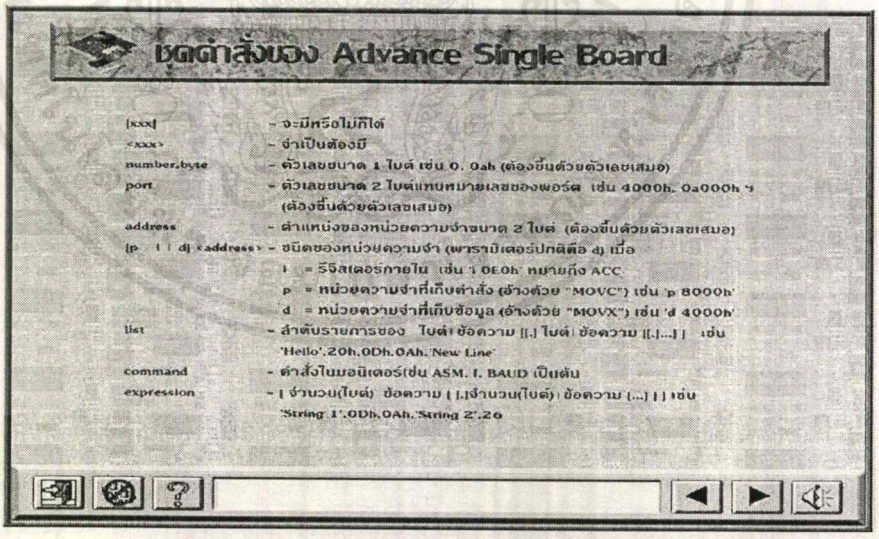


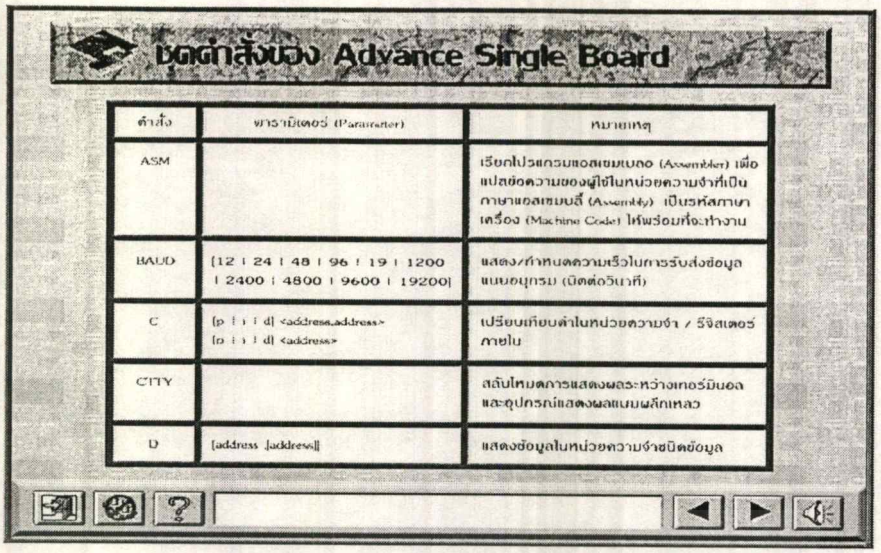




ชุดคำสั่งของ Advance Single Board

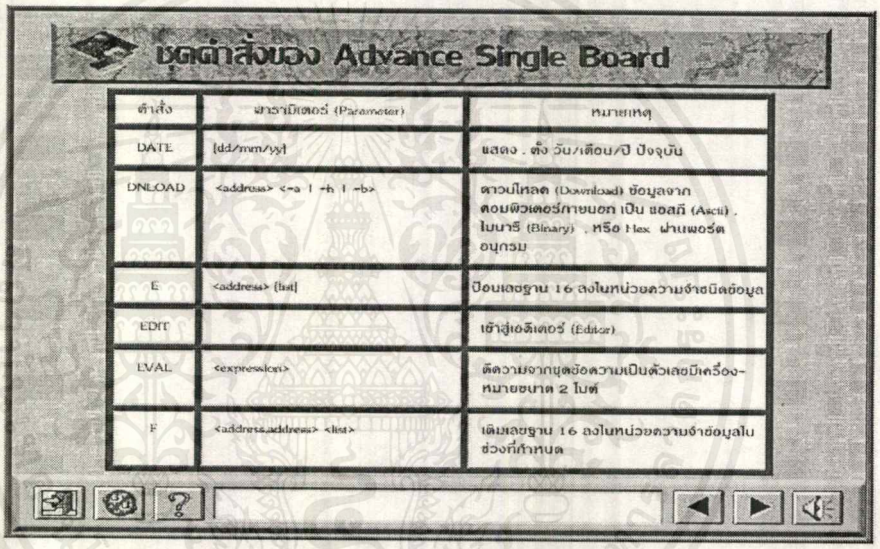






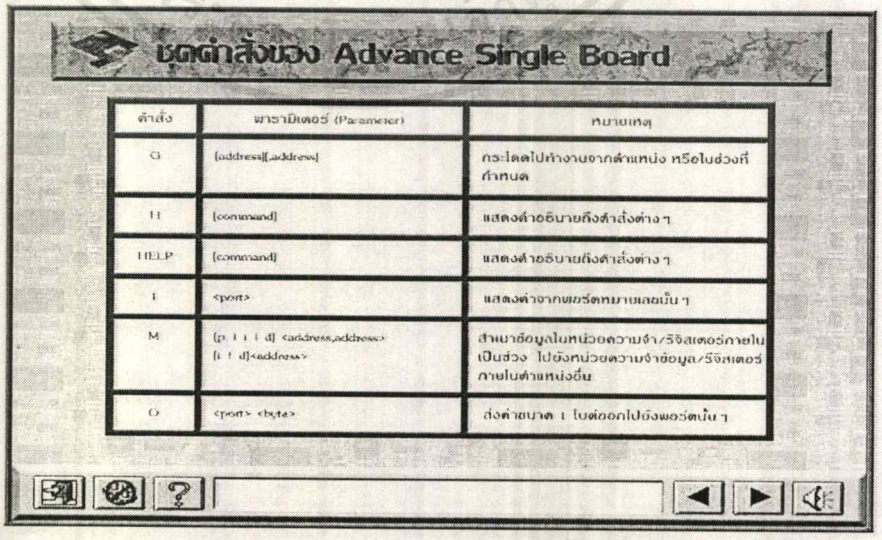
ชุดคำสั่งของ Advance Single Board

คำสั่ง	พารามิเตอร์ (Parameter)	หมายเหตุ
ASM		เรียกโปรแกรมแอสเซมเบลอ (Assembler) เพื่อแปลข้อความของผู้ใช้ในหน่วยความจำที่เป็นภาษาแอสเซมบลี้ (Assembly) เป็นรหัสภาษาเครื่อง (Machine Code) ให้พร้อมที่จะทำงาน
BAUD	[12 ¦ 24 ¦ 48 ¦ 96 ¦ 19 ¦ 1200 ¦ 2400 ¦ 4800 ¦ 9600 ¦ 19200]	แสดง/กำหนดความเร็วในการรับส่งข้อมูลแบบอนุกรม (บิตต่อวินาที)
C	[p ¦ i ¦ d] <address,address> [n ¦ i ¦ d] <address>	เปรียบเทียบค่าในหน่วยความจำ / รีจิสเตอร์ภายใน
CITY		สลับโหมดการแสดงผลระหว่างเทอร์มินอลและอุปกรณ์แสดงผลแบบผลึกเหลว
D	[address ,[address]]	แสดงข้อมูลในหน่วยความจำชนิดข้อมูล



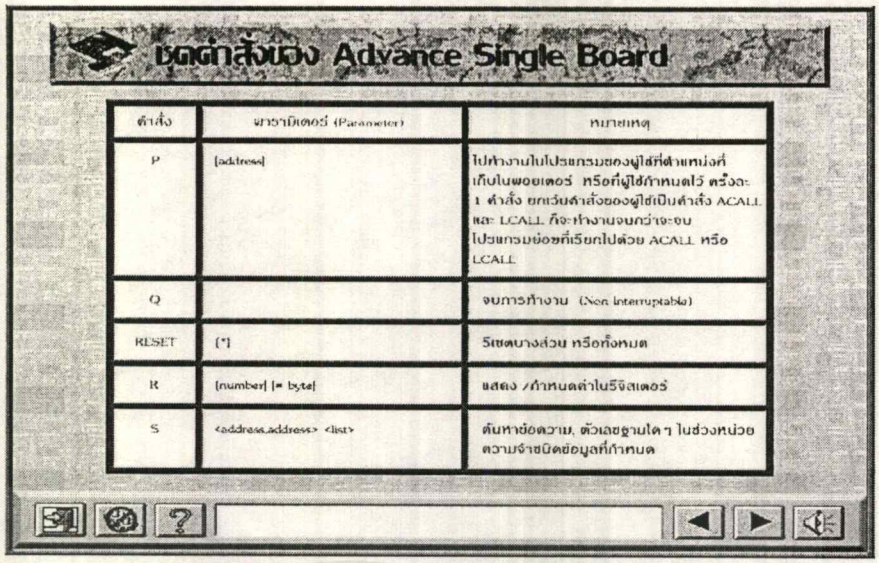
ชุดคำสั่งของ Advance Single Board

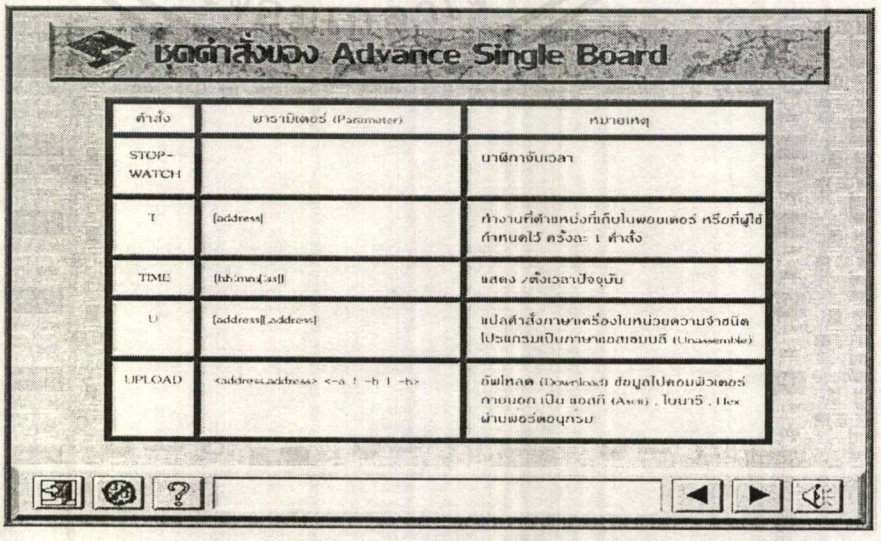
คำสั่ง	พารามิเตอร์ (Parameter)	หมายเหตุ
DATE	[dd/mm/yy]	แสดง , ตั้ง วัน/เดือน/ปี ปัจจุบัน
DNLOAD	<address> <-a ¦ -h ¦ -b>	ดาวน์โหลด (Download) ข้อมูลจากคอมพิวเตอร์ภายนอก เป็น แอสกี (Ascii) , ไบนารี (Binary) , หรือ Hex ผ่านพอร์ตอนุกรม
E	<address> [list]	ป้อนเลขฐาน 16 ลงในหน่วยความจำชนิดข้อมูล
EDIT		เข้าสู่เอดิเตอร์ (Editor)
EVAL	<expression>	ตีความจากชุดข้อความเป็นตัวเลขมีเครื่องหมายขนาด 2 ไบต์
F	<address,address> <list>	เติมเลขฐาน 16 ลงในหน่วยความจำข้อมูลในช่วงที่กำหนด

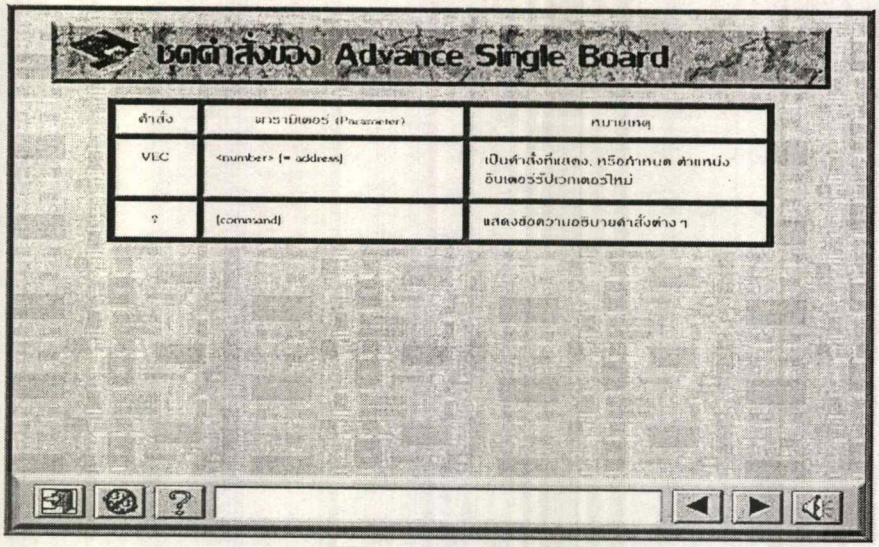


ชุดคำสั่งของ Advance Single Board

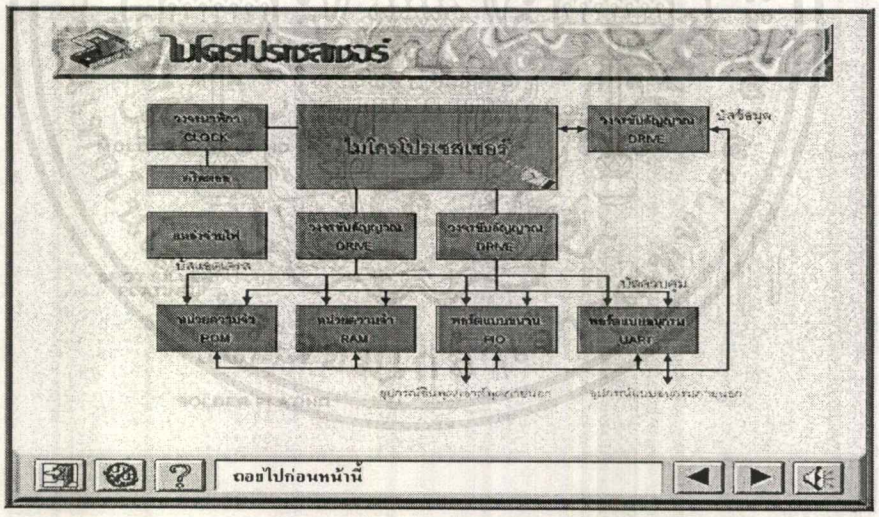
คำสั่ง	พารามิเตอร์ (Parameter)	หมายเหตุ
G	[address][,address]	กระโดดไปทำงานจากตำแหน่ง หรือในช่วงที่กำหนด
H	[command]	แสดงคำอธิบายถึงคำสั่งต่าง ๆ
HELP	[command]	แสดงคำอธิบายถึงคำสั่งต่าง ๆ
I	<port>	แสดงค่าจากพอร์ตหมายเลขนั้น ๆ
M	[p ¦ i ¦ d] <address,address> [i ¦ d]<address>	สำเนาข้อมูลในหน่วยความจำ/รีจิสเตอร์ภายในเป็นช่วง ไปยังหน่วยความจำข้อมูล/รีจิสเตอร์ภายในตำแหน่งอื่น
O	<port> <byte>	ส่งค่าขนาด 1 ไบต์ออกไปยังพอร์ตนั้น ๆ



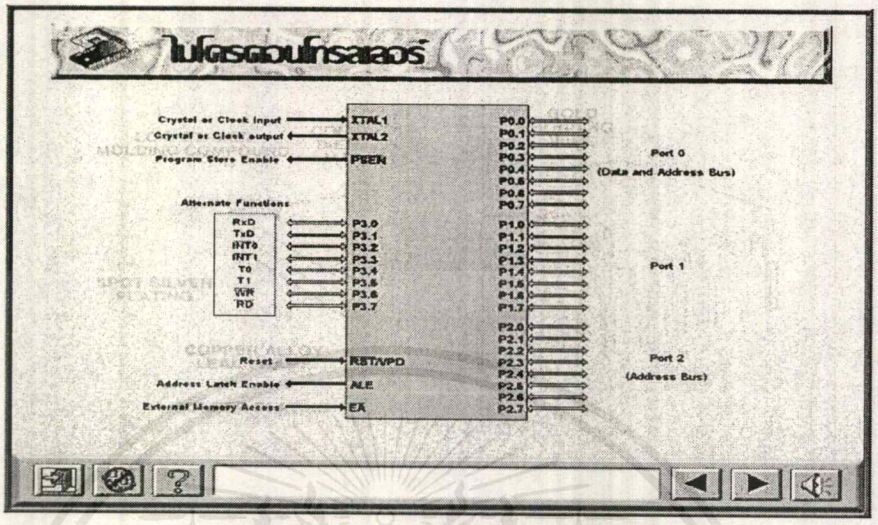




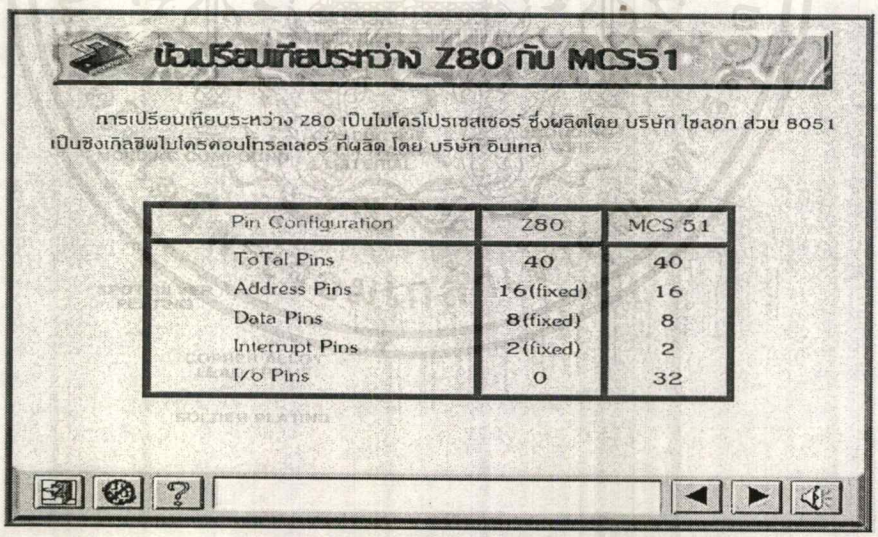
การทำงานของ MCS-51



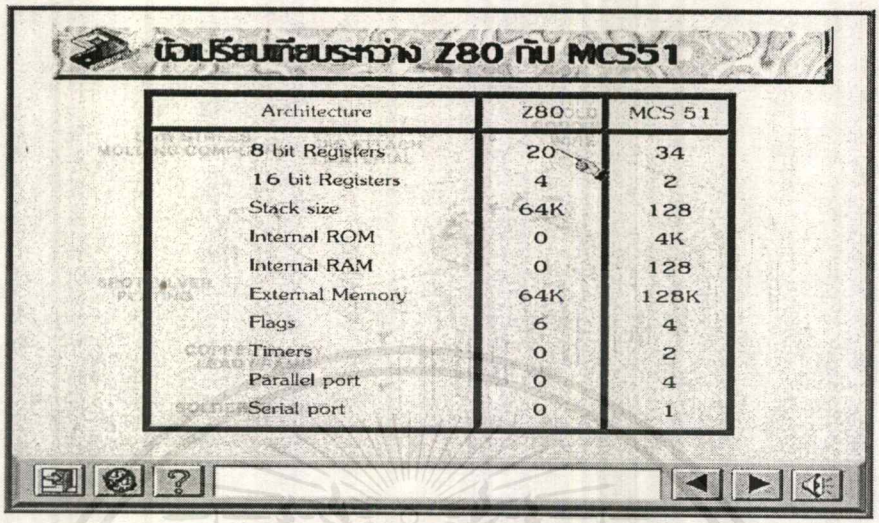
คำบรรยาย หน่วยประมวลผลส่วนมากจะนึกถึง Microprocessor เช่น Z-80 แต่ก่อนจะนำมาใช้งานได้นั้นจะต้องต่ออุปกรณ์ภายนอกเช่น หน่วยความจำ ROM, RAM , Timer/Counter , Port ,Serial Port เพื่อให้สามรถทำงานได้



คำบรรยาย Microcontroller เริ่มได้รับความนิยมเพิ่มมากขึ้น และสามารถทำงานได้ด้วย Chip เพียงตัวเดียว โดยอุปกรณ์ภายนอกที่จำเป็นมารวมกัน เช่น RAM, ROM, Timer/Counter MCS-51 ก็เป็น Microcontroller ตัวหนึ่งที่ได้รับความสนใจ

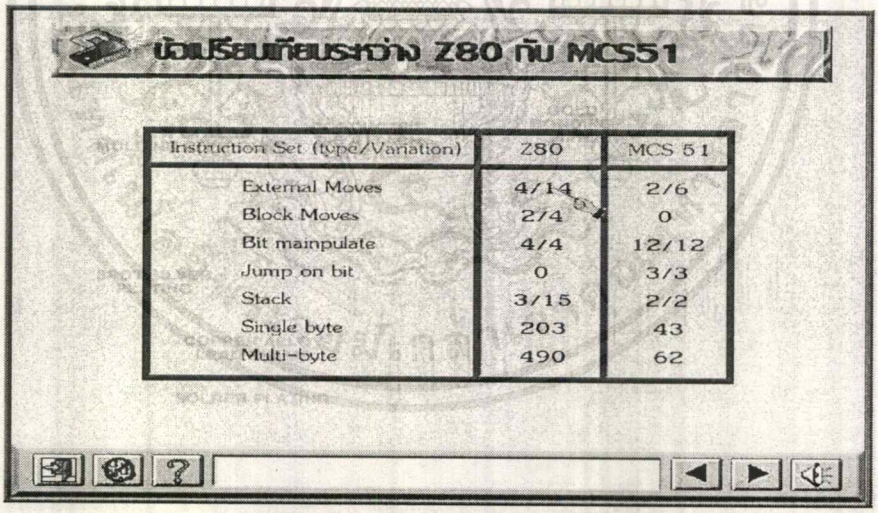


คำบรรยาย การเปรียบเทียบระหว่าง Z-80 กับ MCS-51 ซึ่ง Z-80 เป็น Microprocesser ซึ่งผลิตโดยบริษัท ไซรอก ส่วน MCS-51 เป็น Single Chip ที่ผลิตโดบบริษัท Intel



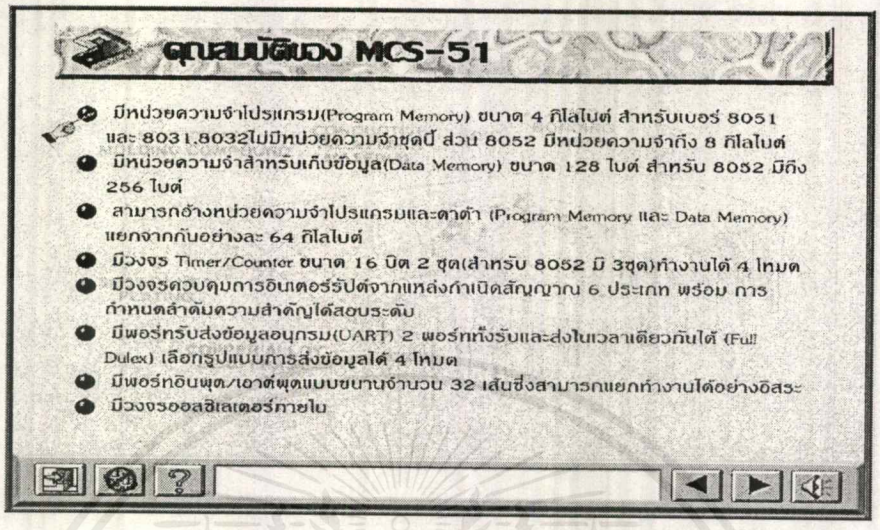
Architecture	Z80	MCS 51
8 bit Registers	20	34
16 bit Registers	4	2
Stack size	64K	128
Internal ROM	0	4K
Internal RAM	0	128
External Memory	64K	128K
Flags	6	4
Timers	0	2
Parallel port	0	4
Serial port	0	1

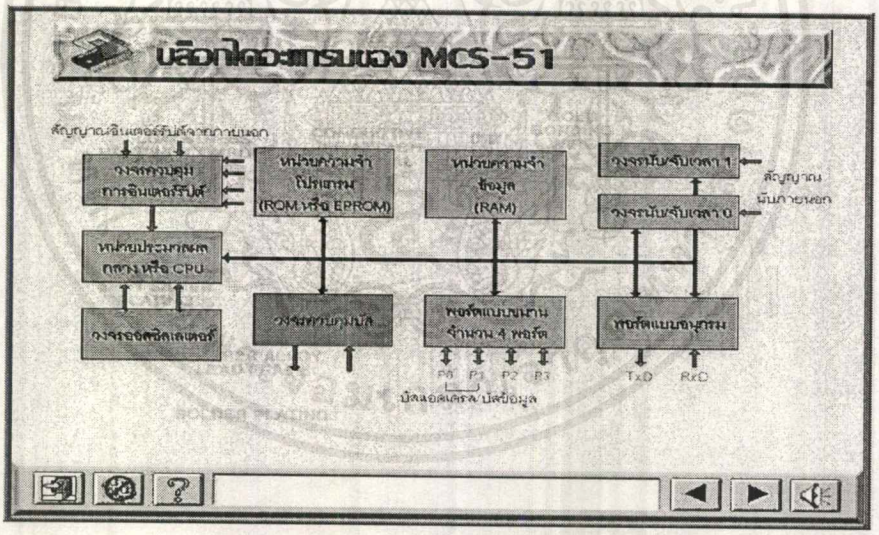
คำบรรยาย การเปรียบเทียบระหว่าง Z-80 กับ MCS-51 ทางด้านArchitec-ture แสดงดังในตารางข้างบน



Instruction Set (type/Variation)	Z80	MCS 51
External Moves	4/14	2/6
Block Moves	2/4	0
Bit mainpulate	4/4	12/12
Jump on bit	0	3/3
Stack	3/15	2/2
Single byte	203	43
Multi-byte	490	62

คำบรรยาย การเปรียบเทียบระหว่าง Z-80 กับ MCS-51 ทางด้าน Instruction Set โดยจะแสดงให้เห็นระหว่าง ชนิดคำสั่งกับคำสั่งจริง ดังในตารางข้างบน





คำบรรยาย โครงสร้างของไมโครคอนโทรลเลอร์ตระกูล MCS-51 ประกอบด้วย 3 ส่วนหลักๆ คือ

ส่วนที่ 1 CPU(Central Processing Unit) หรือตัวประมวลผล ส่วนนี้จะมีวงจรที่ทำหน้าที่สร้างสัญญาณควบคุมในการติดต่อกับส่วนอื่นๆเรียกว่า วงจรควบคุม(Control Unit)สัญญาณที่สร้างจากวงจรควบคุมได้ แก่สัญญาณสำหรับการติดต่อกับหน่วยความจำ, อุปกรณ์รับข้อมูลหรือส่งข้อ

มูลออกจากตัว MCS 51 ซึ่งส่วนควบคุมการขัดจังหวะ(Interrupt Control) และส่วนควบคุมบัส(Bus Control) ก็เป็นส่วนหนึ่งของวงจรควบคุมด้วย

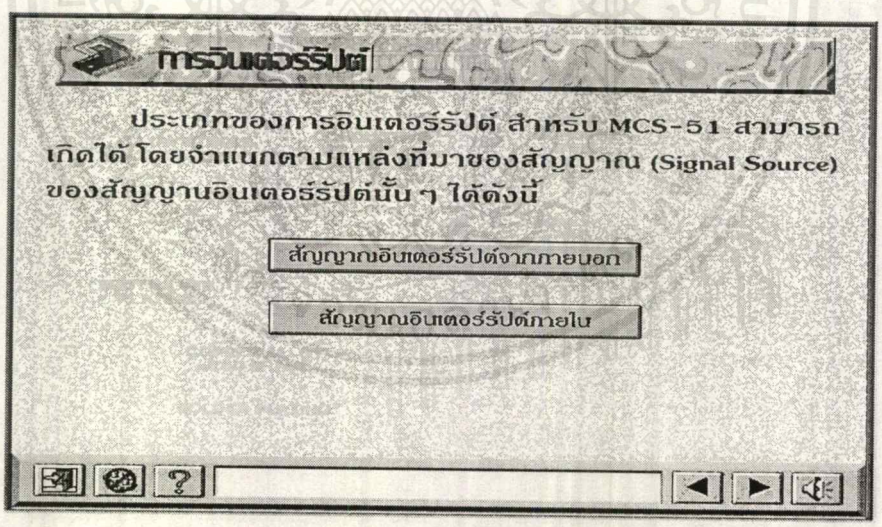
ใน CPU นี้ยังประกอบด้วยส่วนย่อยอีกเรียกว่าส่วนประมวลผล (ALU) ส่วนนี้จะทำหน้าที่ประมวลผลข้อมูลเช่น การบวก การลบ คูณ หรือหาร แล้วนำผลลัพธ์ไปเก็บไว้ในรีจิสเตอร์หรือหน่วยความจำที่ต้องการ

ส่วนที่ 2 หน่วยความจำ (Memory) มีไว้สำหรับจดจำข้อมูล ถ้าจะให้เห็นภาพพจน์ของหน่วยความจำได้ดีก็คือ เหมือนกล่องเก็บเอกสารจำนวนมากที่นำมาวางต่อเรียงกันแต่ละกล่องมีเอกสาร 1 แผ่น

ในระบบของ MCS-51 จะมีการแบ่งหน่วยความจำเหมือนกับซีพียูทั่วๆไปคือ จะแบ่งเป็น 2 ลักษณะตามชนิดของข้อมูลที่เก็บดังนี้

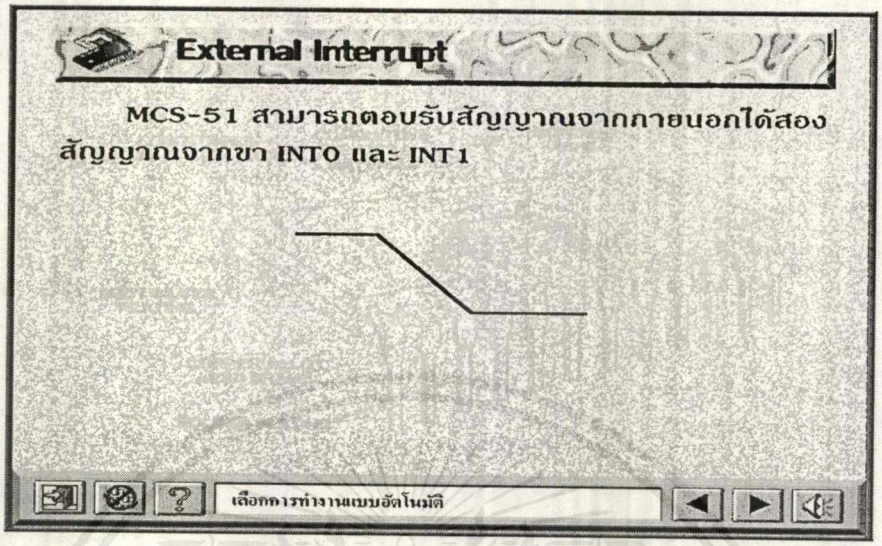
1. หน่วยความจำโปรแกรม 2. หน่วยความจำข้อมูล

ส่วนที่ 3 อุปกรณ์อินพุทและเอาท์พุท(Input/Output Device) เป็นส่วนที่จะส่งข้อมูลเข้าหรือออกจาก MCS-51 ทำให้ติดต่อกับภายนอกได้ ซึ่งอุปกรณ์อินพุทและเอาท์พุท ได้แก่ 4 I/O Port, Timer 0, Timer 1, Serial Port



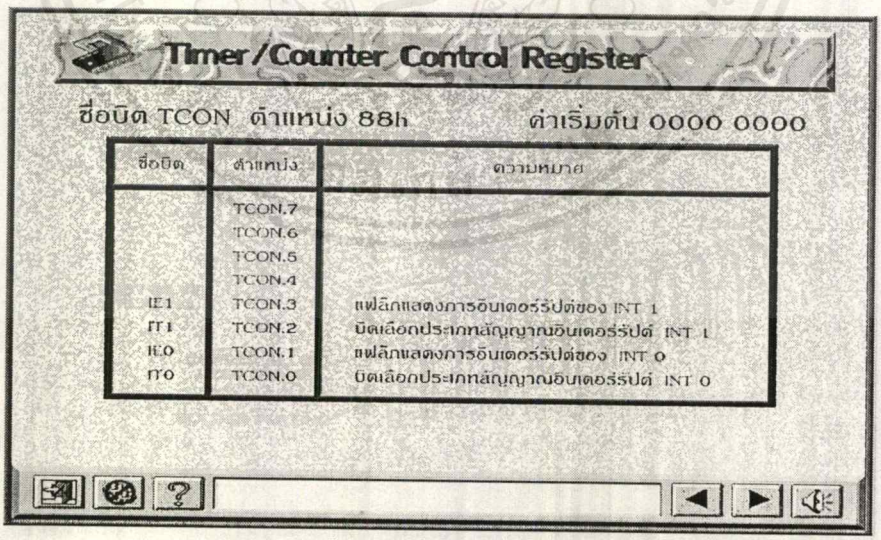
คำบรรยาย การอินเตอร์รัปต์ ประเภทการอินเตอร์รัปต์ สำหรับ MCS-51 สามารถเกิดได้โดยจำแนกตามแหล่งที่มาของสัญญาณ (Signal Source) ของสัญญาณอินเตอร์รัปต์นั้นๆ ได้ดังนี้

1. สัญญาณอินเตอร์รัปต์จากภายนอก 2. สัญญาณอินเตอร์รัปต์จากภายใน



คำบรรยาย สัญญาณอินเตอร์รัปต์จากภายนอก(External Interrupt)MCS-51 สามารถตอบรับสัญญาณจากภายนอกได้สองสัญญาณจากขา Int0 และ Int1

ซึ่งสามารถโปรแกรมให้เป็นแบบกระตุ้นด้วยขอบสัญญาณ (TRANSITIONACTIVATED) หรือ กระตุ้นด้วยระดับสัญญาณ (LEVEL-ACTIVATED)ขึ้นอยู่กับบิท IT0 และ IT1 ในรีจิสเตอร์ TCON



คำบรรยาย TCON (Timer/Counter Control Register) สำหรับเก็บสถานะการทำงานของอินเตอร์หรับจาภายนอก

TF1 TR1 TF0 TR0 IE1 IT1 IE0 IT0

IE1 บิทแสดงสถานะสัญญาณอินเตอร์รัพท์ภายนอก ชนิดที่ 1 จะถูกเซตเองโดยฮาร์ดแวร์ เมื่อมีสัญญาณอินเตอร์รัพท์ และจะถูกเคลียร์เองโดยคำสั่ง RETI ที่อยู่ในโปรแกรมส่วนบริการอินเตอร์รัพท์ เมื่ออินเตอร์รัพท์ที่เกิดขึ้นได้มาจากการตรวจสอบ จากการเปลี่ยนสถานะของสัญญาณ IT1 บิทเลือกประเภทการตรวจสอบสัญญาณอินเตอร์รัพท์ที่เกิดขึ้นที่ขา INT1 โดย

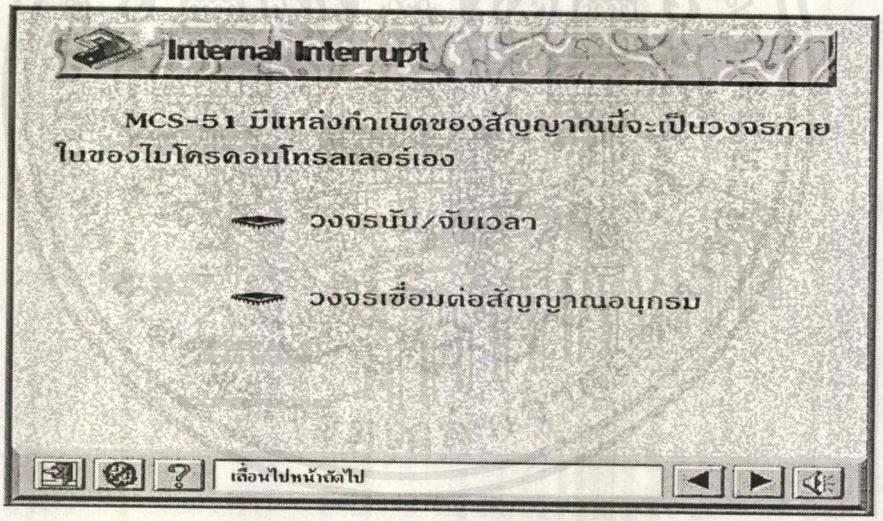
1 ตรวจสอบการเปลี่ยนระดับสัญญาณจาก 1 เป็น 0 ที่ขา INT1

0 ตรวจสอบระดับสัญญาณที่ขา INT1

IE0 บิทแสดงสถานะสัญญาณอินเตอร์รัพท์ภายนอก ชนิดที่ 0 จะถูกเซตเองโดยฮาร์ดแวร์ เมื่อมีสัญญาณอินเตอร์รัพท์ และจะถูกเคลียร์เองโดยคำสั่ง RETI ที่อยู่ในโปรแกรมส่วนบริการอินเตอร์รัพท์ เมื่ออินเตอร์รัพท์ที่เกิดขึ้นได้มาจากการตรวจสอบจากการเปลี่ยนสถานะของสัญญาณ IT0 บิทเลือกประเภทการตรวจสอบสัญญาณอินเตอร์รัพท์ที่เกิดขึ้นที่ขา INT0 โดย

1 ตรวจสอบการเปลี่ยนระดับสัญญาณจาก 1 เป็น 0 ที่ขา INT0

0 ตรวจสอบระดับสัญญาณที่ขา INT0

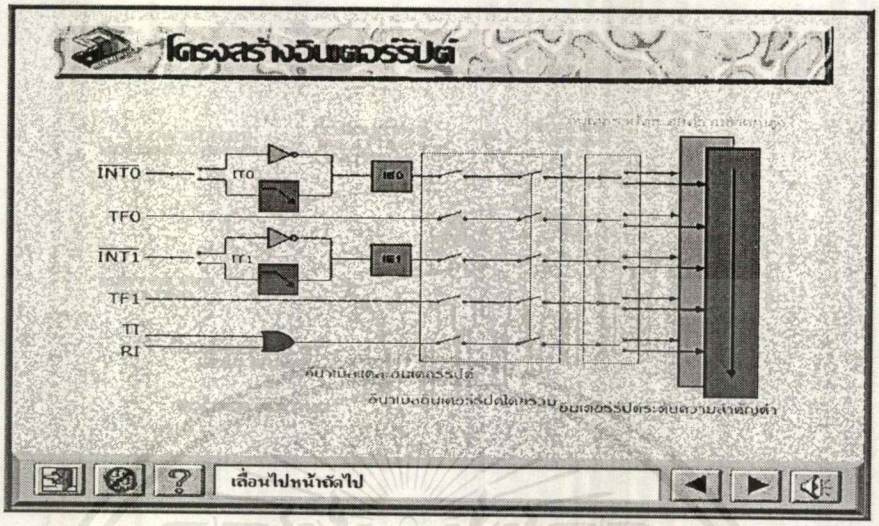


คำบรรยาย สัญญาณอินเตอร์รัปต์จากภายใน (Internal Interrupt) MCS-51 มีแหล่งกำเนิดของสัญญาณนี้ จะเป็นวงจรภายในของไมโครคอนโทรลเลอร์เอง

อินเตอร์รัพท์จากวงจรนับ/จับเวลา เกิดขึ้นโดยแฟลก TF0 และ TF1 เมื่อ เกิดอินเตอร์รัพท์ขึ้นแฟล็กจะถูกเคลียร์โดยฮาร์ดแวร์เมื่อกระโดดไปทำงานที่ SERVICE ROUTINE

อินเตอร์รัพท์จากวงจรเชื่อมต่อสัญญาณอนุกรม เกิดขึ้นจากทางรับ หรือ ทางส่งข้อมูลโดยที่โปรแกรมต้องตรวจเช็คว่าเป็นอินเตอร์รัพท์จากด้านรับ (RI) หรือด้านส่ง (TI) และจะต้องทำการ

เคลียร์แฟลก อินเตอร์รัพท์ด้วยซอร์ฟแวร์



คำบรรยาย ระบบอินเตอร์รัพท์ของ 8051 สัญญาณที่เข้ามาทำการอินเตอร์รัพท์ 8051 เกิดขึ้นได้ 5 ลักษณะ คือ

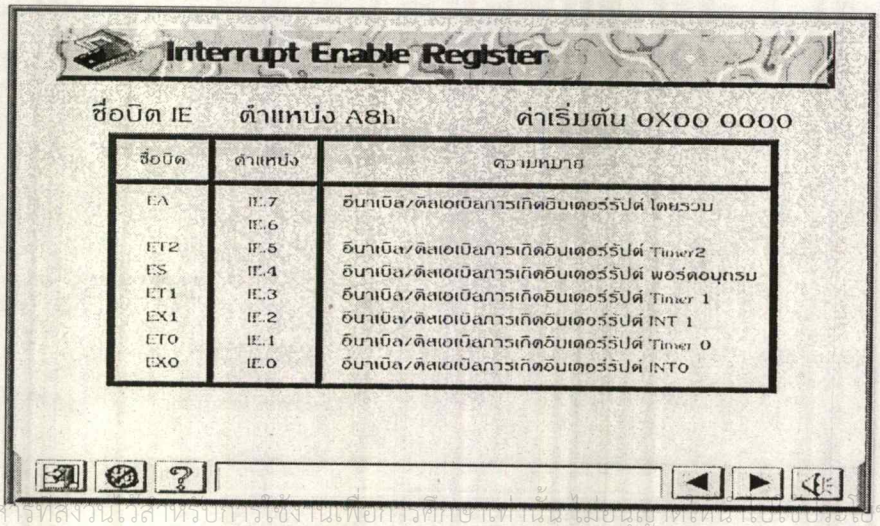
INT0 สัญญาณอินเตอร์รัพท์จากภายนอกทางขาสัญญาณ P3.2 โดย MCS-51 จะทำการสุ่มตัวอย่างเมื่อสิ้นสุดทุกแมชชีนไซเคิล

INT1 สัญญาณอินเตอร์รัพท์จากภายนอกทางขาสัญญาณ P3.3 โดย MCS-51 จะทำการสุ่มตัวอย่างเมื่อสิ้นสุดทุกแมชชีนไซเคิล

Timer 0 สัญญาณการเกิดโอเวอร์โฟลว์ของ Timer 0

Timer 1 สัญญาณการเกิดโอเวอร์โฟลว์ของ Timer 1

พอร์ทอนุกรม การเกิดอินเตอร์รัพท์ที่เกิดขึ้นจากการรับส่งข้อมูลอนุกรม ทำให้มีผลต่อแฟล็กอินเตอร์รัพท์ R1 และ T1 ตามลำดับ



คำบรรยาย IE (INTERRUPT ENABLE REGISTER) เป็นรีจิสเตอร์ที่ใช้ในการ ENABLE หรือ DISABLE ขอสัญญาณอินเตอร์รัพท์ต่างบิทต่างของ IE ดังมีรายละเอียดดังนี้

EA	X	ET2	ES	ET1	EX1	ET0	EX0

EX0 บิทนี้ใช้สำหรับการ ENABLE สัญญาณที่เข้ามาทางขา INT0 ให้เกิดการขัดจังหวะ

ET0 TIMER 0 INTERRUPT ENABLE BIT ข้อมูลบิทนี้จะใช้ ENABLE หรือ DISABLEสัญญาณขัดจังหวะที่มาจากวงจร TIMER 0 (TF0)

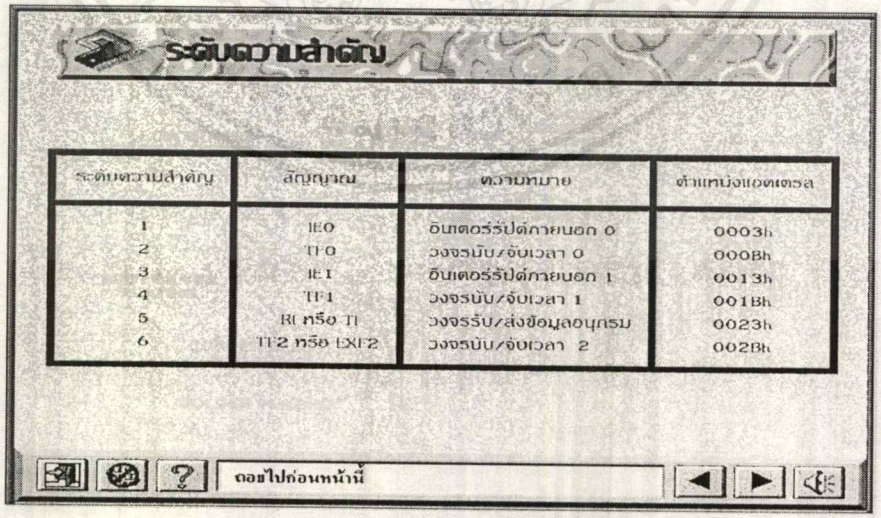
EX1 บิทนี้จะใช้ ENABLE หรือDISABLE สัญญาณที่เข้ามาทางขา INT0 ให้เกิดการขัดจังหวะ

ET1 TIMER 1 INTERRUPT ENABLE BIT ข้อมูลบิทนี้จะใช้ ENABLE หรือ DISABLE สัญญาณขัดจังหวะที่มาจากวงจร TIMER 1 (TF1)

ES ข้อมูลในบิทนี้จะ ENABLE หรือ DISABLE การขัดจังหวะจาก SERIAL PORT อันเนื่องจากมีข้อมูลเข้ามายัง SBUF ได้ส่งออกไปทาง SERIAL PORT หมดแล้ว

ET2 TIMER 2 INTERRUPT ENABLE BIT จะใช้งานเฉพาะใน 8052 และ 83152 เท่านั้นบิทนี้จะใช้ ENABLE หรือ DISABLE สัญญาณขัดจังหวะที่มาจากวงจร TIMER 2

EA บิทนี้จะควบคุมทั้ง 6 บิท ถ้าข้อมูลในบิทนี้เป็น 0 จะเป็นนนการ DISABLE ทุกบิทของสัญญาณอินเตอร์รัพท์ ทำให้ไม่เกิดการขัดจังหวะโปรแกรมได้เลยแต่ ถ้าบิทนี้เป็น 1 การ ENABLE หรือ DISABLE ใน 6 บิทที่กล่าวมาแล้วจะขึ้นกับข้อมูล ในแต่ละบิทนั้น



คำบรรยาย ลำดับความสำคัญของการอินเตอร์รัพท์ แหล่งสัญญาณอินเตอร์รัพท์แต่ละสัญญาณสามารถโปรแกรมได้ว่า ให้เป็นลำดับความสำคัญสูง หรือลำดับความสำคัญต่ำโดยที่ลำดับ

ความสำคัญต่ำจะถูกอินเตอร์รัพท์ด้วยสัญญาณอินเตอร์รัพท์ที่มีความสำคัญสูงกว่าและลำดับความสำคัญสูงจะไม่ถูกอินเตอร์รัพท์โดยสัญญาณอินเตอร์รัพท์ที่มีความสำคัญต่ำกว่า

ถ้ามีการอินเตอร์รัพท์ด้วยลำดับความสำคัญเท่ากันมากกว่า 1 สัญญาณ CPU จะทำการตรวจ (POLLING) และตัดสินใจว่าจะให้บริการกับสัญญาณอินเตอร์รัพท์ตัวใดฉะนั้นในแต่ละลำดับความสำคัญยังมีการจัดลำดับความสำคัญไว้อีกดังรายละเอียดข้างล่าง

SOURCE PIORITY WITHIN LEVEL

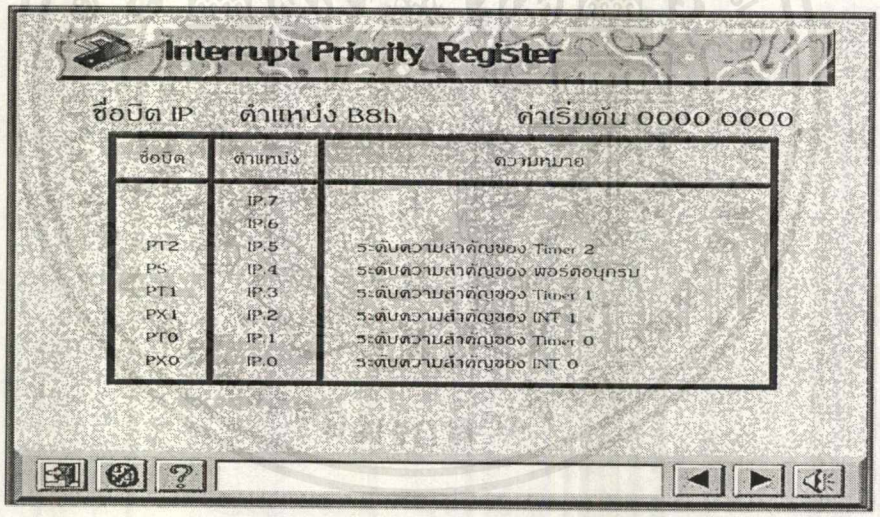
1. IE0 HIGHEST ,2. TF0 ,3. IE1 ,4. TF1 ,5. RI + TI LOWEST

การตอบสนองสัญญาณอินเตอร์รัพท์ CPU จะกระโดดไปที่ตำแหน่งของโปรแกรมบริการอินเตอร์รัพท์ตาม ชนิดของอินเตอร์รัพท์ ดังนี้

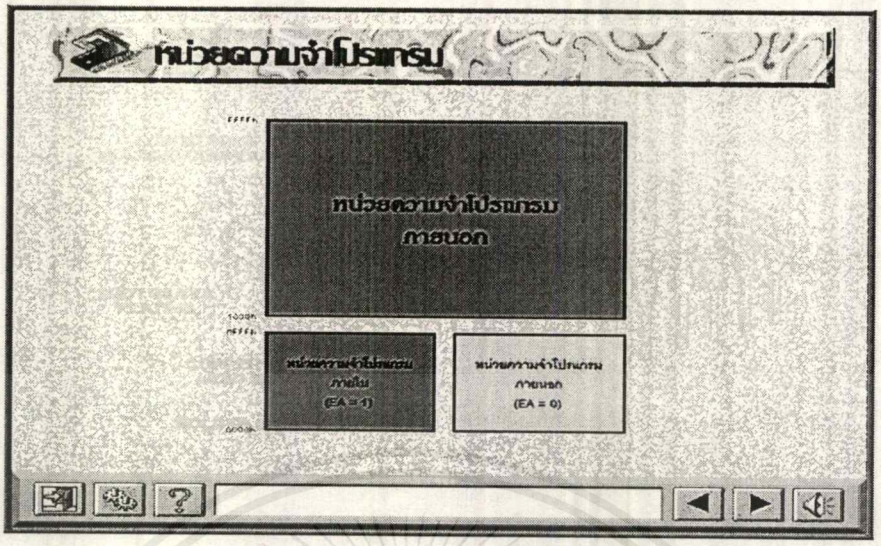
SOURCE VECTOR ADDRESS

1. IE0 0003H ,2. TF0 000BH ,3. IE1 0013H ,4. TF1 001BH

5. RI + TI 0023H



คำบรรยาย Interrupt Priority register เก็บค่ากำหนดลำดับความสำคัญของการอินเตอร์รัพท์ ชื่อบิต IP ตำแหน่ง B8h ค่าเริ่มต้น 0000 0000



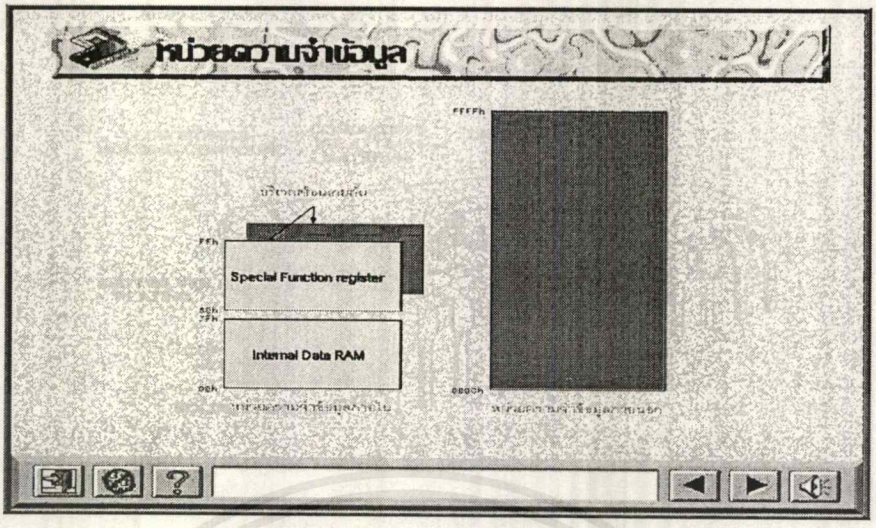
คำบรรยาย หน่วยความจำโปรแกรม หมายถึงหน่วยความจำที่อ่านได้อย่างเดียว(ROM) ซึ่งบรรจุโปรแกรมที่จะให้MCS-51 ทำงาน

ใน MCS-51 จะแบ่งหน่วยความจำโปรแกรมออกเป็น 2 ส่วนคือ

หน่วยความจำโปรแกรมภายในก็คือหน่วยความจำประเภท ROM หรือ EPROM ที่อยู่ภายในตัว MCS-51

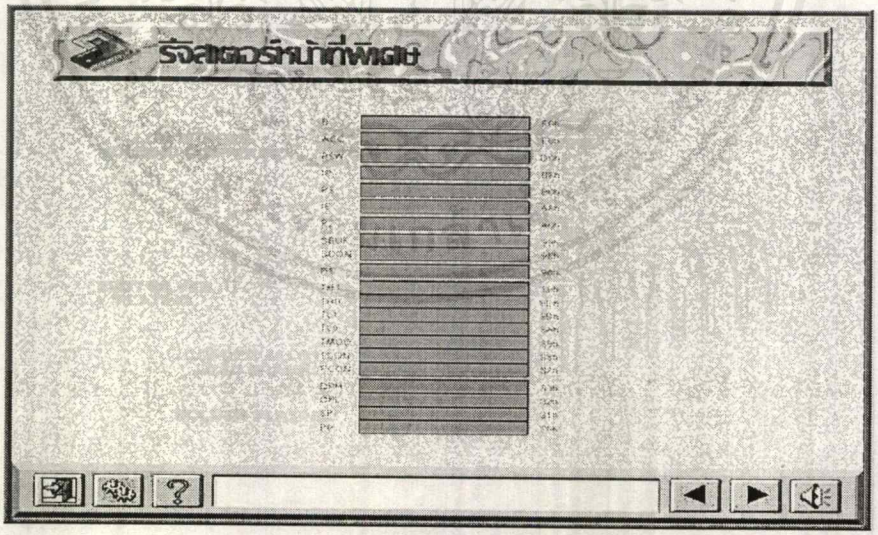
ส่วนหน่วยความจำโปรแกรมภายนอกก็คือ หน่วยความจำที่ต่ออยู่ภายนอกตัว MCS-51 โดยอาจจะเป็นการต่อเพื่อขยายหน่วยความจำเพิ่มเนื่องจากหน่วยความจำภายในไม่พอ หรืออาจต่อเป็นหน่วยความจำโปรแกรมภายนอกทั้งหมดเลยก็ได้ โดย MCS-51 สามารถเลือกให้รันโปรแกรมในหน่วยความจำโปรแกรมภายในหรือภายนอกก็ได้ โดยพิจารณาจากลอจิกของสัญญาณ EA

สัญญาณ EA(External Access)ใช้ในการกำหนดเลือกว่า จะอ่านข้อมูลมาจากหน่วยความจำโปรแกรมภายนอกหรือภายในตัวไอซีไมโครคอนโทรลเลอร์เอง ซึ่งหากเป็นระดับลอจิกต่ำจะอ่านข้อมูลมาจากหน่วยความจำโปรแกรมภายนอก และกรณีตรงข้ามจะอ่านข้อมูลมาจากหน่วยความจำโปรแกรมภายใน



คำบรรยาย หน่วยความจำข้อมูลหมายถึง หน่วยความจำที่เป็น RAM ซึ่งเราสามารถอ่านหรือเขียนข้อมูลเปลี่ยนแปลงได้ตลอดเวลา แต่ไม่สามารถรันโปรแกรมบนส่วนนี้ได้

หน่วยความจำข้อมูลของ MCS-51 ก็ถูกแบ่งเป็น 2 ส่วนเหมือนกันคือ หน่วยความจำข้อมูลภายในและภายนอก โดยหน่วยความจำข้อมูลภายนอกจะเข้าถึงได้ก็ด้วยคำสั่ง MOVX เท่านั้น สำหรับหน่วยความจำข้อมูลภายในจะมีทั้งหมด 256 ไบต์ โดยแบ่งเป็น 128 ไบต์ส่วนบน ซึ่งใช้เป็นที่อยู่ของรีจิสเตอร์ฟังก์ชันพิเศษ และอีก 128 ไบต์ส่วนล่าง ซึ่งถูกใช้งานทั่วไป



คำบรรยาย MCS-51 ประกอบด้วยรีจิสเตอร์ที่ทำหน้าที่พิเศษ "SFRs" (SPECIAL FUNCTION REGISTER)

แอกคิวมูเลเตอร์ (ACCUMULATOR) หรือ ACC เป็นรีจิสเตอร์ขนาด 8 บิท ทำหน้าที่ในการเก็บข้อมูลที่จะส่งให้กับหน่วยทำงานภายในซีพียูและเก็บผลลัพธ์ที่ได้จากการทำงานนั้น เป็นแอคคิวมูลเลเตอร์หรือ รีจิสเตอร์ A

รีจิสเตอร์ B ใช้ในคำสั่งคูณและ หาร ส่วนในคำสั่งอื่นสามารถใช้เหมือนกับรีจิสเตอร์ทั่วๆไปในการพักข้อมูล

แสต็กพอยเตอร์ (STACK POINTER) SP เป็นรีจิสเตอร์ขนาด 8 บิท เมื่อใช้คำสั่ง PUSH และ CALL SP จะเพิ่มขึ้นก่อนที่จะเก็บข้อมูลหรือแอดเดรส ซึ่ง STACK จะอยู่ที่ไหนก็ได้ในหน่วยความจำ (RAM) บน 8051 ของการ RESET แสตกจะมีอยู่ที่ 07H นั่นคือ แสตกจะเริ่มที่ 80H

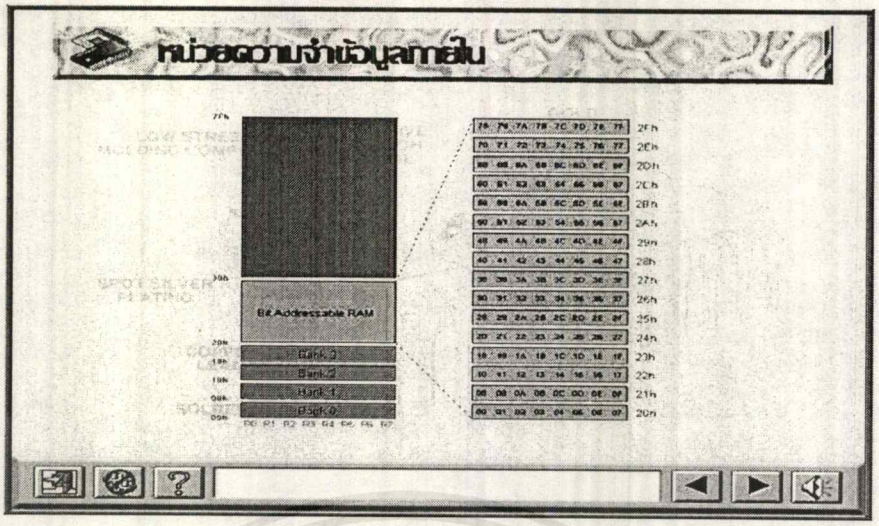
ดาต้าพอยเตอร์ (DATA POINTER) รีจิสเตอร์ DPTR ประกอบด้วย DPH และ DPL ซึ่งจะรวมกันเป็นรีจิสเตอร์ขนาด 16 บิท การเข้าถึง DPTR สามารถทำได้ทั้งแบบ 16 บิท และ แบบ 8 บิท หน้าที่ของ DPTR ถ้าใช้แบบ 16 บิท จะทำหน้าที่เป็นตัวชี้ข้อมูลที่อยู่ในหน่วยความจำภายนอก

รีจิสเตอร์ที่เกี่ยวข้องกับพอร์ท (PORT REGISTER) พอร์ท 0, พอร์ท 1, พอร์ท 2, และ พอร์ท 3 เป็น SFR ตัวหนึ่งที่สามารถแลทช์ข้อมูลได้ สามารถใช้เป็นอินพุทพอร์ทและเอ๊าท์พุทพอร์ตได้ทั้ง 3 พอร์ท

SERIAL DATA BUFFER แบ่งเป็นรีจิสเตอร์บัฟเฟอร์ในทางส่ง และบัฟเฟอร์ในทางรับ เมื่อมีการส่งข้อมูลภายในให้ SBUF ข้อมูลจะถูกส่งมาที่บัฟเฟอร์ในทางส่ง ที่ซึ่งจะใช้ทำการส่งข้อมูลอนุกรม การส่งข้อมูลให้ SBUF จะเป็นการเริ่มต้นการส่งด้วย และถ้าอ่านข้อมูลจาก SBUF ข้อมูลจะถูกอ่านจากบัฟเฟอร์ในทางรับ

TIMER REGISTERS รีจิสเตอร์คู่(TH0,TL0) และ (TH1,TL1) เป็นตัวจับเวลาและตัวนับขนาด 16 บิท

CONTROL REGISTERS รีจิสเตอร์หน้าที่พิเศษ IP, IE, TMOD, TCON, T2CON, SCON และ PCON ประกอบด้วยบิทควบคุมและบิทสถานะสำหรับระบบการอินเตอร์รัพท์

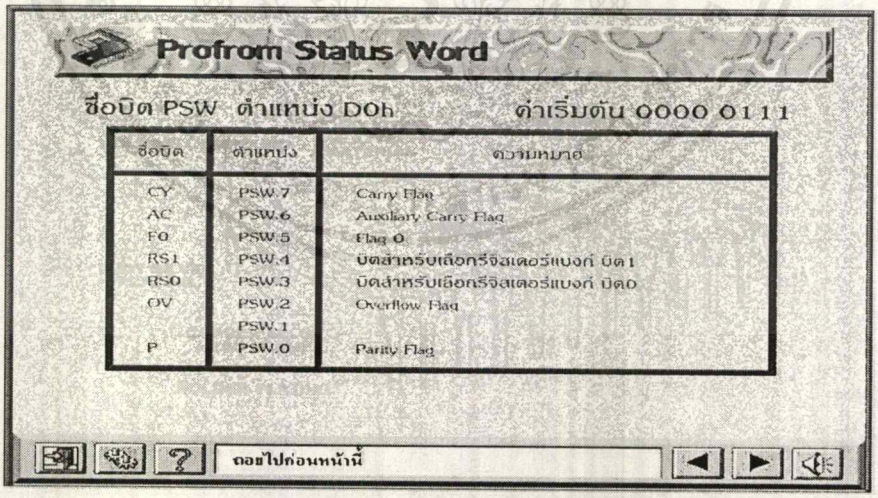


คำบรรยาย หน่วยความจำภายใน (Internal Data Ram) หน่วยความจำตำแหน่ง Address 00h - 7Fh ยังจำแนกได้อีก 3 ส่วนตามประเภทการใช้งานได้ดังนี้

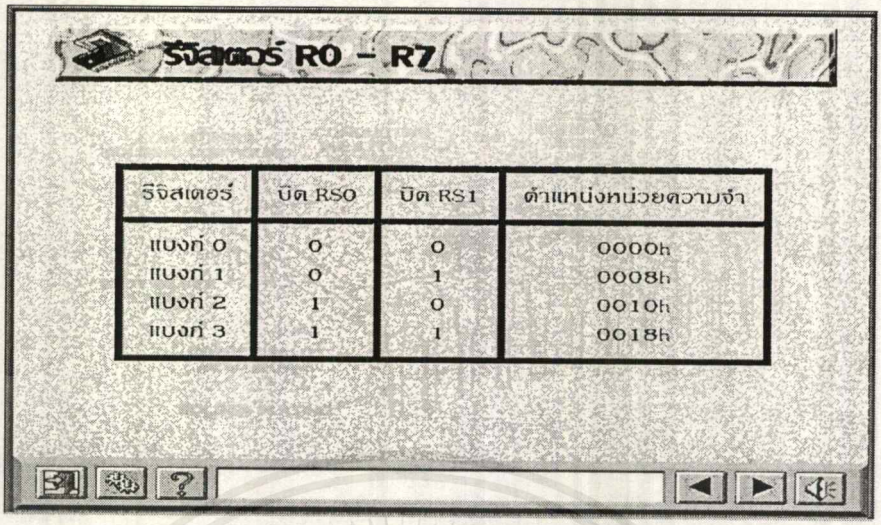
บริเวณ Address 30h - 7Fh เป็นบริเวณที่สามารถนำไปใช้ได้อย่างอิสระ

บริเวณ Address 20h - 2Fh จำนวน 16 byte บริเวณพื้นที่เป็นของผู้ใช้ จะต่างจากหน่วยความจำอื่นๆ คือ ผู้ใช้สามารถอ้างเป็น bit หรือ byte ข้อมูลได้โดยตรง

บริเวณ Address 00h - 1Fh จำนวน 32 byte จำแนกเป็นกลุ่มข้อมูลละ 8 bit รวมทั้งหมด 4 กลุ่ม แต่ละกลุ่มจะถูกใช้เหมือน Register ทั่วไป R0 - R7

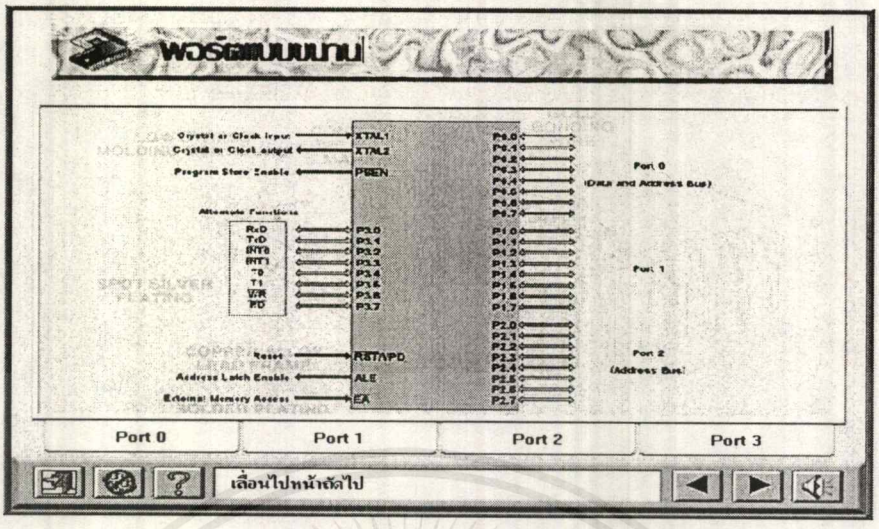


คำบรรยาย Perform status word เก็บค่ากำหนดแบงค์ใช้งานรีจิสเตอร์ทั่วไป และสถานะของ Flag จากการทำงานชื่อบิต PSW ตำแหน่ง D0h ค่าเริ่มต้น 0000 0000



คำบรรยาย รีจิสเตอร์ R0 - R7 การใช้งานในแบงค์ไหนก็ตามจะมีชื่อว่า R0 - R7 ดั้งนั้นในการใช้งานต้องกำหนดว่าจะใช้งานจากแบงค์ใด ส่วนแบงค์อื่นสามารถสามารถอ้างถึงได้โดยแอดเดรสนั้นๆโดยตรง การกำหนดโดยการเลือกแต่ละกลุ่มของรีจิสเตอร์ทำได้โดยง่ายเพียกกำหนดบิตที่อยู่ในรีจิสเตอร์ PSW

คำบรรยาย วงจรออสซิเลเตอร์ MCS-51 มีวงจรออสซิลเลเตอร์อยู่ภายใน สำหรับการสร้างพัลส์ของสัญญาณนาฬิกา ซึ่งจะนำไปเป็นฐานเวลา หรือการกำหนดจังหวะการทำงานของหน่วยการทำงานทั้งหมดให้สอดคล้องกัน (Synchronization) โดยปกติแล้วก็มักจะทำโดยการใช้คริสตอลซึ่งเชื่อมต่อเข้ากับขาของสัญญาณ XTAL1 และ XTAL2 พร้อมกับตัวเก็บประจุ หรืออาจจะเป็นสัญญาณนาฬิกาจากภายนอก

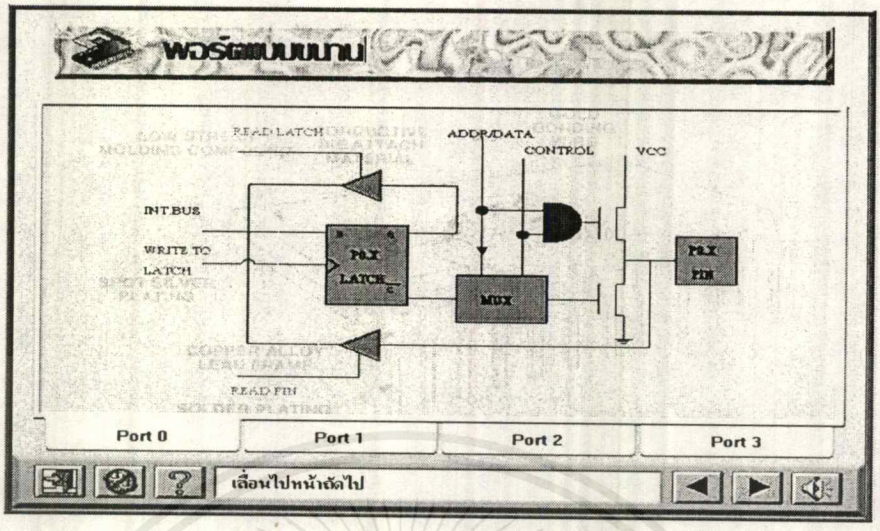


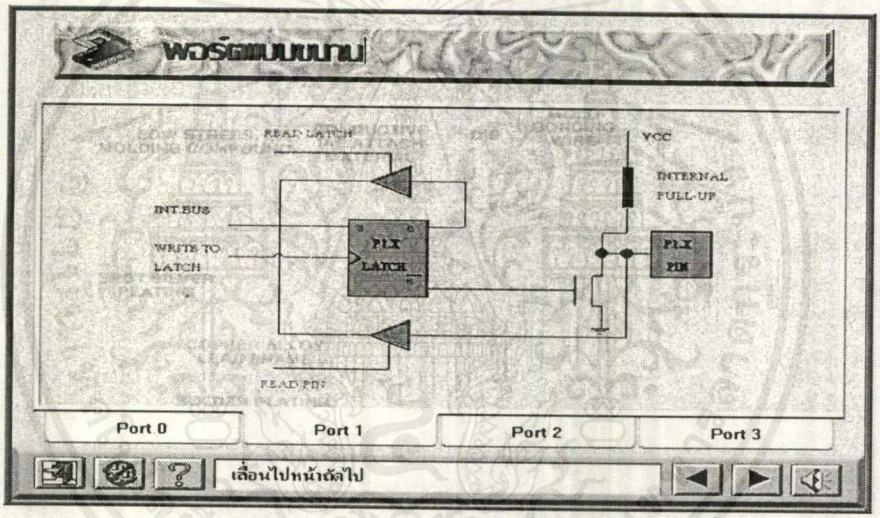
คำบรรยาย โครงสร้างและการทำงานของพอร์ต MCS-51 มี I/O พอร์ท อยู่ 4 พอร์ท โดยแต่ละพอร์ทจะเป็นพอร์ตแบบ 2 ทิศทาง มีการแลทช์ข้อมูลได้ รวมทั้งมีวงจรขับทางด้านเอาท์พุท และบัฟเฟอร์ทางด้านอินพุทพอร์ท 0 และ พอร์ท 2 ใช้สำหรับติดต่อกับหน่วยความจำภายนอก ในการใช้ MCS-51 ติดต่อกับหน่วยความจำภายนอก

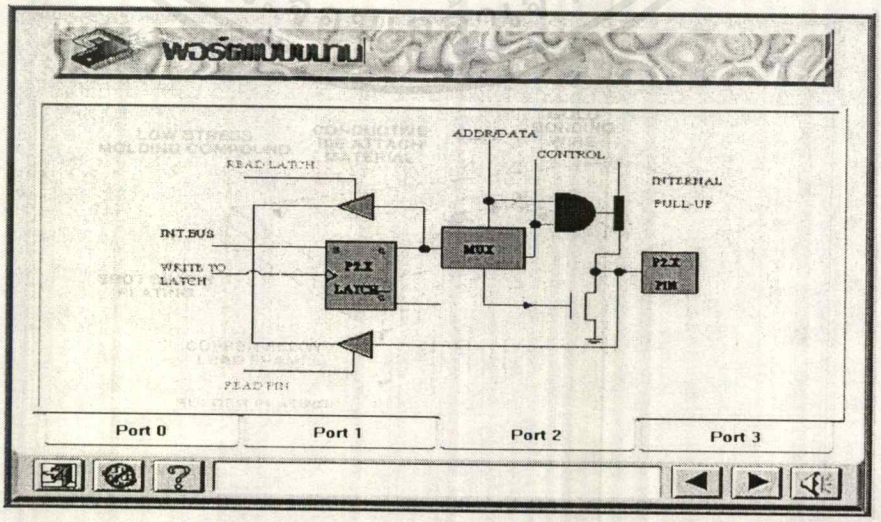
พอร์ท 0 จะให้เอาท์พุทเป็น LOW BYTE ของแอดเดรสของหน่วยความจำภายนอกและจำทำการ MULTIPLEX กับข้อมูลที่จะเขียนหรืออ่าน สรุปคือ P0 จะเป็นได้ทั้ง ADDRESS และ DATA ส่วน พอร์ท 2 จะให้แอดเดรสไบท์สูง (MSB) ของหน่วยความจำภายนอก

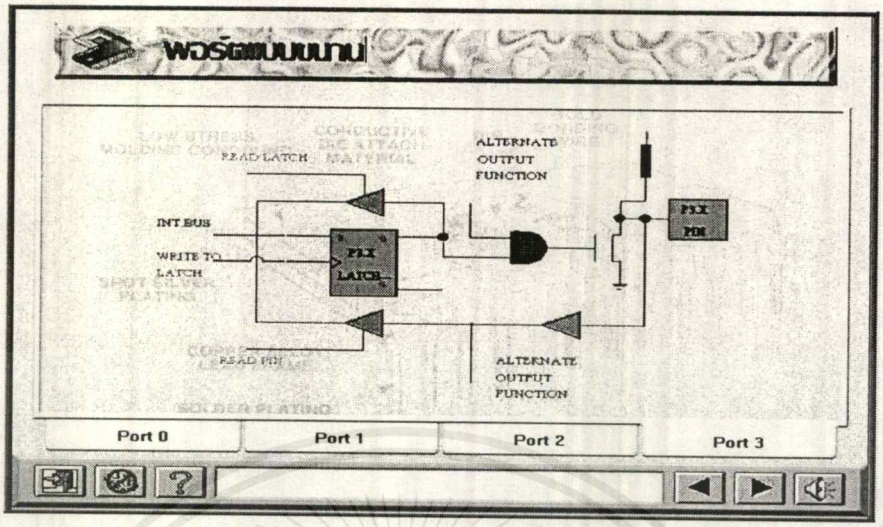
พอร์ท 3 ทั้งหมดจะทำงานหลายหน้าที่ ดังรายละเอียดต่อไปนี้

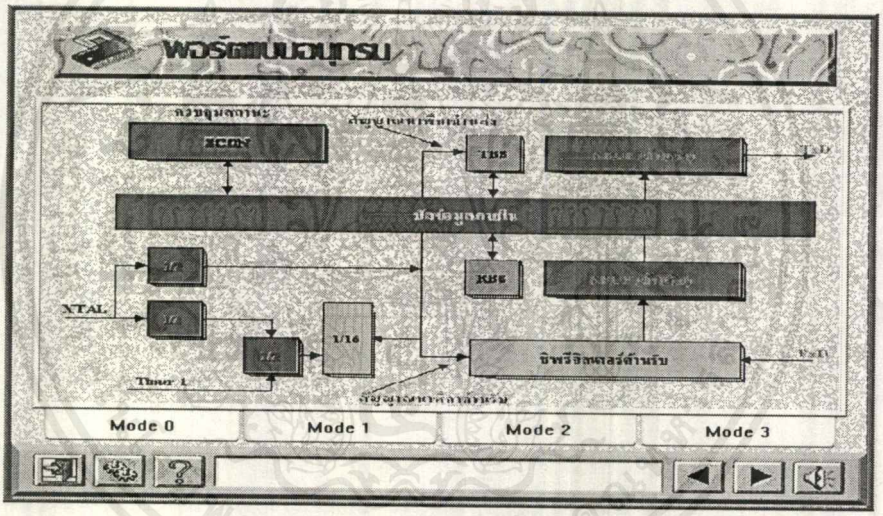
P3.0	RXD (SERIAL INPUT PORT)
P3.1	TXD (SERIAL OUTPUT PORT)
P3.2	INT0 (EXTERNAL INTERRUPT)
P3.3	INT1 (EXTERNAL INTERRUPT)
P3.4	T0 (TIMER/COUNTER 0 EXTERNAL INPUT)
P3.5	T1 (TIMER/COUNTER 1 EXTERNAL INPUT)
P3.6	WR (EXTERNAL DATA MEMORY WRITE STROBE)
P3.7	RD (EXTERNAL DATA MEMORY READ STROBE)





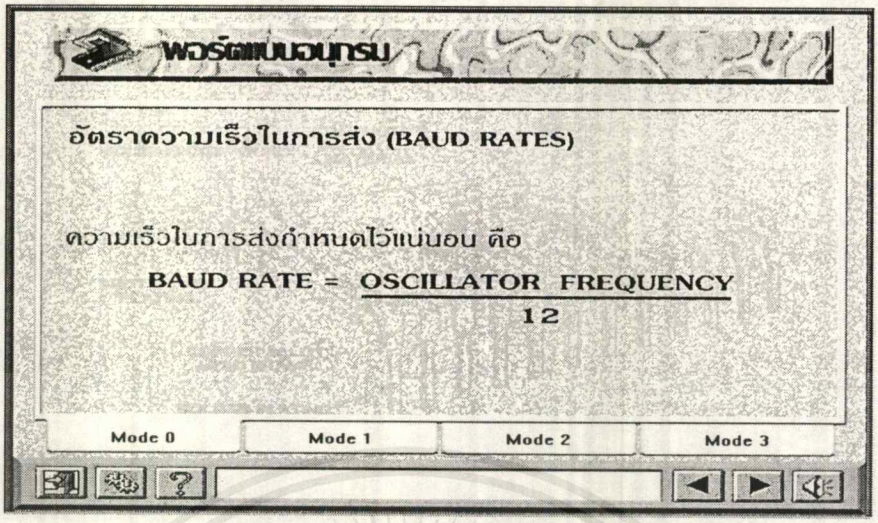






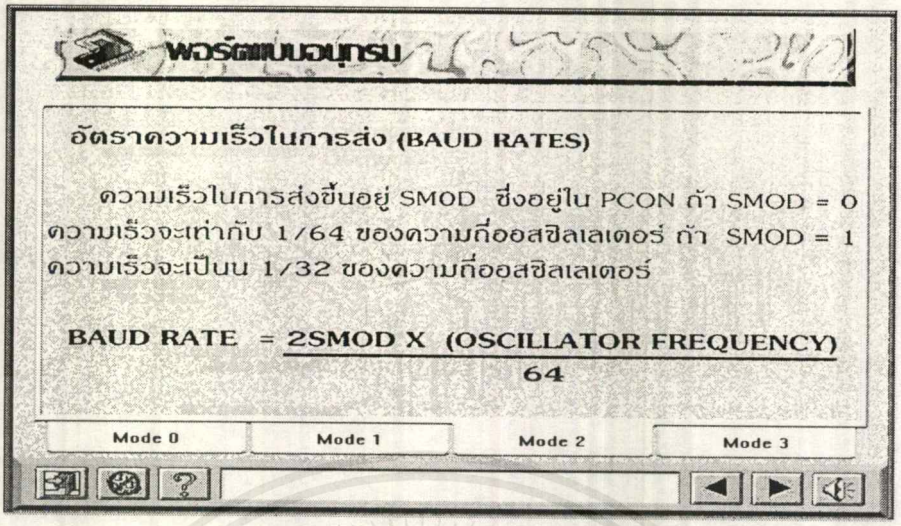
คำบรรยาย การสื่อสารข้อมูลแบบอนุกรม (Serial Port) พอร์ตอนุกรมนี้เป็น Full Duplex ที่สามารถรับข้อมูลใน Byte ทีสองได้โดยที่ Byte แรกยังไม่ถูกอ่านออกไปจาก Buffer แต่อย่างไรก็ตามข้อมูล Byte แรกจะต้องถูกอ่านไปก่อนที่การรับข้อมูลใน Byte ที่สองจะเสร็จสมบูรณ์มิฉะนั้นข้อมูล Byte แรกจะสูญเสียไป (ถูกทับด้วยข้อมูลที่ตามมา)ข้อมูลที่จะใช้ในการส่งและรับจะถูกพักไว้ ณ ที่เดียวกันคือ SBUF

การเขียนข้อมูลไปที่ SBUF จะเป็นการโหลดข้อมูลให้กับ TRANSMIT REGISTER และการอ่าน SBUF จะเป็นการอ่านข้อมูลจาก RECEIVER REGISTER พอร์ทอนุกรมแบ่งการทำงานออกเป็น 4 โหมด



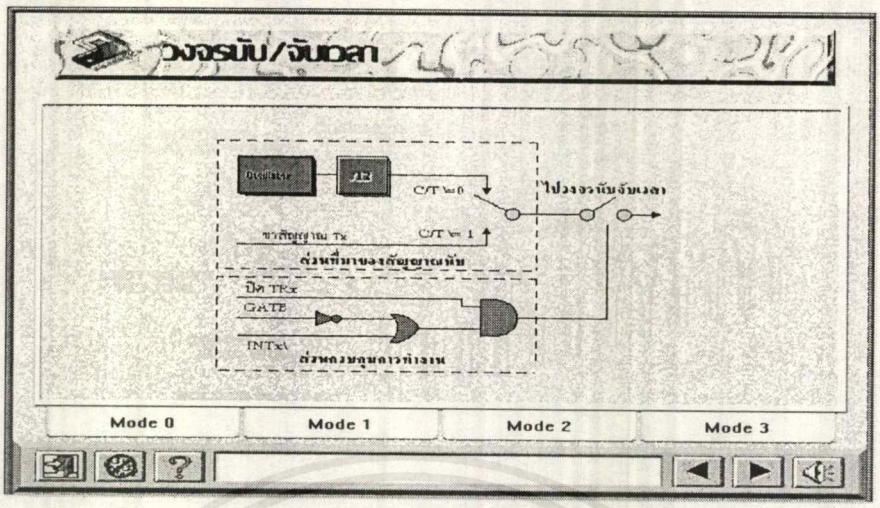
โหมด 0 ข้อมูลจะเข้ามาทาง RXD ส่วนข้อมูลทางออกจะออกทาง TXD ความเร็วในการส่ง (Baud Rate) จะถูกกำหนดตายตัวเป็น 1/12 ของความถี่ออสซิลเลเตอร์ของระบบ ในโหมด 0 จะเป็นการส่งข้อมูลขนาด 8 บิท (โดย LSB ออกไปก่อน)

โหมด 1 ส่งและรับข้อมูลขนาด 10 บิท ซึ่งประกอบด้วย Start Bit (0), ข้อมูล 8 บิท (LSB ออกก่อน)Stop Bit ในขณะรับข้อมูล Stop Bit จะถูกส่งให้ RB8 ในรีจิสเตอร์หน้าที่พิเศษ SCON ความเร็วในการส่งไม่กำหนดตายตัว



โหมด 2 ส่งและรับข้อมูลขนาด 11 บิท ประกอบด้วย Start Bit (0), ข้อมูล 8 บิท (LSB ออกก่อน), ข้อมูลบิทที่ 9 ที่สามารถโปรแกรมได้ และอีก 1 STOP BIT บิทที่ 9 ของข้อมูลสามารถ SET เป็น 0 หรือ 1 ก็ได้ประโยชน์อาจใช้เป็นตัวส่งพาริตี้บิทโดยนำค่าของแฟลก P ใน PSW มาไว้ใน TB8 และในขณะทำการรับข้อมูลบิทที่ 9 ของข้อมูลจะถูกโหลดเข้าไปที่ RB8 ของ SCON ความเร็วในการส่งจะถูกโปรแกรมเป็น 1/32 หรือ 1/64 ของออสซิลเลเตอร์

โหมด 3 การทำงานเหมือกับโหมด 2 เพียงแต่ความเร็วในการส่งไม่กำหนดตายตัว



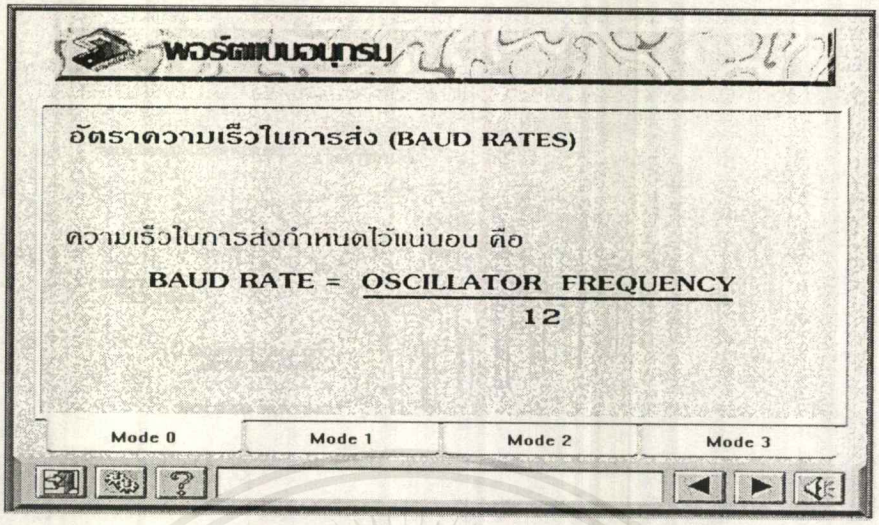
คำบรรยาย วงจรนับ/จับเวลา (Timer/Counter)8051 มีอยู่ 2 ตัว คือ T0 และ T1

กำหนดให้ทำงานเป็น Timer รีจิสเตอร์จะเพิ่มค่าทุกๆแมชชีนไซเคิลการทำงานของซีพียู กล่าวได้ว่าการทำงานเป็นตัวนับหน่วยเวลาจากออสซิเลเตอร์ของซีพียู โดยผ่าวงจรหาร 12 ของคาบเวลา ฉะนั้นอัตราการนับในแต่ละครั้งเท่ากับ 1/12 เท่าของความถี่ออสซิเลเตอร์

กำหนดให้ทำงานเป็น Counter สัญญาณ INPUT ที่จะป้อนให้ Counter นั้นทำงานที่ขอบขาลง (1 TO 0) คือ ต้องเป็นพัลส์ HIGH 1 แมชชีนไซเคิลและเป็น LOW 1 แมชชีนไซเคิล ฉะนั้นความถี่สูงสุดที่ Counter จะนับได้นั้นประมาณ 1/24 ของความถี่ออสซิเลเตอร์

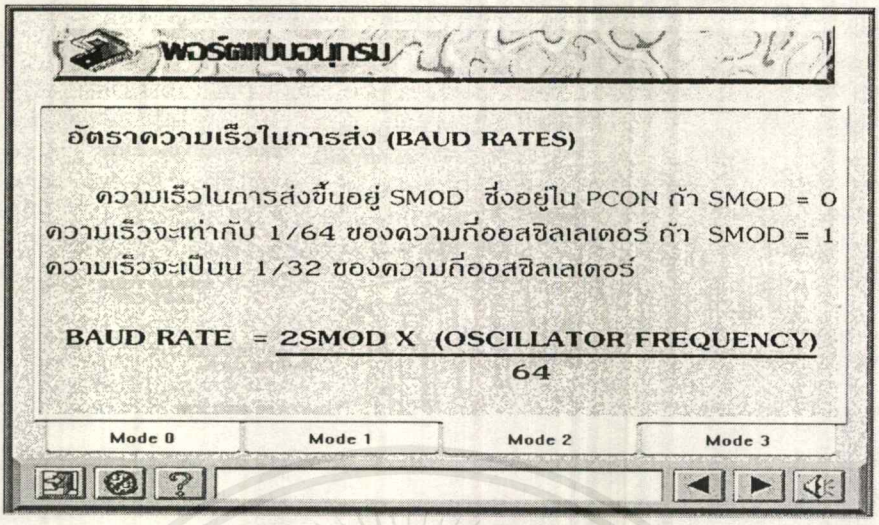
TIMER/COUNTER จะทำงานได้ก็ต่อเมื่อ TRx = 1 และ GATE = 0 หรือ INTx = 1 (ถ้าเซ็ต GATE = 1 TIMER/COUNTER จะถูกควบคุมด้วยสัญญาณ INT1 จากภายนอก ประโยชน์ในการทำงานแบบนี้คือ ใช้วัดความกว้างของพัลส์จากอินพุทภายนอก)

การทำงานของ TIMER/COUNTER แบ่งเป็น 4 โหมด



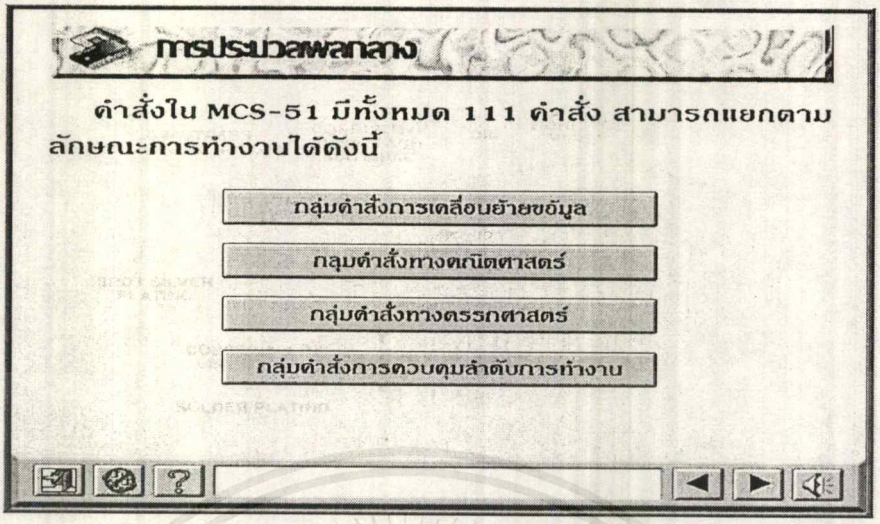
คำบรรยาย MODE 0 Timer/Counter การทำงานในโหมดนี้รีจิสเตอร์ถูกกำหนดให้เป็นแบบ 13 บิท โดยการนับจากค่าที่ทุกบิทเป็น HIGH ไปจนทุกๆ บิทเป็น 0 เกิด overflow และจะให้สัญญาณอินเตอร์รัพท์ โดยเซ็ตแฟลก TF0 หรือ TF1 การที่จะให้ TIMER/COUNTER ตัวใดอยู่ในโหมดใดนั้นกำหนดได้จากรีจิสเตอร์ TMOD

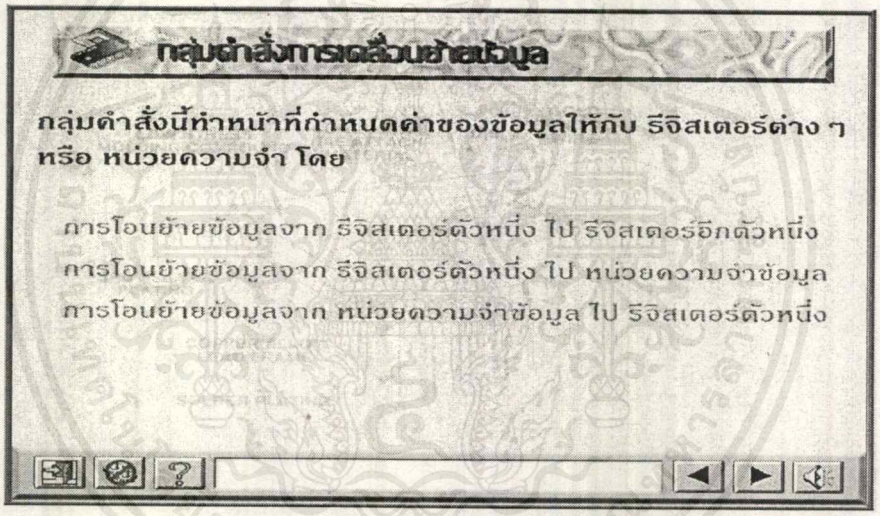
คำบรรยาย MODE 1 Timer/Counter การทำงานในโหมดนี้รีจิสเตอร์ถูกกำหนดให้เป็นแบบ 16 บิท โดยการนับจากค่าที่ทุกบิทเป็น HIGH ไปจนทุกๆ บิทเป็น 0 เกิด overflow และจะให้สัญญาณอินเตอร์รัพท์ โดยเซ็ตแฟลก TF0 หรือ TF1

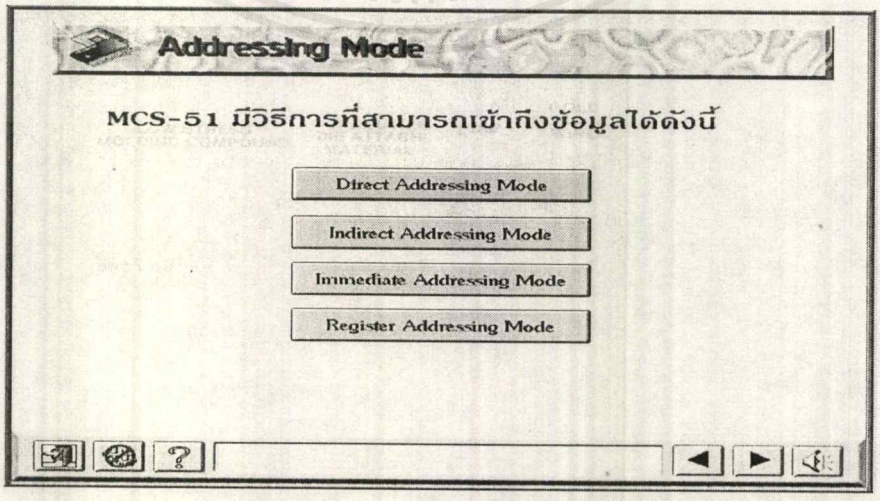


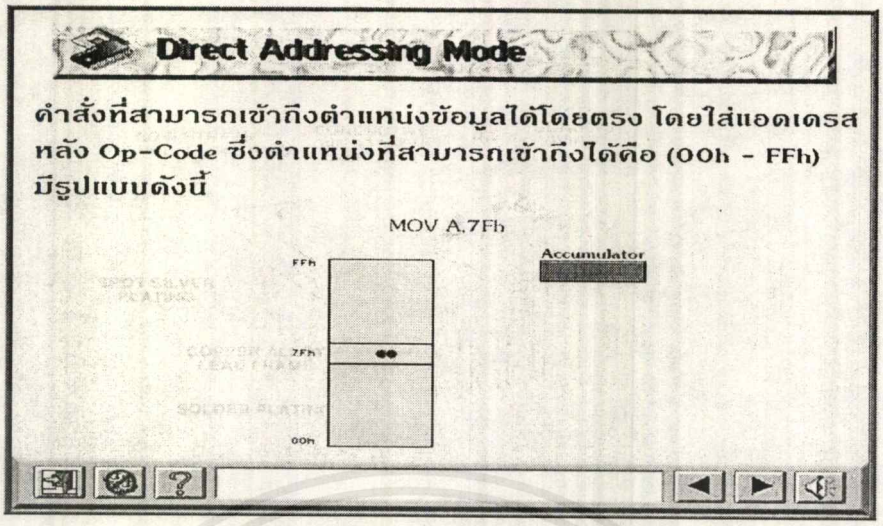
คำบรรยาย MODE 2 Timer/Counter การทำงานในโหมดนี้รีจิสเตอร์ จะเป็นแบบ 8 บิท โดยที่ TLx จะสามารถโหลดข้อมูลจาก THx ได้ใหม่(AUTO-RELOAD) เมื่อเกิด overflow จาก TLx โดยที่ค่าใน THx จะไม่ถูกเปลี่ยนการทำงานอื่นๆจะเหมือนกับโหมด 0

คำบรรยาย MODE 3 Timer/Counter การทำงานในโหมดนี้จะใช้ได้เฉพาะ Timer 0 เท่านั้น โดยแยก TL0 และ TH0 ของ TIMER 0 ใช้โดยอิสระ TL0 จะใช้บิทควบคุมคือ GATE, TR0, INT0, TF0 และ TH0 ถูกใช้เป็น TIMER (นับแมชชีนไซเคิล) และรับช่วงการใช้ TR1 และ TF1 ของ TIMER 1 ฉะนั้นในโหมด 3 นี้ TH0 จะควบคุมการอินเตอร์รัพท์ของ TIMER 1 (TF1) เมื่อใช้ TIMER 0 ในโหมด 3 แล้ว TIMER 1 สามารถจะสลับใช้ระหว่างโหมด 3 และโหมดอื่นได้ หรือใช้เป็น BAUD RATE GENERATOR





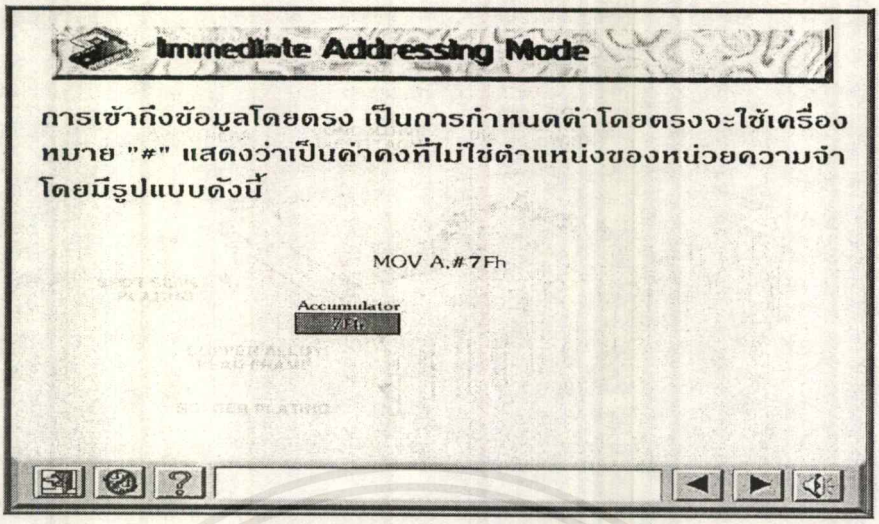




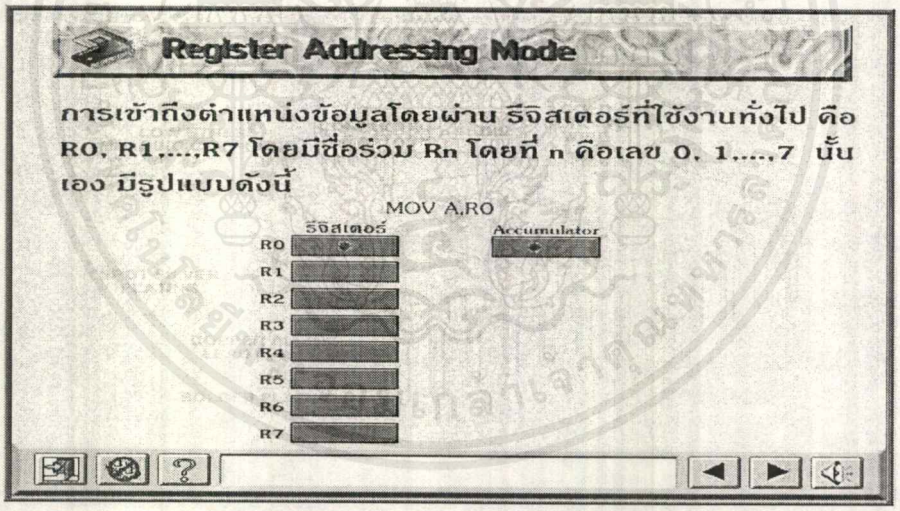
คำบรรยาย Direct Addressing Mode เป็นคำสั่งที่สามารถเข้าถึงตำแหน่งข้อมูลได้โดยตรง โดยใส่ Address หลัง Opcode ซึ่งตำแหน่งที่สามารถเข้าถึงได้คือ00-FFh ซึ่งมีรูปแบบดังตัวอย่าง

MOV A, 7Fh จะเป็นการบอกถึงตำแหน่ง Address ที่จะไปนำข้อมูลต้นทางมาซึ่งเป็น Address 7Fh ของหน่วยความจำข้อมูลภายใน 8051 มาเก็บไว้ใน Register A

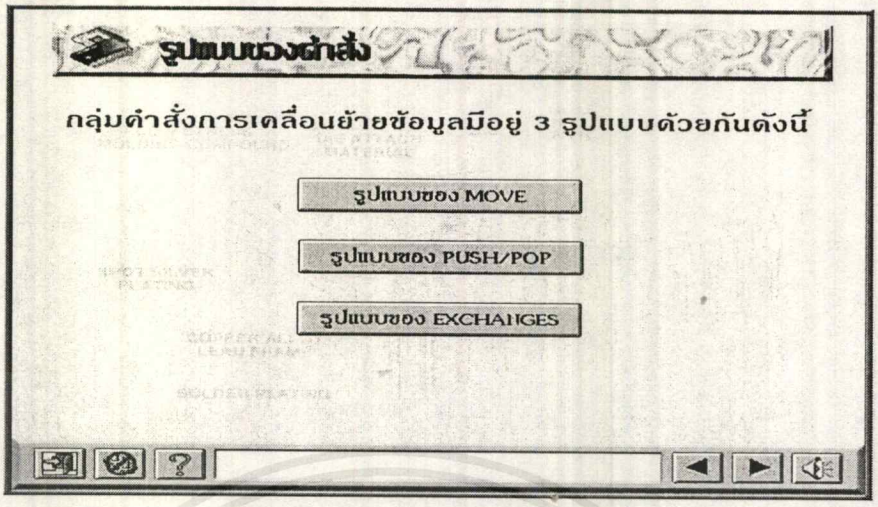
คำบรรยาย Indirect Addressing Mode คือการเข้าถึงข้อมูลโดยอ้อม วิธีนี้จะระบุตำแหน่งโดยผ่านทาง Register R0 และ R1 เท่านั้น ส่วน DPTR สามารถอ้างหน่วยความจำภายนอกได้โดยใช้สัญลักษณ์ @ นำหน้า โดยมีรูปแบบดังนี้คือ MOV A, @R0



คำบรรยาย Immediate Addressing Mode คือการเข้าถึงข้อมูลโดยตรง การกำหนดค่าจะใช้เครื่องหมาย # แสดงค่าเป็นค่าคงที่ไม่ใช่ตำแหน่งของหน่วยความจำ โดยมีรูปแบบของคำสั่งคือ MOV A, 7F จะเป็นการนำค่าข้อมู 7F ไปยัง Accumulator โดยตรง

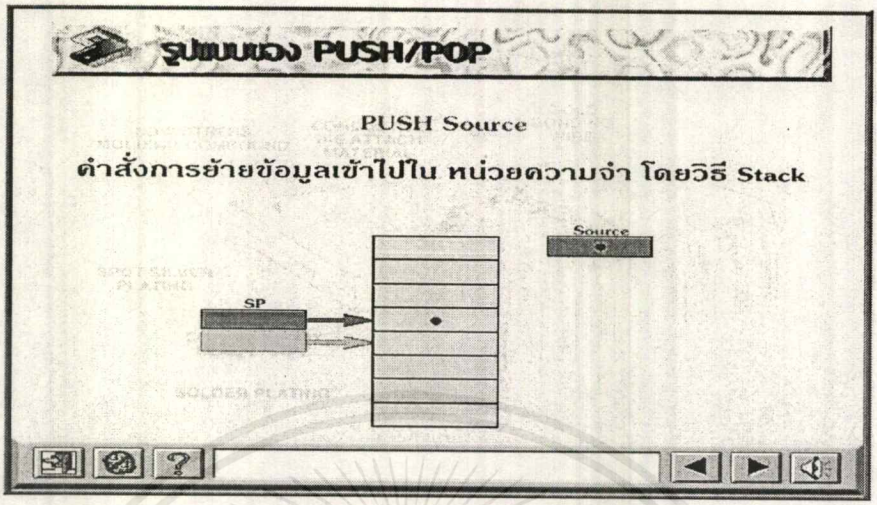


คำบรรยาย Register Addressing Mode เป็นการอ้างตำแหน่งของข้อมูลที่จะดำเนินการมาจาก Register ดังนั้นจึงมีการใช้ชื่อของ Register ในส่วนของ Operand ต้นทางหรือปลายทางของข้อมูล อย่างไรก็ตาม Register ที่จะใช้ในการอ้าง Address แบบนี้คือ Register A,Register DPTR, Register R0 - R7 จากตัวอย่างจะเป็นการ MOV A, R0



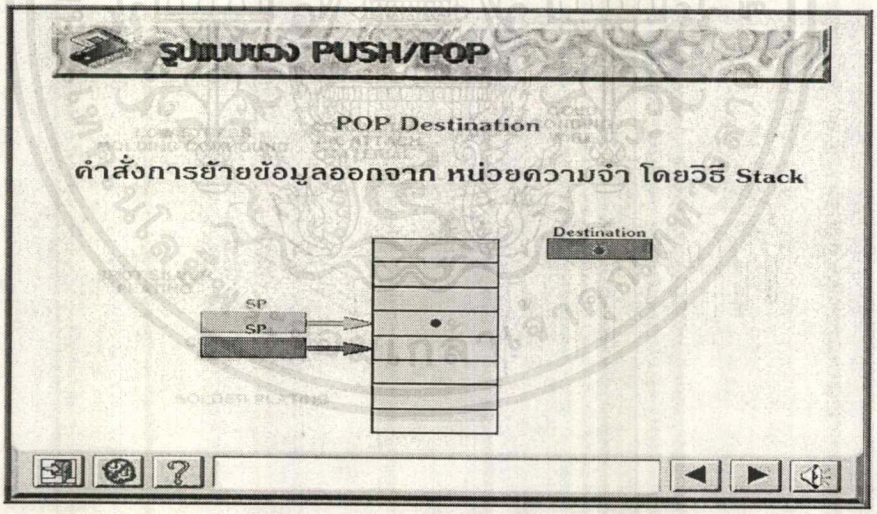
คำบรรยาย รูปแบบของคำสั่งการโอนย้ายข้อมูล กลุ่มคำสั่งนี้มีอยู่ 3 รูปแบบคือ 1.รูปแบบของคำสั่ง MOV 2.รูปแบบของคำสั่ง PUSH, POP 3. รูปแบบของคำสั่ง EXC

คำบรรยาย รูปแบบของการ MOVE คำสั่ง MOV จะเป็นการย้ายข้อมูลจากต้นทาง(Source) ไปยังปลายทาง(Destination) ก็คือจะเป็นคำสั่งการย้ายข้อมูลจาก Register หรือ หน่วยความจำ ดังตัวอย่างในภาพจะเป็นการย้ายข้อมูลจาก Source ไปยัง Destination

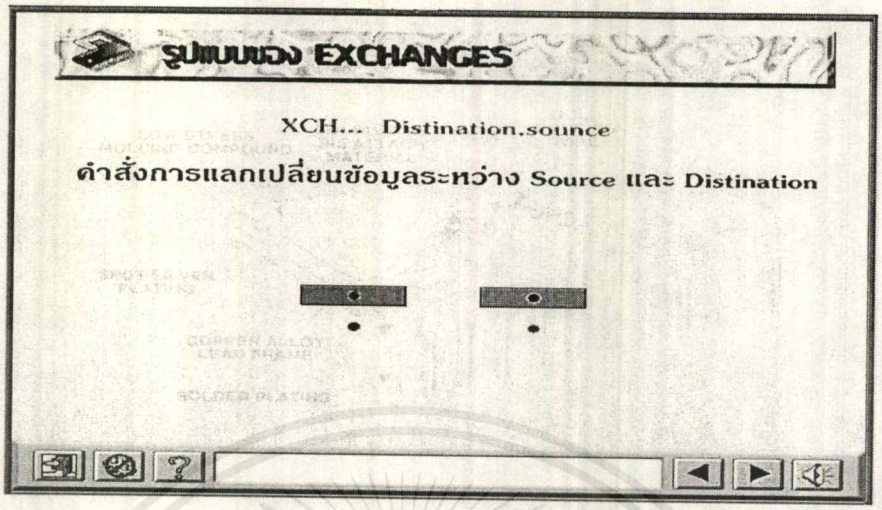


คำบรรยาย รูปแบบของคำสั่ง PUSH , POP เป็นคำสั่งการโอนย้ายข้อมูลเข้าไปในหน่วยความจำโดยวิธี stack โดยมีรูปแบบคือ

PUSH Source คือนำค่าที่อยู่ในตำแหน่ง Source ไปใส่ไว้ใน stack

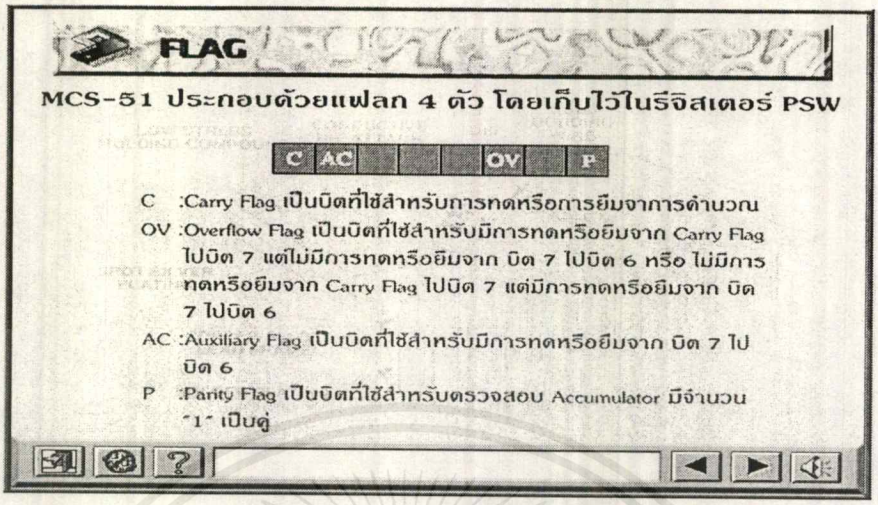


POP Destination เป็นคำสั่งการโอนย้ายข้อมูลจากหน่วยความจำโดยวิธี stack คือจะนำค่าที่เก็บไว้ใน stack ออกไปไว้ยังตำแหน่ง Destination



คำบรรยาย รูปแบบของคำสั่ง XCH มีการแลกเปลี่ยนข้อมูลระหว่างตำแหน่งต้นทางและปลายทางคือมีการสลับตำแหน่งข้อมูลระหว่างต้นทางและปลายทาง

คำบรรยาย กลุ่มคำสั่งทางคณิตศาสตร์ กลุ่มคำสั่งนี้ทำหน้าที่คำนวณ เป็นการนำข้อมูล 2 ตัว มากระทำทางคณิตศาสตร์ โดยนำผลการคำนวณไปเก็บไว้ยัง Accumulator



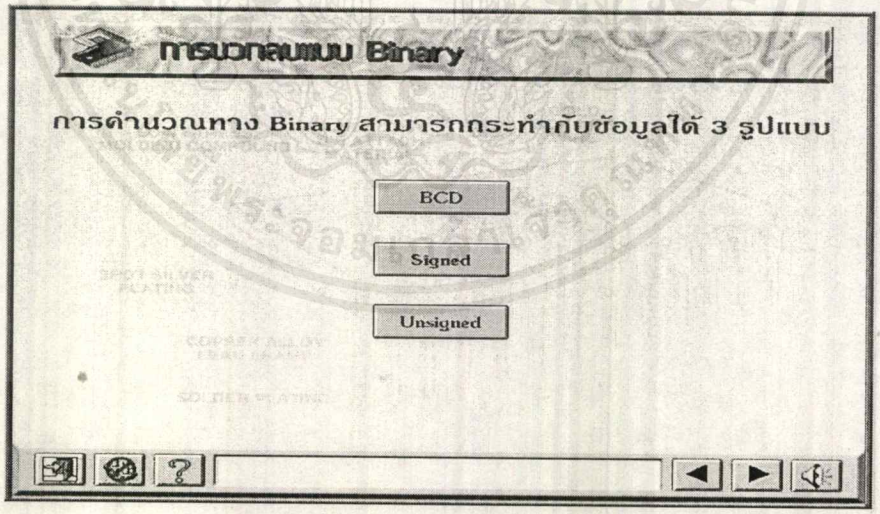
คำบรรยาย MCS-51 ประกอบด้วย Flag 4 ตัวโดยเก็บไว้ใน PSW โดยตัวแรกคือ

C Carry Flag เป็น bit ที่ใช้ในการทดหรือการยืมจากการคำนวณ

OV Overflow Flag เป็น bit ที่ใช้สำหรับเมื่อมีการทด

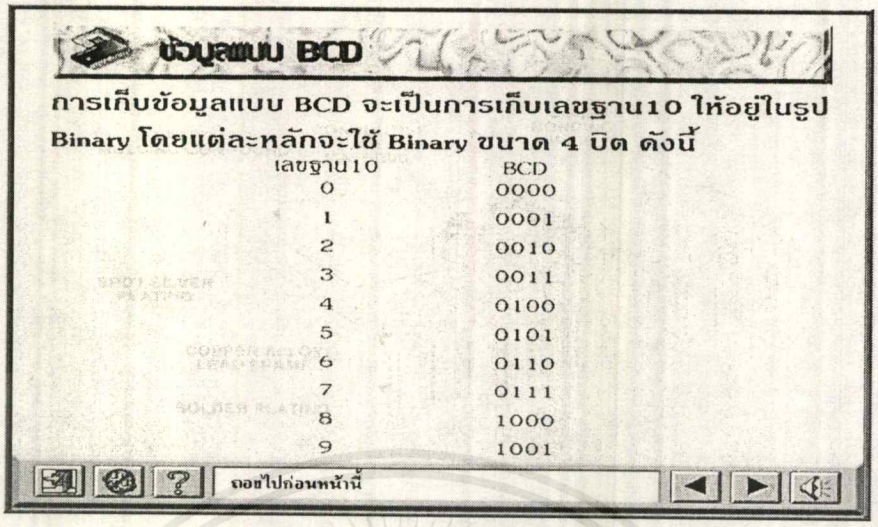
AC Auxireary Flag เป็น bit ที่ใช้เมื่อมีการทดหรือยืมจาก bit 7 ไป bit 6

P Parity Flag เป็น bit ที่ใช้สำหรับ Accumulator มี จำนวนเลข 1เป็นจำนวนคู่



คำบรรยาย การบวกลบแบบ binary การคำนวณทาง Binary สามารถกระทำกับข้อมูลได้ 3 แบบคือ

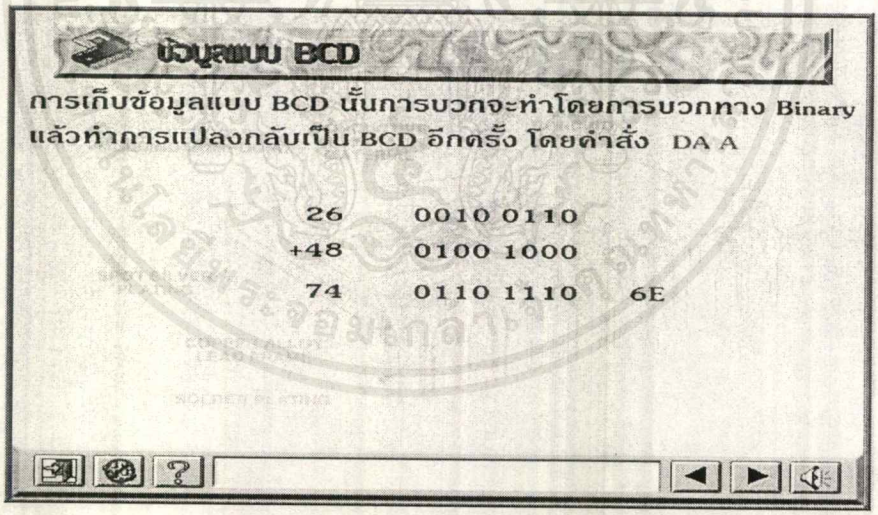
1.ข้อมูลแบบ BCD 2.ข้อมูลแบบ Signed 3.ข้อมูลแบบ Unsigned



คำบรรยาย ข้อมูลแบบ BCD การเก็บข้อมูลแบบ BCD จะเป็นการเก็บข้อมูลเลขฐานสิบให้อยู่ในรูป Binary โดยแต่ละหลักจะใช้ binary ขนาด 4 บิท แทนด้วยเลขฐานสิบนั้นๆ ตัวอย่างคือ

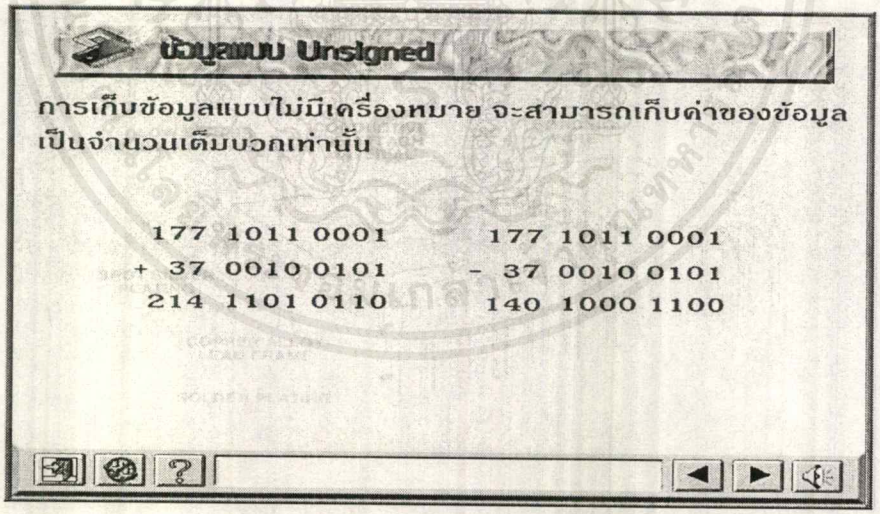
เลขฐานสิบ 0 จะแทนด้วยเลข binary คือ 0000

เลขฐานสิบ 1 จะแทนด้วยเลข binary คือ 0001 และไล่อย่างนี้ไปเรื่อยๆดังแสดงในภาพ



คำบรรยาย ข้อมูลแบบ BCD การเก็บข้อมูลแบบ BCD นั้น การบวกจะทำโดยการบวกทาง binary ก่อนแล้วทำการแปลงกลับเป็น BCD อีกครั้งด้วยคำสั่ง DA A ดังตัวอย่าง

คำบรรยาย ข้อมูลแบบ Signed หรือแบบมีเครื่องหมาย การเก็บค่าขอลข้อมูลแบบมีเครื่องหมายสามารถเก็บค่าของข้อมูลเป็นจำนวนเต็มบวกหรือจำนวนเต็มลบ โดยเครื่องหมายจะใช้บิทสูงสุด (MSB) เป็นบิทแสดงเครื่องหมายเป็นลบข้อมูลที่มีค่าเป็นลบจะถูกเก็บไว้ในรูปtwo complement โดยวิธีทำให้เป็น Two complement คือ กลับค่าบิททุกบิท เป็นตรงกันข้าม แล้วบวกด้วยตัวเลข 1 ก็จะได้ค่าของ two complement

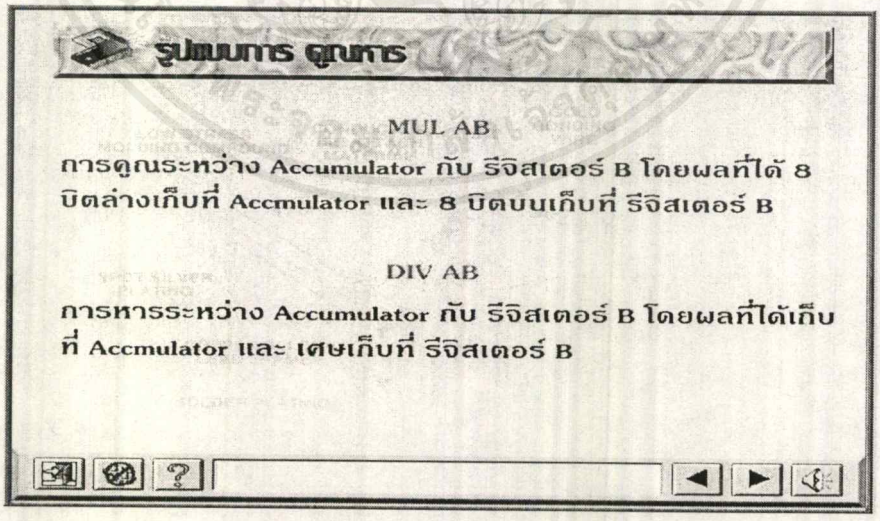


คำบรรยาย ข้อมูลแบบ Unsign คือข้อมูลแบบไม่มีเครื่องหมาย จะสามารถเก็บค่าของข้อมูลจำนวนเต็มบวกเท่านั้น

คำบรรยาย คำสั่ง ADD A, DATA เป็นการบวกค่าระหว่าง Accumulator กับข้อมูลที่กำหนด คือ การนำข้อมูล DATA มาบวกเข้ากับข้อมูลที่เก็บใน Accumulator แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ที่ Accumulator ดังเดิม

คำสั่ง ADDC A, DATA เป็นการบวกค่าระหว่าง Accumulator กับข้อมูลที่กำหนดและบิททดที่ Carry flag ก็คือการนำข้อมูลมาบวกเข้ากับข้อมูลที่เก็บใน Accumulator รวมทั้งบิททดที่ Carry flag แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ที่ Accumulator ดังเดิม

คำสั่ง SUBB A, DATA คือการลบข้อมูลระหว่าง Accumulator กับข้อมูลที่กำหนดและบิทยืมที่ Carry flag



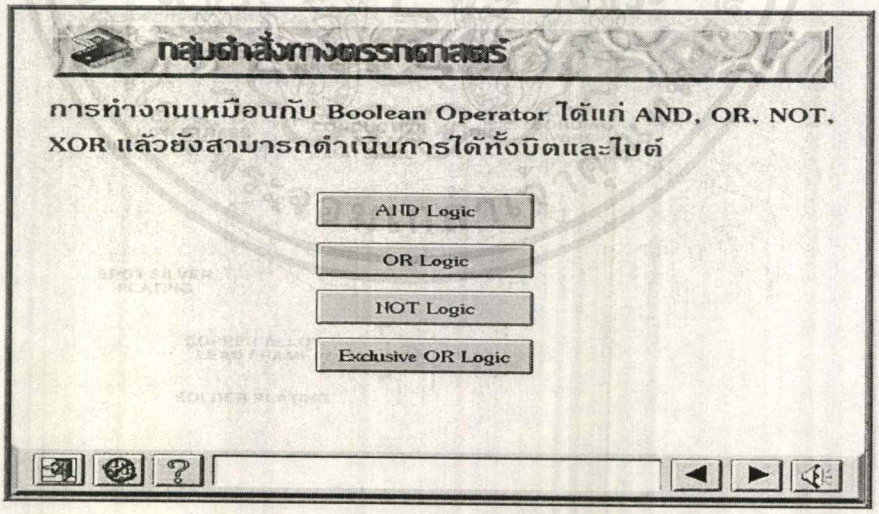
คำบรรยาย รูปแบบของการคูณและหาร MUL A, B คือการคูณระหว่าง Register A และ B โดยผลคูณที่ได้ 8 บิทล่างจะเก็บใน Acc และ 8 บิทบนเก็บที่ Register B

DIV A, B คือการหารระหว่าง Register A และ B โดยผลลัพธ์ที่ได้เก็บที่ Acc และเศษเก็บที่ Register B

คำบรรยาย รูปแบบของการเพิ่มและลดค่าของข้อมูล

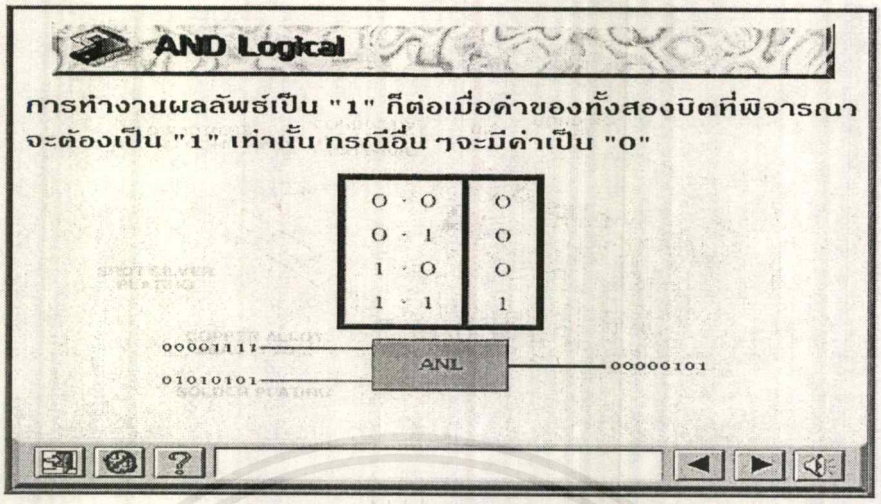
คำสั่ง INC DATA เป็นการเพิ่มค่าของข้อมูล DATA ที่กำหนดขึ้นทีละหนึ่ง

คำสั่ง DEC DATA เป็นการลดค่าของข้อมูล DATA ที่กำหนดลงทีละหนึ่ง

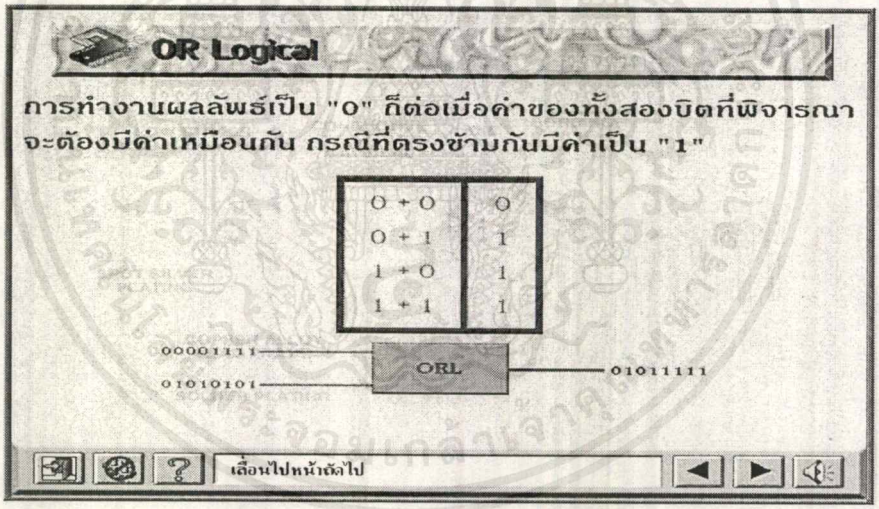


คำบรรยาย กลุ่มคำสั่งทางตรรกศาสตร์ การทำงานจะเหมือนกับ Boolean operation คือ AND, OR, NOT และ XOR และยังสามารถทำงานได้ทั้ง bit และ byte มี 4 คำสั่งใหญ่ คือ

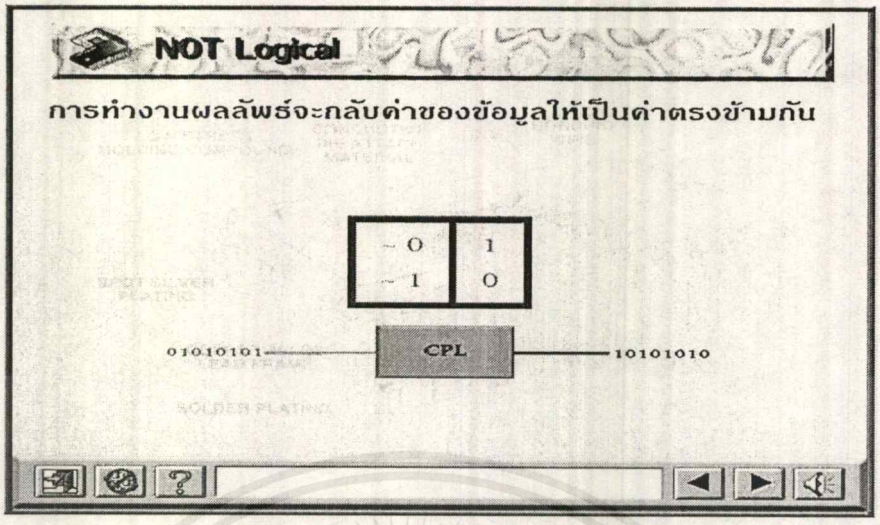
1.AND Logic 2.OR Logic 3.NOT Logic 4.OR Logic



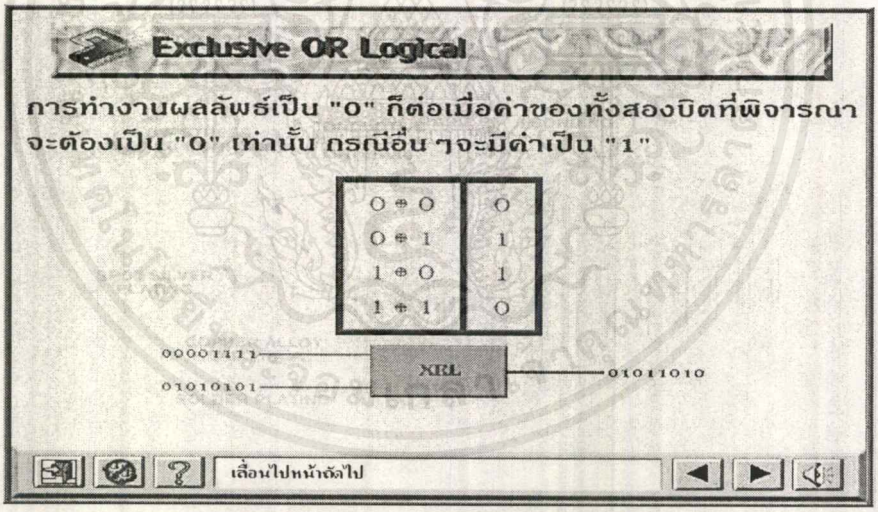
คำบรรยาย การทำงานของ AND logical การทำงานผลลัพธ์ “1” ก็ต่อเมื่อค่าของทั้งสองบิตจะต้องเป็น “1” เท่านั้น



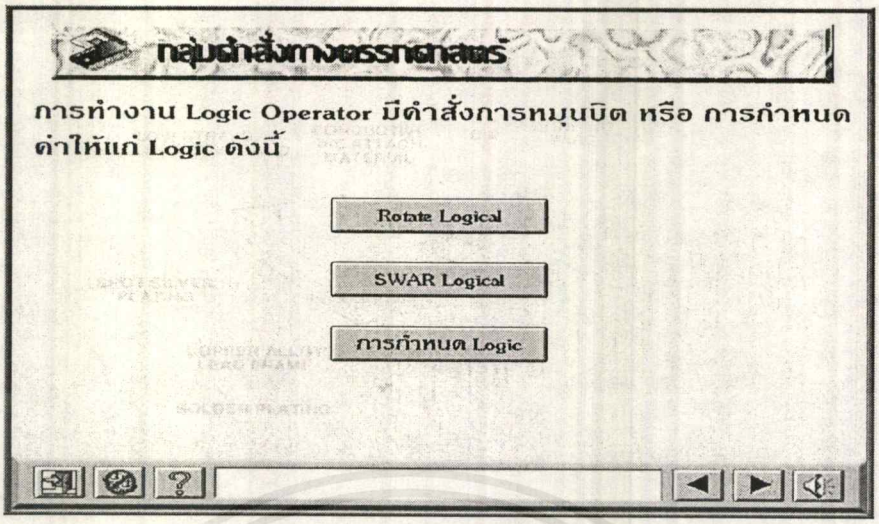
คำบรรยาย การทำงานของ OR Logical การทำงานของ OR Logical นั้น ผลลัพธ์จะเป็น 0 ก็ต่อเมื่อค่าของทั้งสองบิตที่พิจารณาจะต้องมีค่าเหมือนกัน กรณีที่ค่าของทั้งสองบิตมีค่าตรงกันข้าม ผลลัพธ์จะเป็น 1



คำบรรยาย การทำงานของ NOT Logical การทำงานนั้นผลลัพธ์จะกลับคาสของข้อมูลให้เป็นค่าตรงกันข้าม เช่น ถ้าข้อมูลเป็น 0 เมื่อทำการ NOT ผลลัพธ์จะเป็น 1 และในทางกลับกัน ถ้าข้อมูลเป็น 1 เมื่อทำการ NOT ผลลัพธ์จะเป็น 0

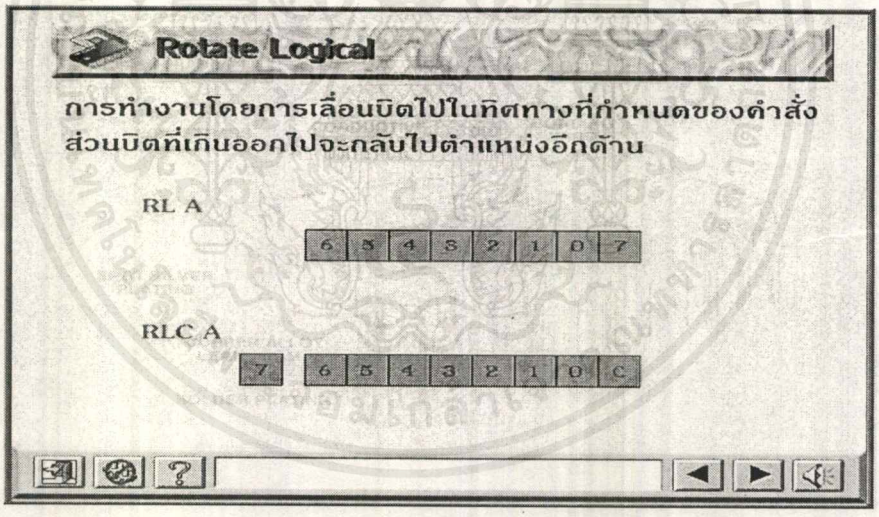


คำบรรยาย การทำงานของ OR Logical การทำงานผลลัพธ์จะเป็น 0 ก็ต่อเมื่อค่าของทั้งสองบิทที่พิจารณาจะต้องมีค่าเหมือนกัน กรณีที่ค่าของทั้งสองบิทมีค่าตรงกันข้าม ผลลัพธ์จะเป็น 1



คำบรรยาย กลุ่มคำสั่งทางตรรกศาสตร์ การทำงานของ Logical Operator มีคำสั่งการหมุนบิท หรือการกำหนดค่าให้แก่ Logical ดังนี้คือ

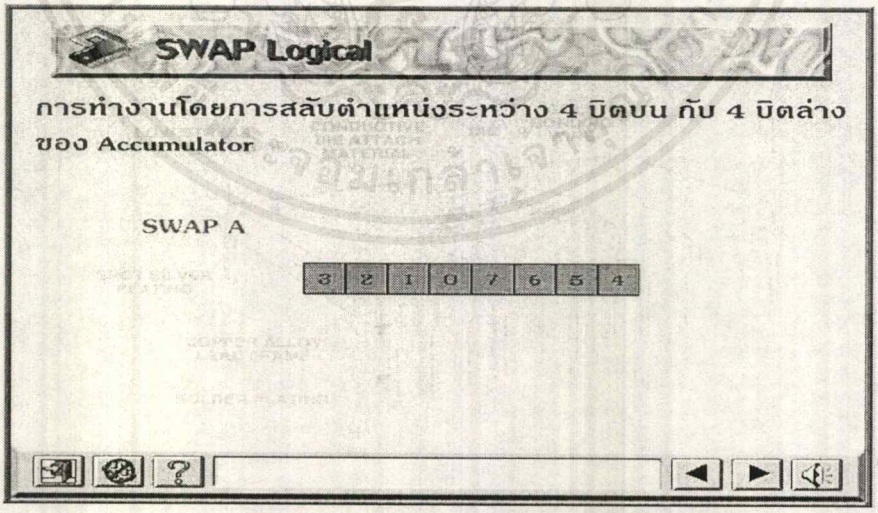
1. Rotate Logical 2. Swap Logical 3. การกำหนด Logical



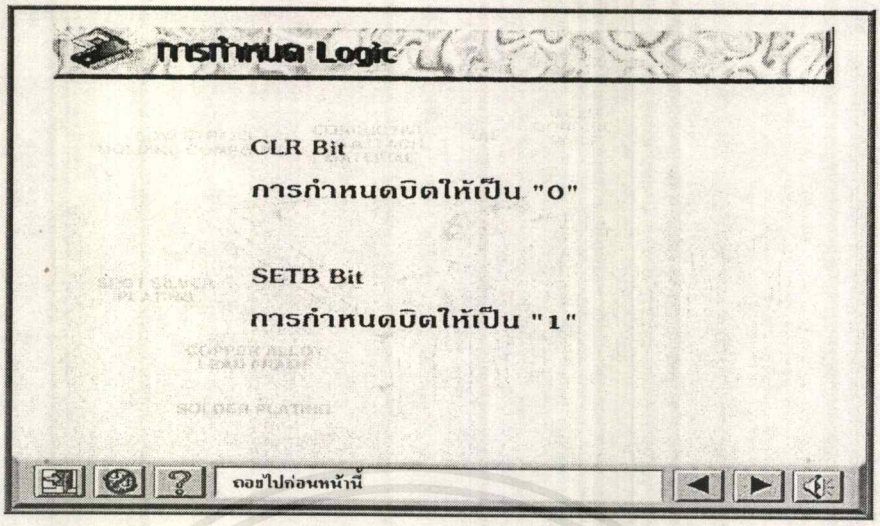
คำบรรยาย Rotate Logical หรือคำสั่งหมุนบิท เป็นคำสั่งที่ดำเนินการเฉพาะกับบิท ภายใน Register A เท่านั้น คือการหมุนบิทนอกจากจะจำแนกตามทิศทางการเลื่อนบิทไปทางซ้ายหรือขวา และยังสามารถมีการนำ Fact ทดเข้าร่วมในการหมุนบิทด้วย โดยหากเป็นการหมุนบิทแบบปกติ บิทข้อมูลภายในทั้ง 8 บิท ก็จะเลื่อนไปอีกตำแหน่งตามทิศทางที่ได้ระบุไว้อยู่ภายใน Register A เท่านั้น แต่หากว่านำ Fact ทดเข้ามาร่วมด้วยแล้ว จะมีการนำบิท 9 เข้ามาเพิ่มในการเลื่อนบิทด้วย ดังตัวอย่างคำสั่ง RL A คือให้ทำการเลื่อนบิทข้อมูลภายใน Register A ไปทางด้านซ้าย 1 ตำแหน่ง โดยบิท 7 ให้เลื่อนมาเก็บยังตำแหน่งบิท 0 ส่วนคำสั่ง RLC A คือให้ทำการเลื่อนบิทข้อมูลไปทางซ้าย 1

ตำแหน่ง โดยบิท 7 ให้เลื่อนมาเก็บยังตำแหน่ง Fact ทด และค่าในตำแหน่ง Fact ทดเดิมให้นำมาเก็บยังตำแหน่งบิท 0

คำบรรยาย คำสั่ง RR A คือให้ทำการเลื่อนบิทข้อมูลภายใน Register A ไปยังด้านขวา 1 ตำแหน่งโดยบิท 0 ให้เลื่อนมาเก็บยังตำแหน่งบิท 7 อีกคำสั่งคือ RRC A คือให้ทำการเลื่อนบิทข้อมูลภายใน Register A ไปยังด้านขวา 1 ตำแหน่งโดยบิท 0 ให้เลื่อนมาเก็บยังตำแหน่งของ Fact ทด และค่าในตำแหน่ง Fact ทดเดิมให้เลื่อนไปเก็บยังบิท 7



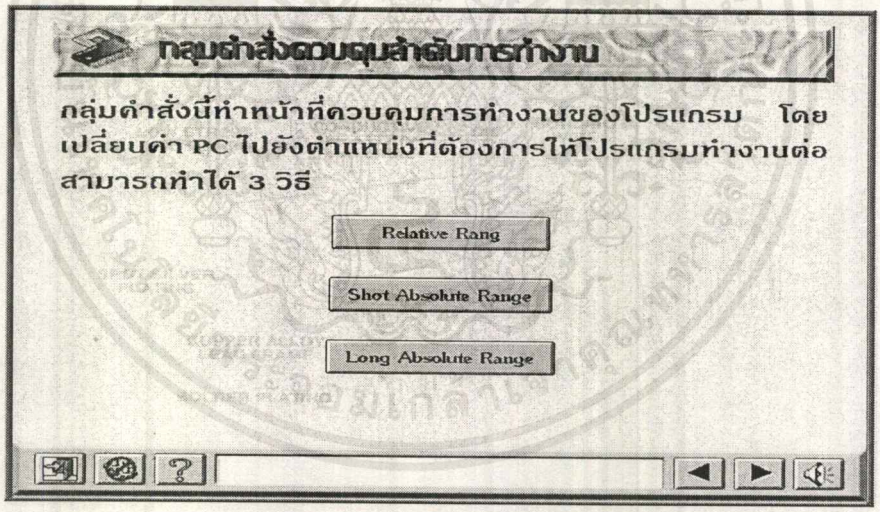
คำบรรยาย SWAP Logical หรือคำสั่งสลับค่าข้อมูลครั้งละ 4 บิท คำสั่งนี้จะสลับค่าภายในที่ใช้เฉพาะ Register A โดยจะย้ายตำแหน่งข้อมูล 4 บิทบน ไปไว้ในตำแหน่ง 4 บิทล่าง โดยข้อมูลใน 4 บิทล่างก็จะย้ายไปอยู่ตำแหน่ง 4 บิทบนเช่นกัน ลักษณะเช่นนี้เรียกว่าการสลับค่า (SWAP)



คำบรรยาย การกำหนด Logic มี 2 คำสั่งคือ

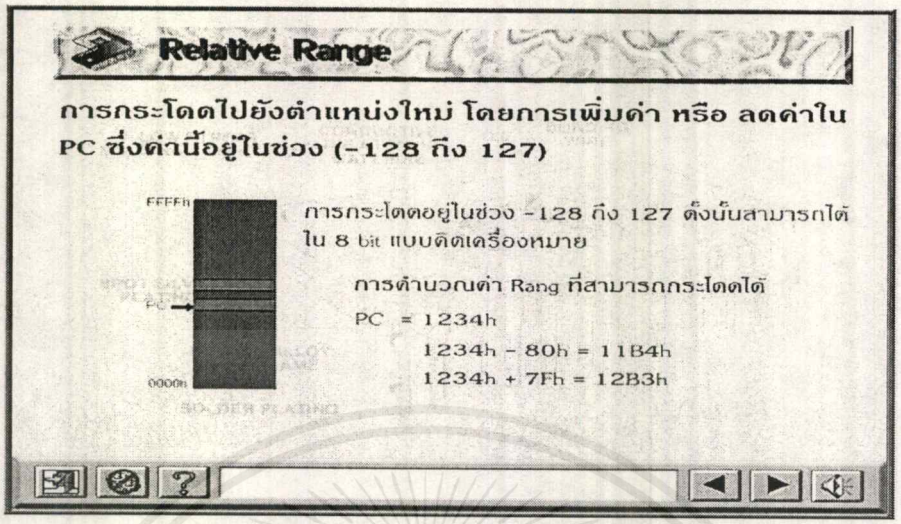
1.คำสั่ง Clear จะเป็นการกำหนดค่า bit ให้เป็น 0

2.คำสั่ง Set bit จะเป็นการกำหนดค่า bit ให้เป็น 1



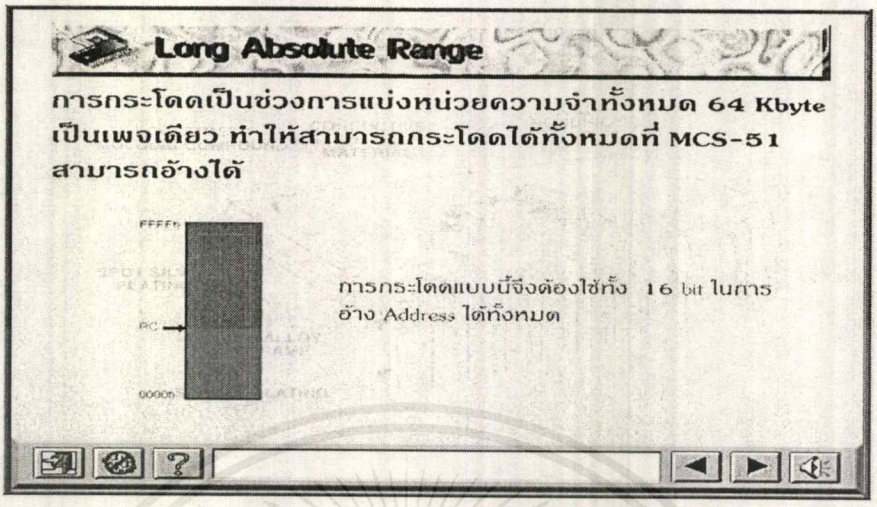
คำบรรยาย กลุ่มคำสั่งควบคุมลำดับการทำงาน กลุ่มคำสั่งนี้ทำหน้าที่ควบคุมการทำงานโดยเปลี่ยนค่า PC ไปยังตำแหน่งที่ต้องการเพื่อให้โปรแกรมไปทำงานต่อ สามารถทำได้ 3 วิธีคือ

1.Relative Absolute Range 2.Long Absolute Range 3.Short Absolute Range

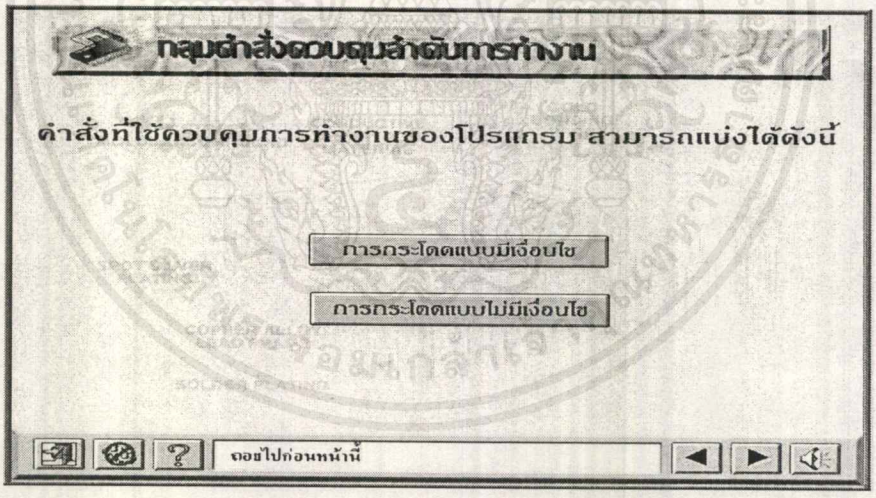


คำบรรยาย Relative Range การกระโดดไปยังตำแหน่งใหม่โดยการลดค่าหรือเพิ่มค่าใน PC ซึ่งค่านี้อยู่ในช่วง -128 ถึง 127 ดังนั้นสามารถทำได้ใน 8 bit แบบคิดเครื่องหมาย

คำบรรยาย Short Absolute Range เป็นการกระโดดในช่วงของการแบ่งหน่วยความจำเป็น page โดยจะแบ่งเป็น page ละ 2 Kbyte ถ้ามีหน่วยความจำทั้งหมด 64 Kbyte จะได้ทั้งหมด 32 page ส่านการคำนวณว่า Range ที่สามารถกระโดดได้ จะเป็นดังตัวอย่าง

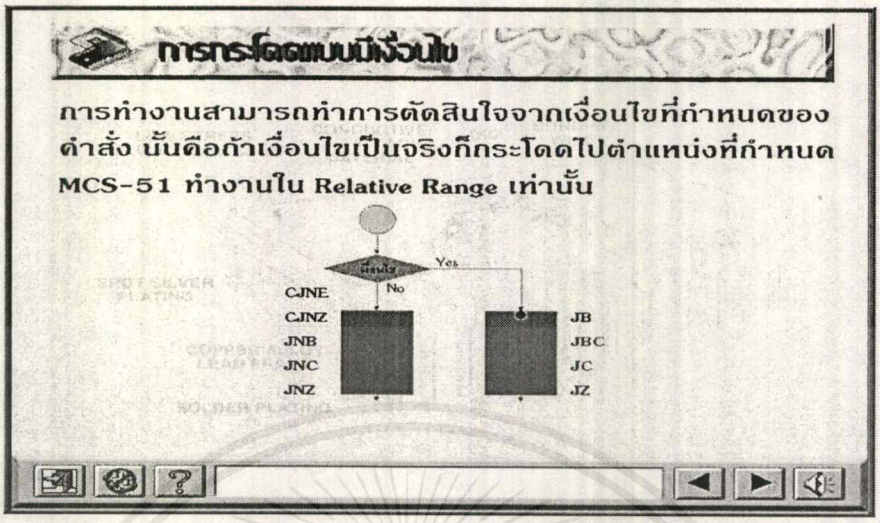


คำบรรยาย Long Absolute Range เป็นการกระโดในช่วงของการแบ่งหน่วยความจำทั้งหมด 64 Kbyte เป็น page เดียว ทำให้สามารถกระโดดได้ทั้งหมดที่ MCS-51 สามารถอ้างได้ การกระโดดแบบนี้จึงต้องใช้ทั้ง 16 บิท โดยการอ้าง Address ให้ได้ทั้งหมด

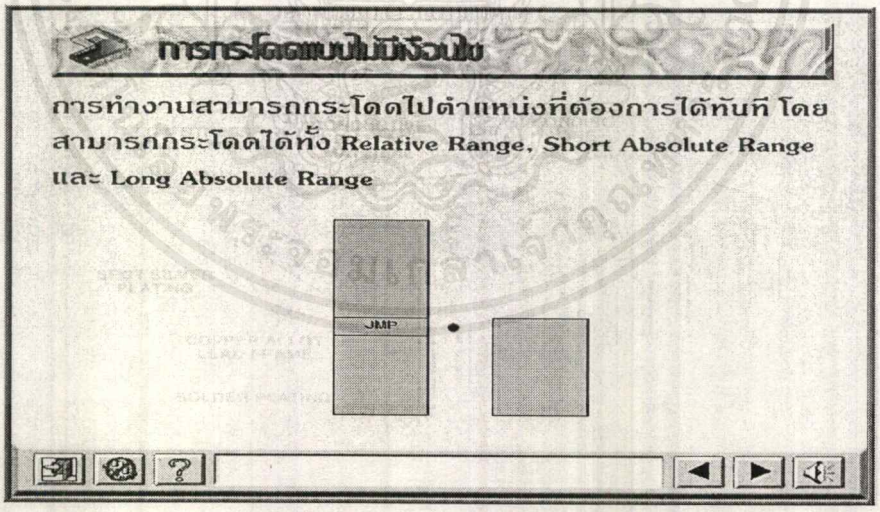


คำบรรยาย กลุ่มคำสั่งควบคุมลำดับการทำงาน คำสั่งที่ใช้ควบคุมลำดับการทำงานของโปรแกรมสามารถแบ่งได้ 2 แบบ คือ

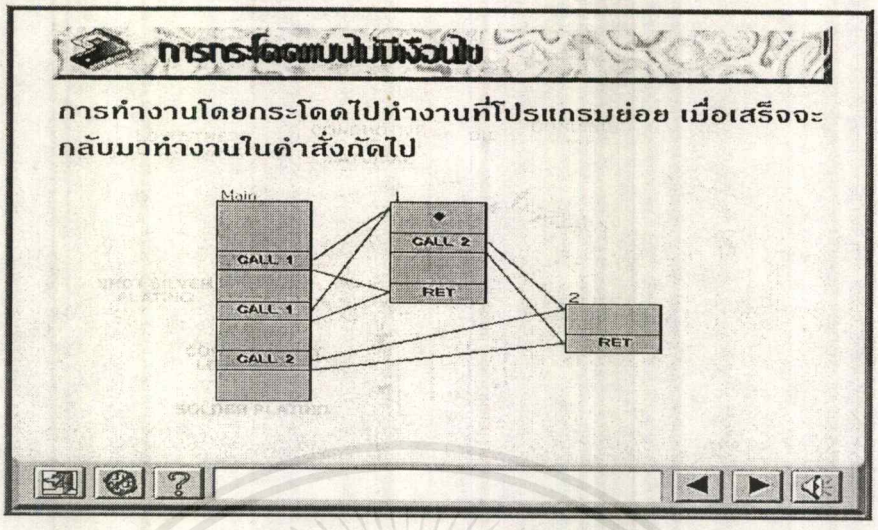
1. การกระโดดแบบมีเงื่อนไข 2. การกระโดดแบบไม่มีเงื่อนไข



คำบรรยาย การกระโดดแบบมีเงื่อนไข การทำงานสามารถทำงานจากการตัดสินใจจากเงื่อนไขที่กำหนดของคำสั่งนั่นคือ ถ้าเงื่อนไขเป็นจริง ก็จะกระโดดไปยังตำแหน่งที่กำหนด MCS-51 ทำงานใน Relative Rang เท่านั้นดังเช่นในตัวอย่าง ถ้าเงื่อนไขเป็น Yes ก็จะกระทำอย่างหนึ่ง ถ้าเงื่อนไขเป็น No ก็จะทำอีกการกระทำหนึ่ง

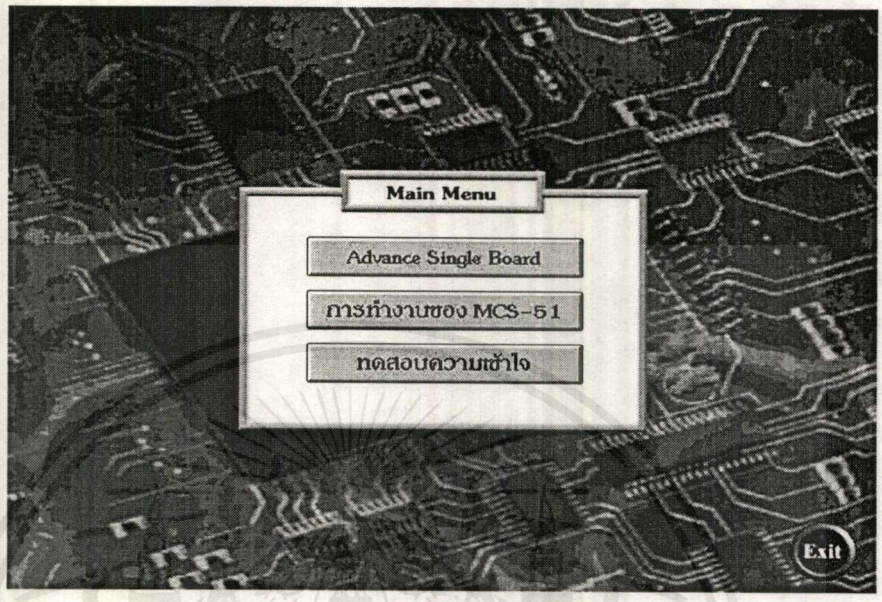


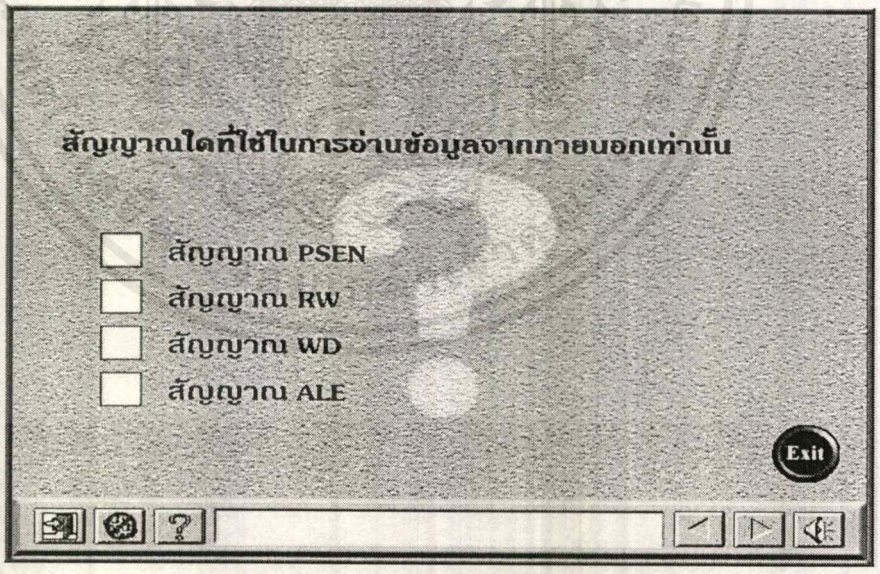
คำบรรยาย การกระโดดแบบไม่มีเงื่อนไข การทำงานสามารถกระโดดไปตำแหน่งที่ต้องการได้ทันที โดยสามารถกระโดดได้ทั้ง Relative Range, Short Absolute Range และ Long Absolute Range

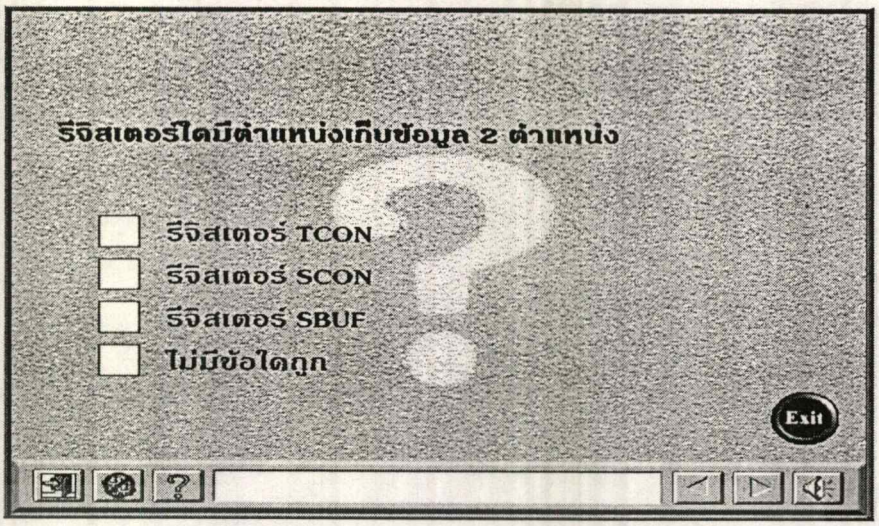


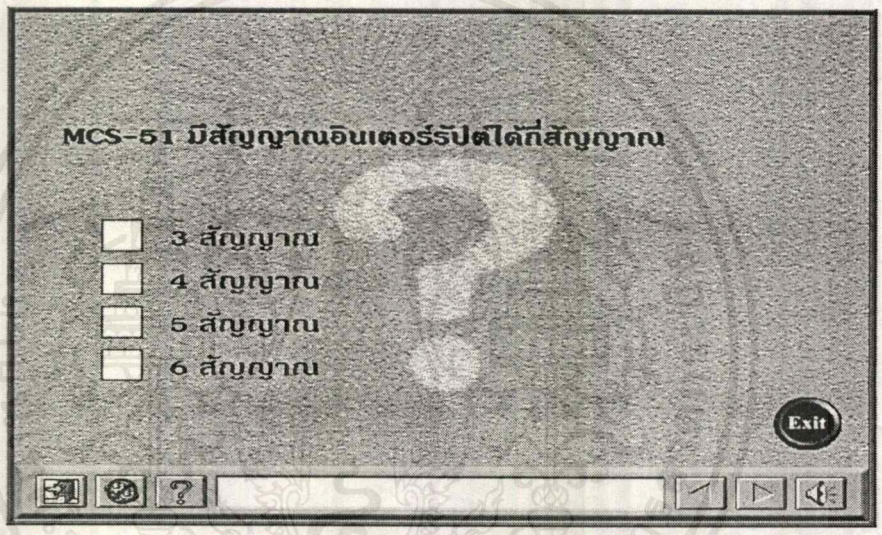
คำบรรยาย การกระโดดแบบไม่มีเงื่อนไขอีกแบบคือ การทำงานโดยกระโดดไปทำงานที่โปรแกรมย่อย เมื่อเสร็จจะกลับมาทำงานในคำสั่งถัดไป

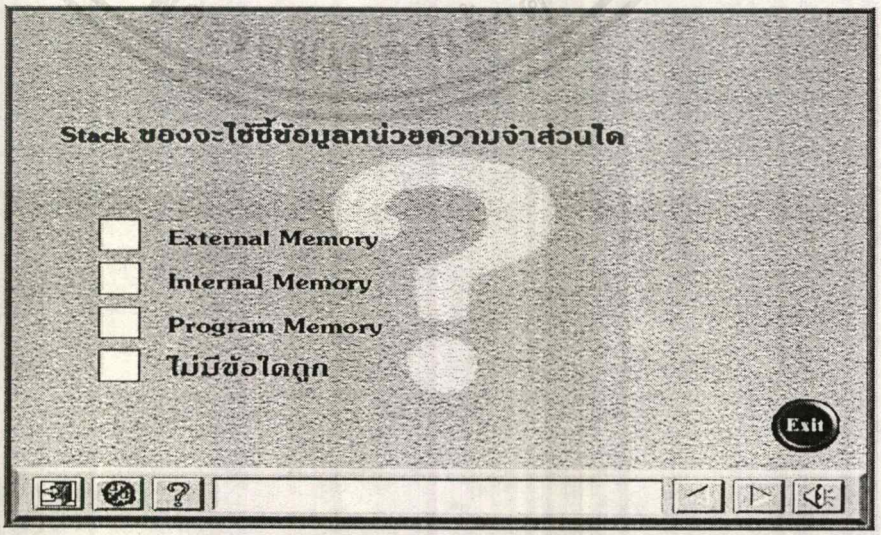
แบบทดสอบความเข้าใจ

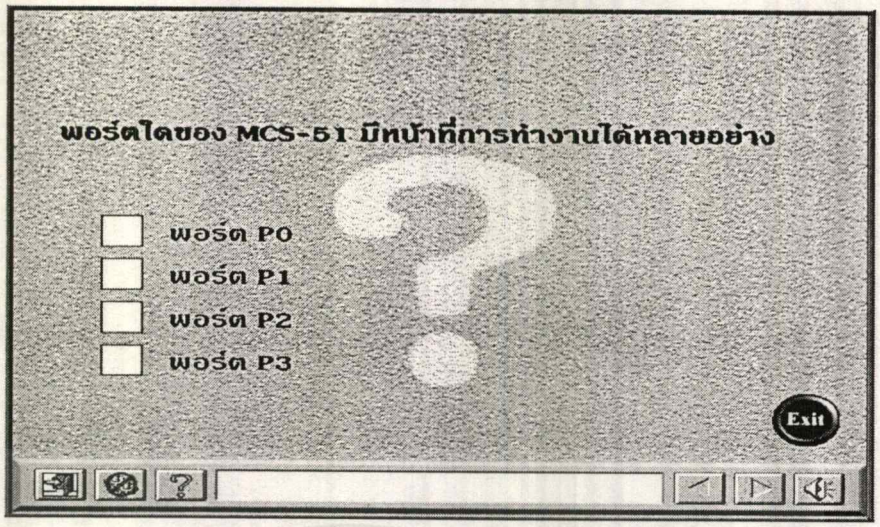


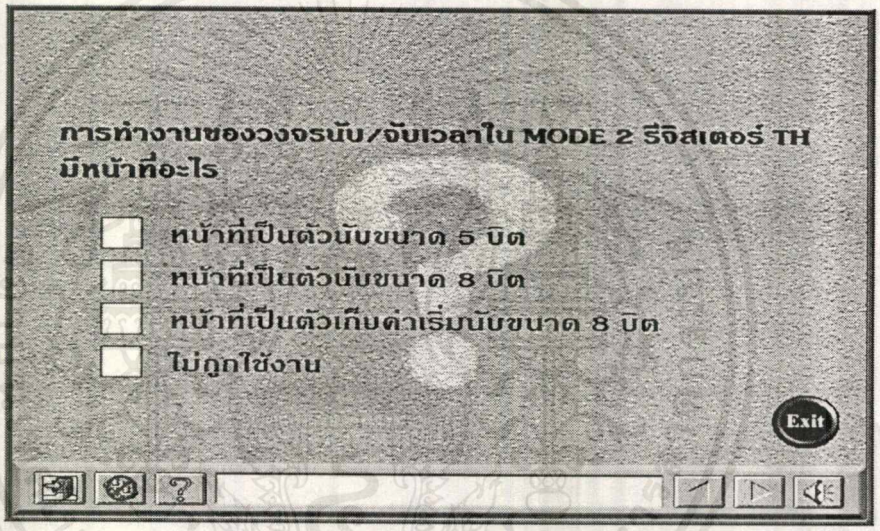


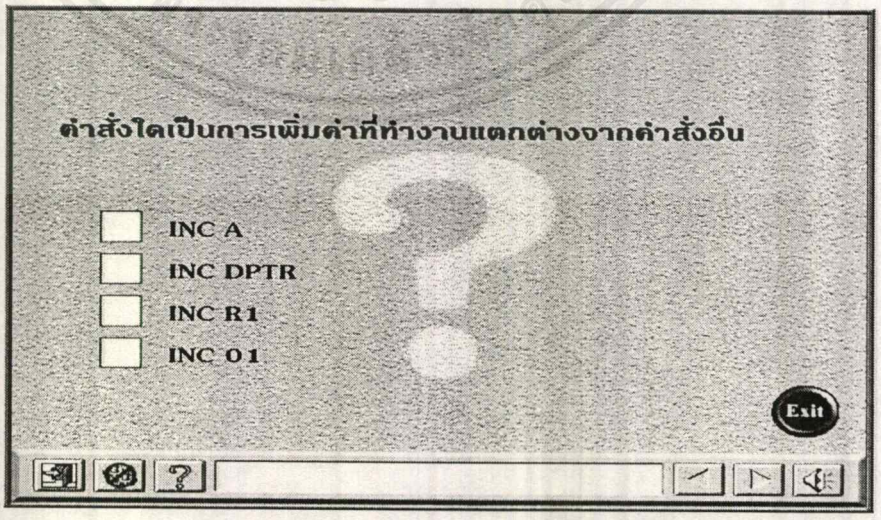


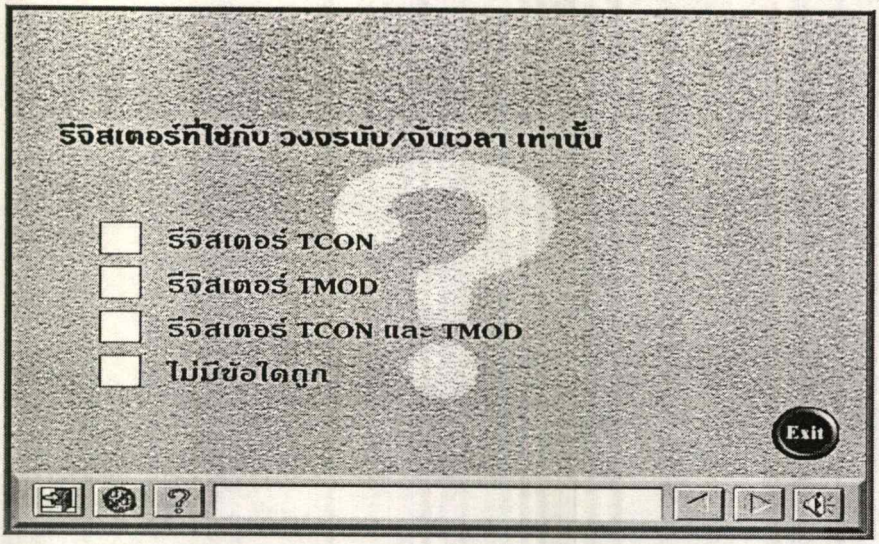


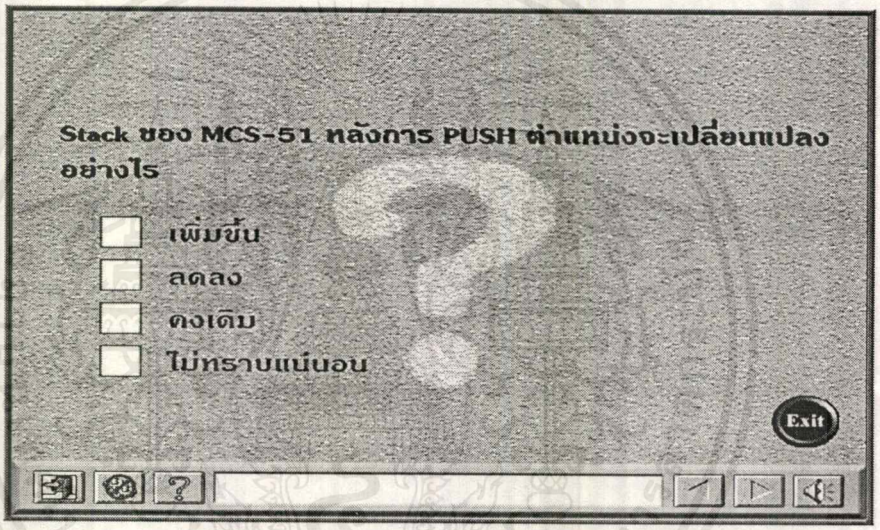


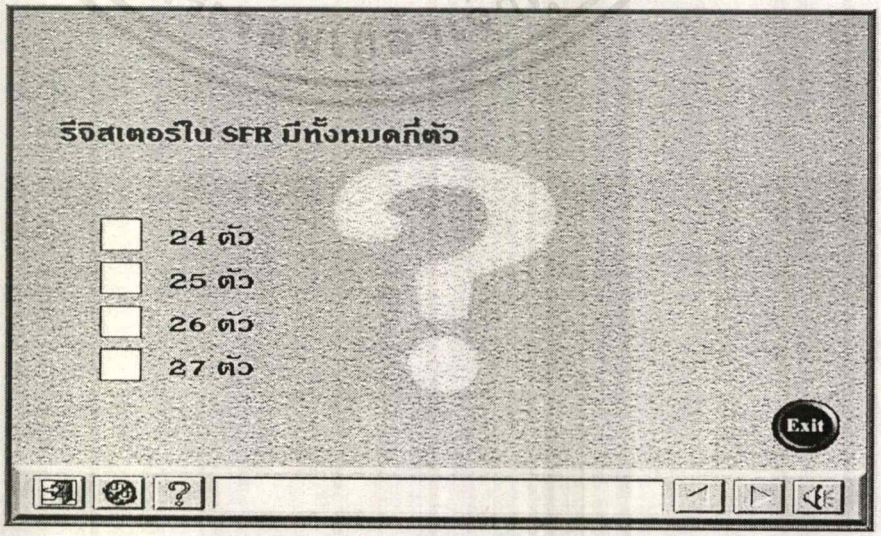


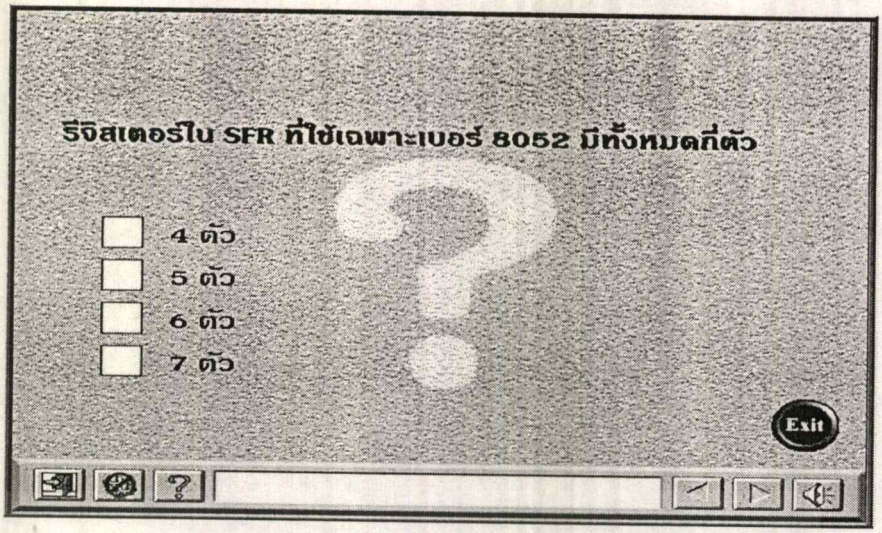


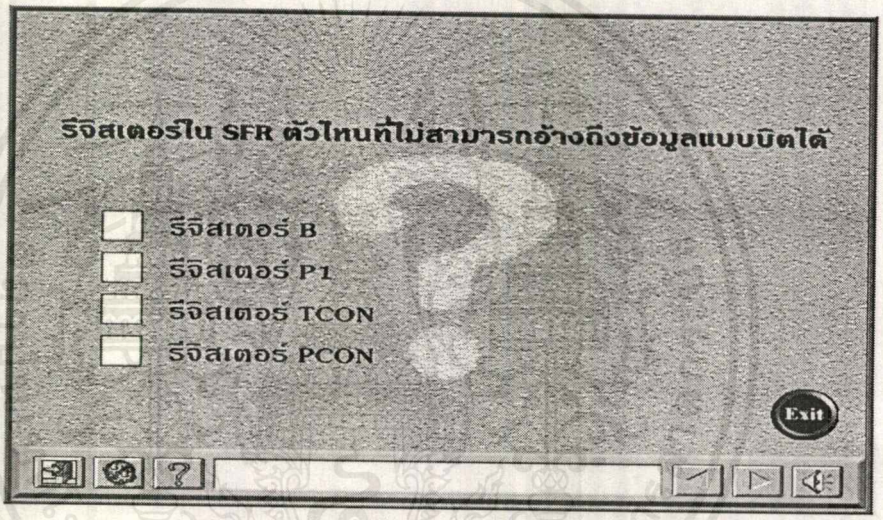


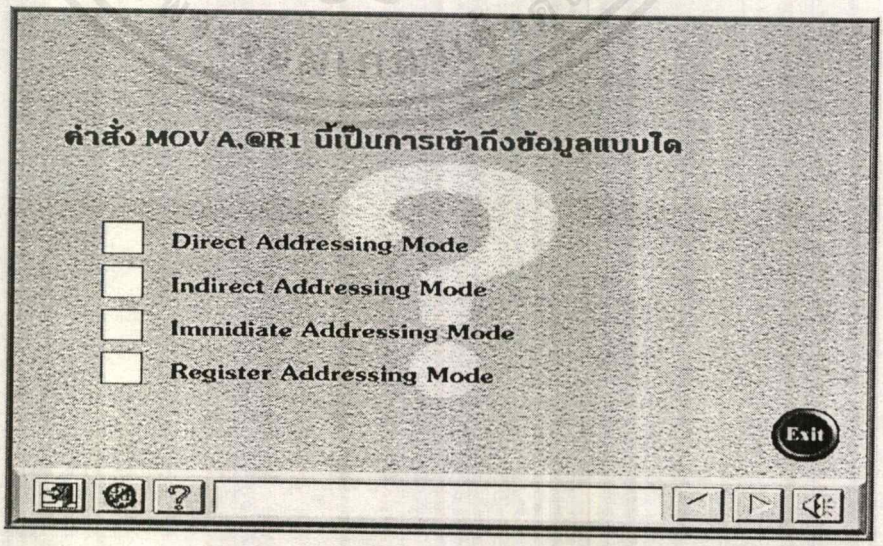


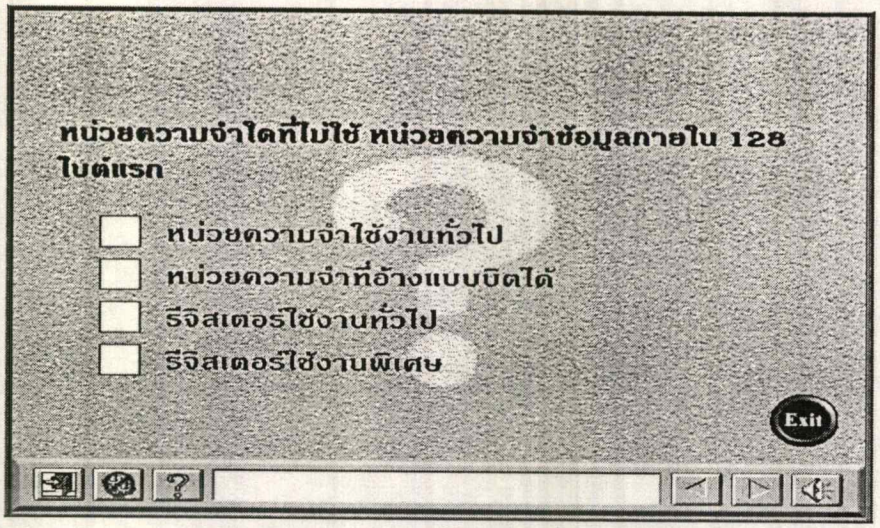


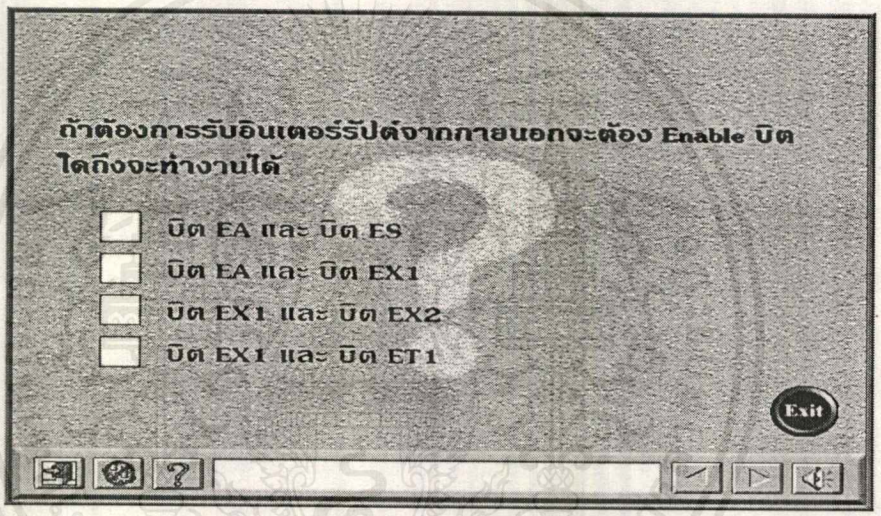


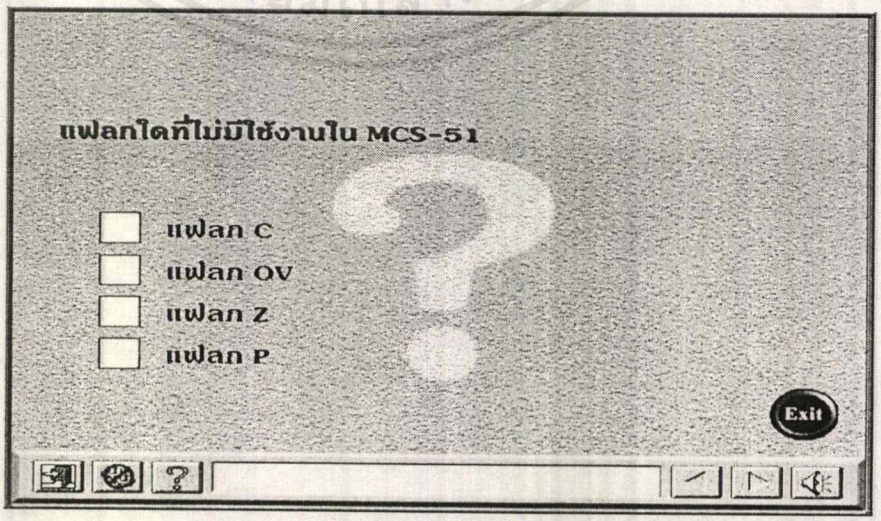


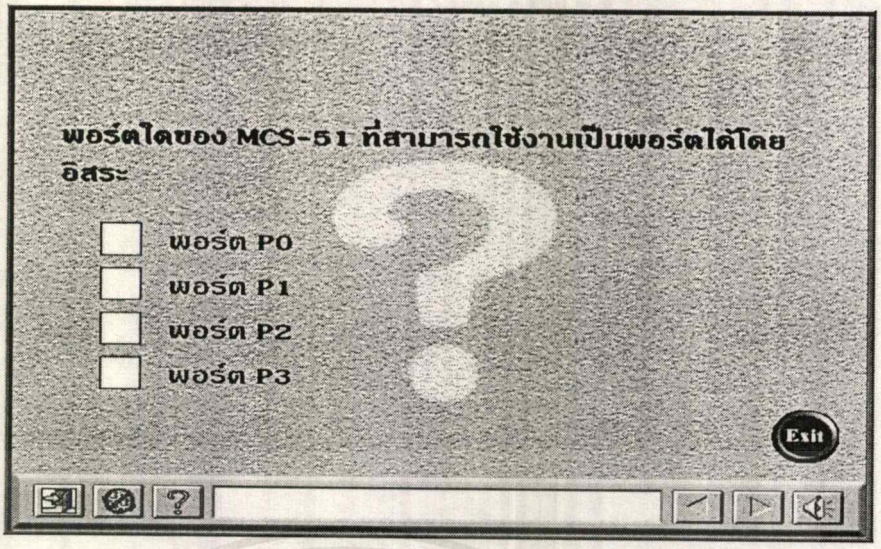


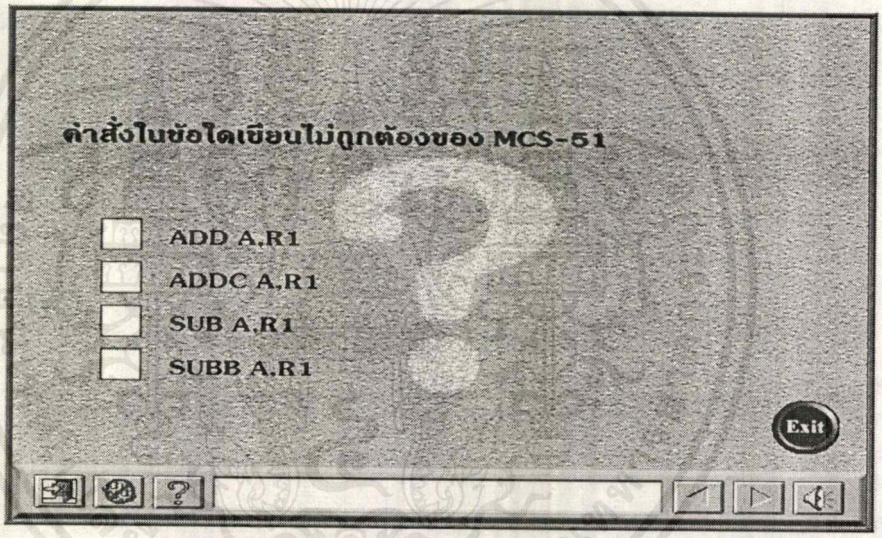


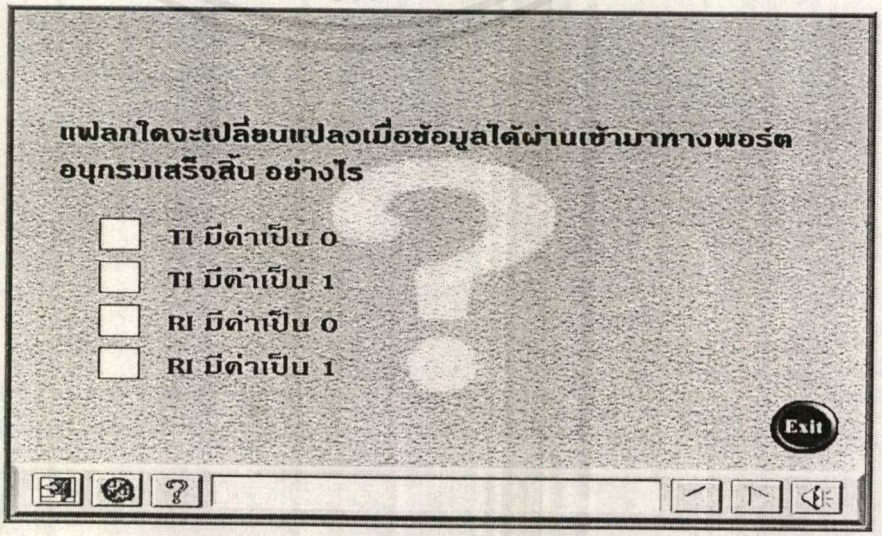


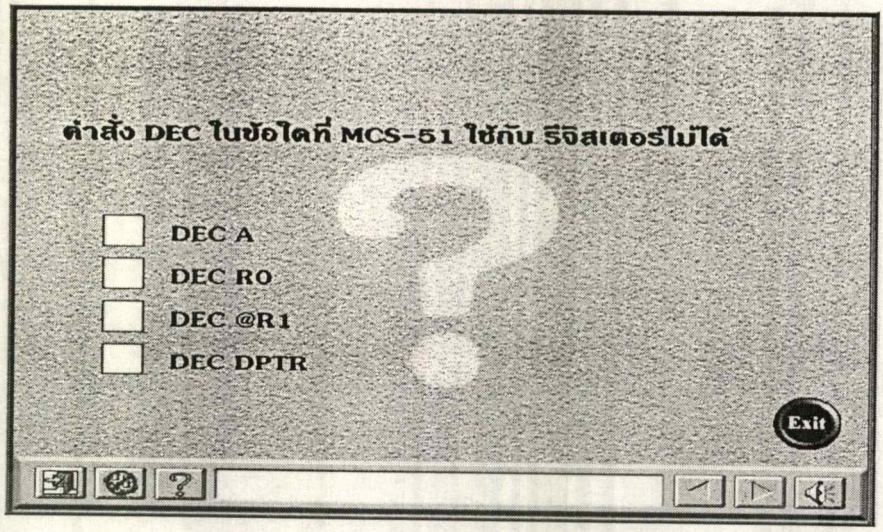


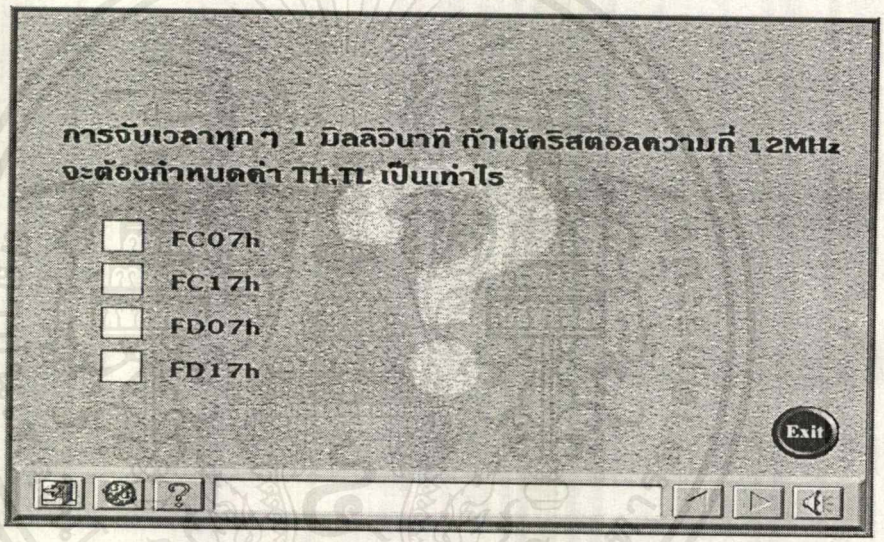


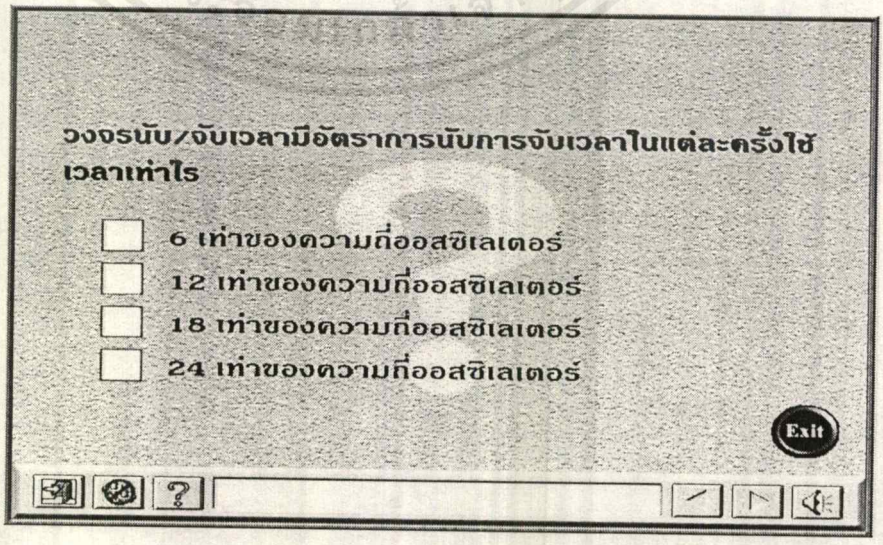


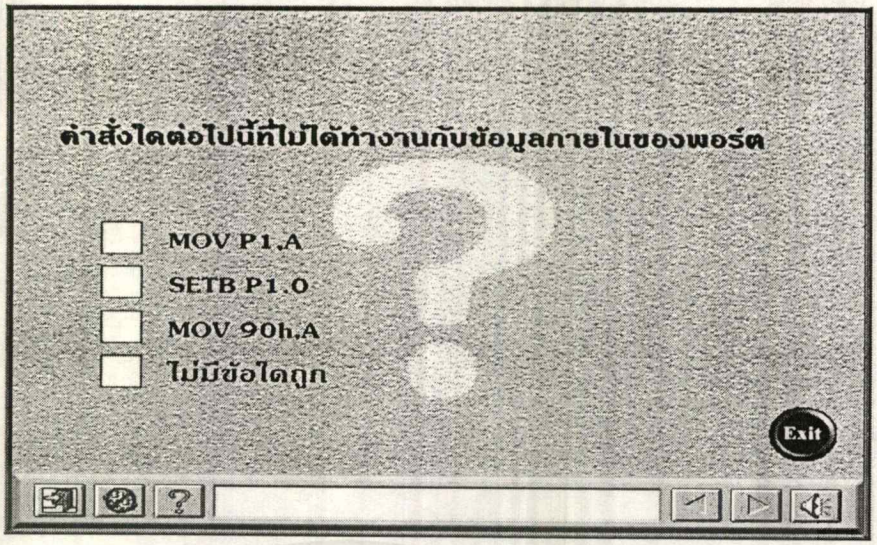


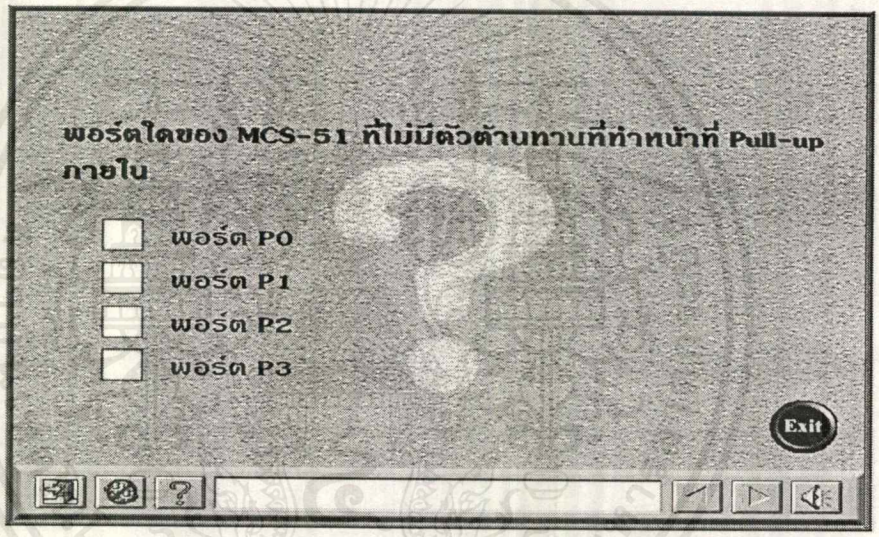


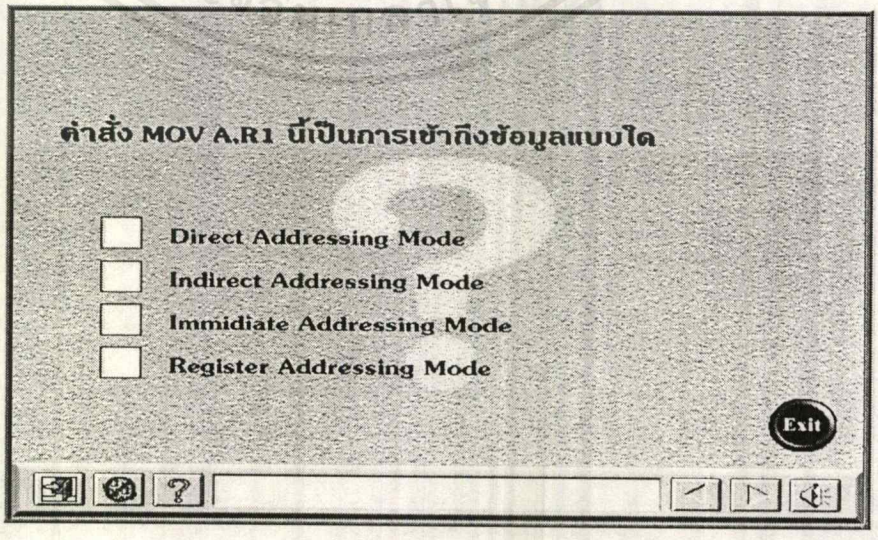


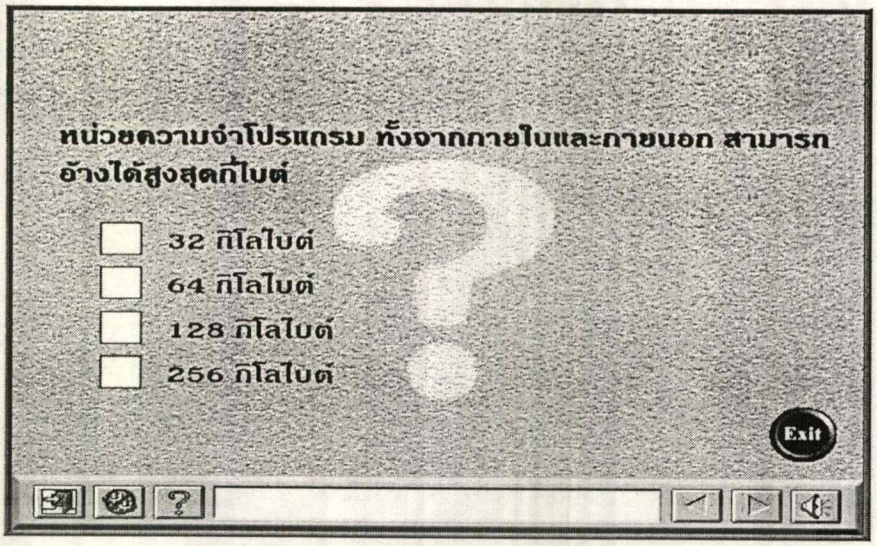


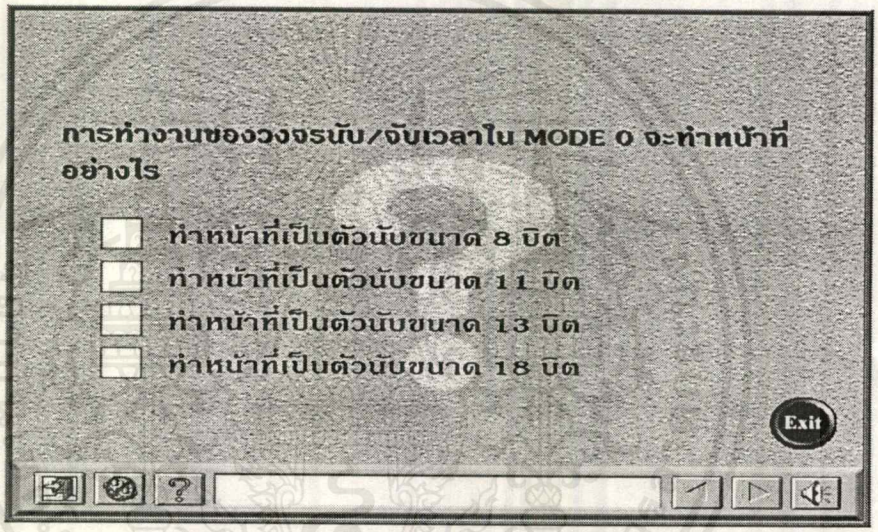


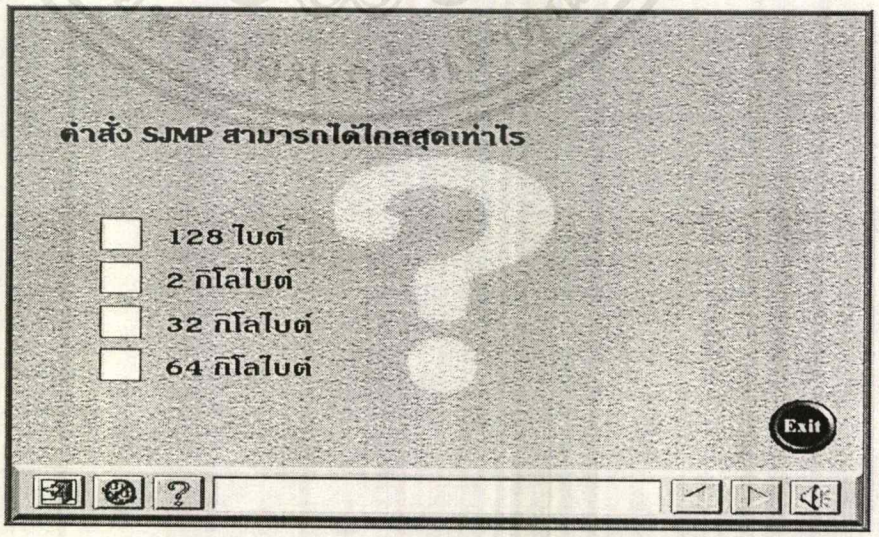


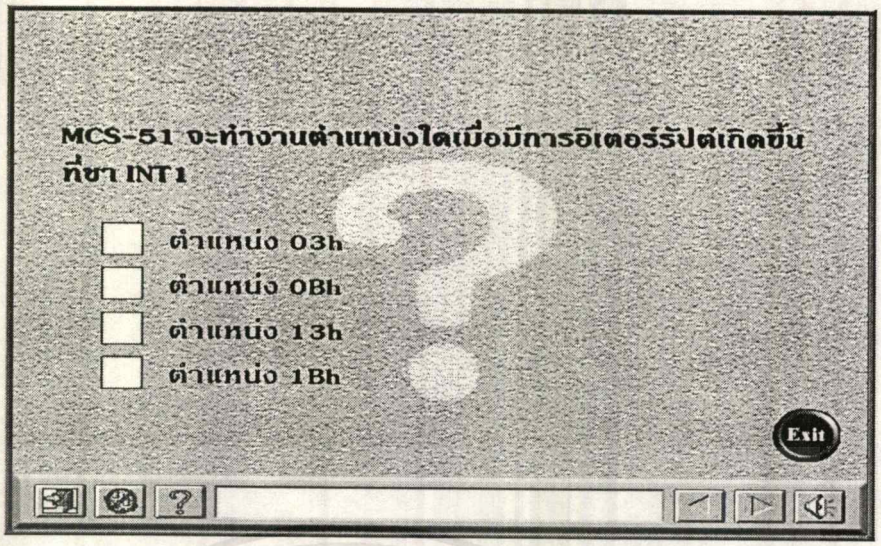


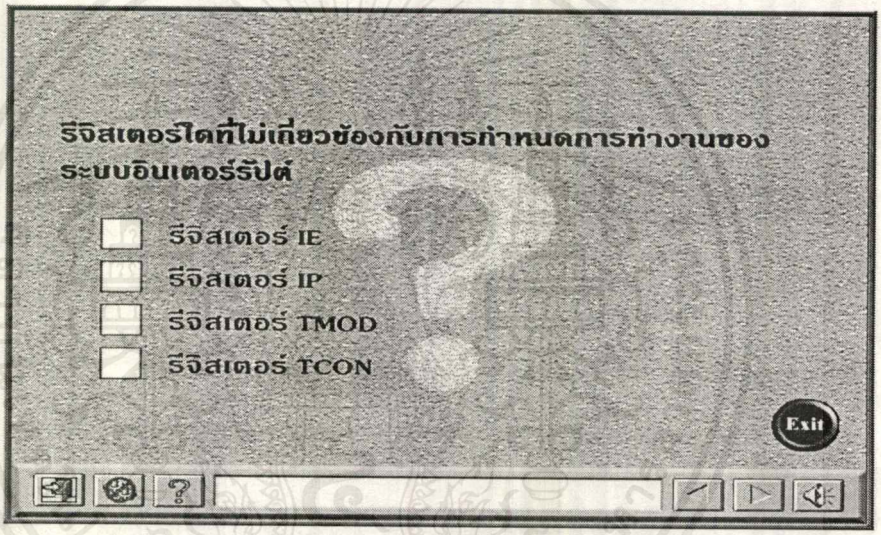


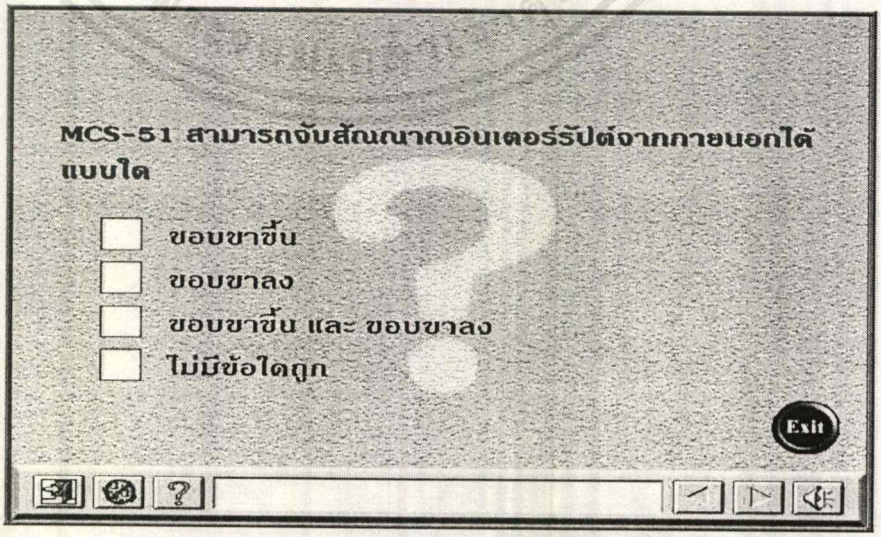


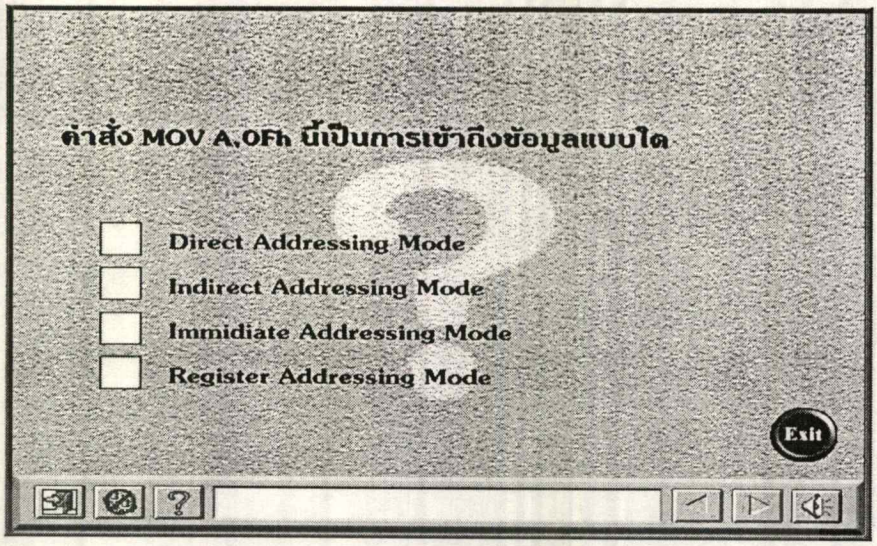


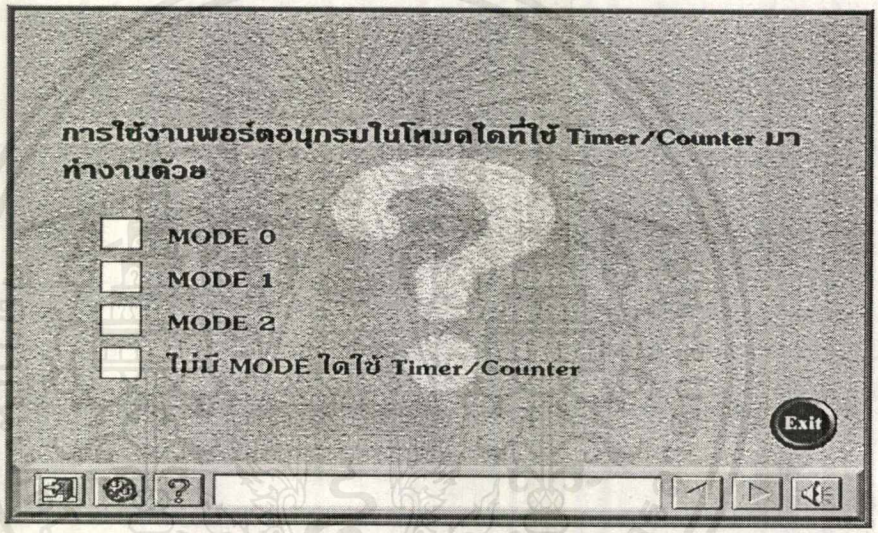


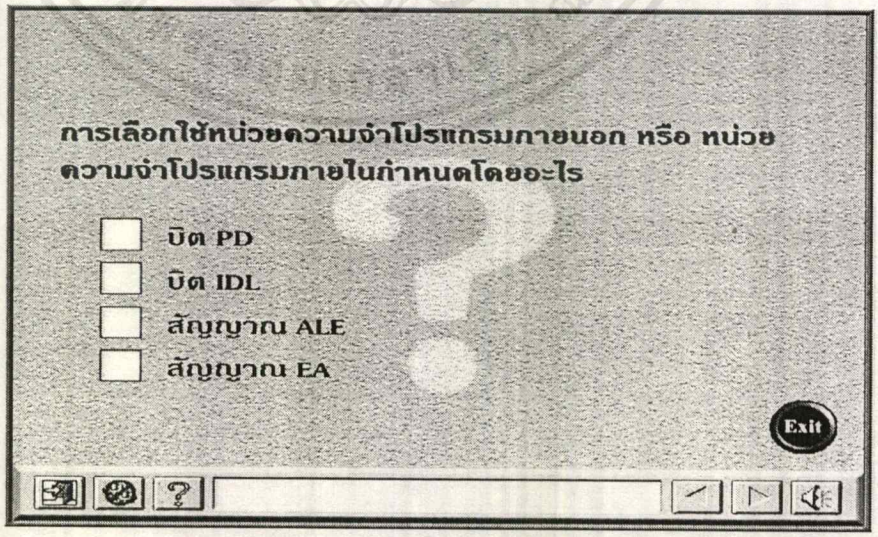


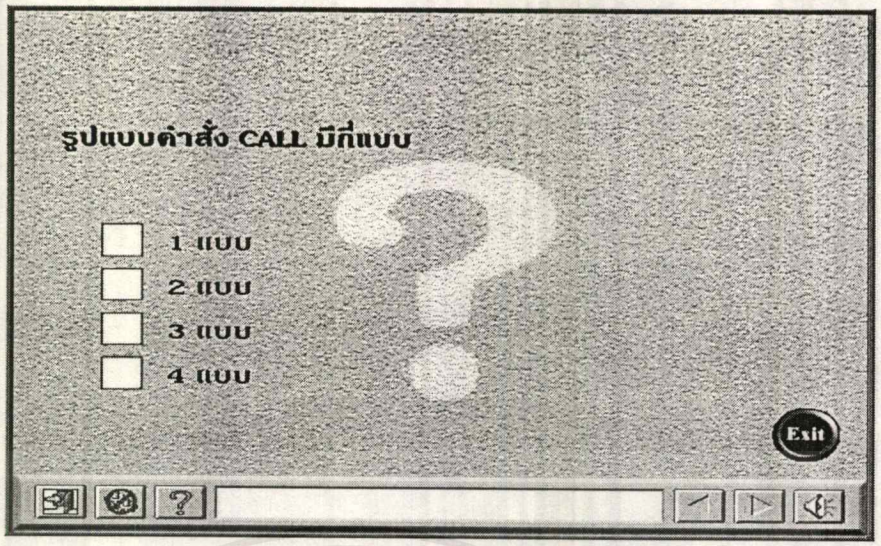


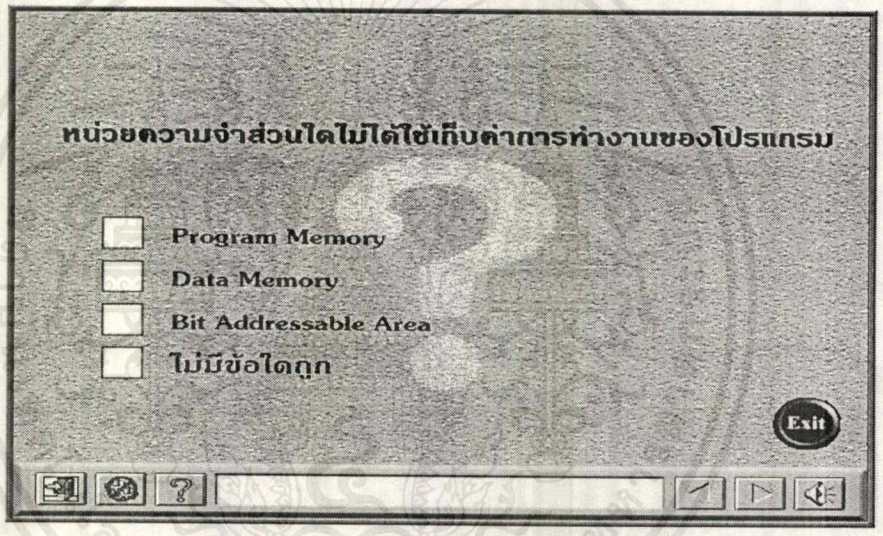


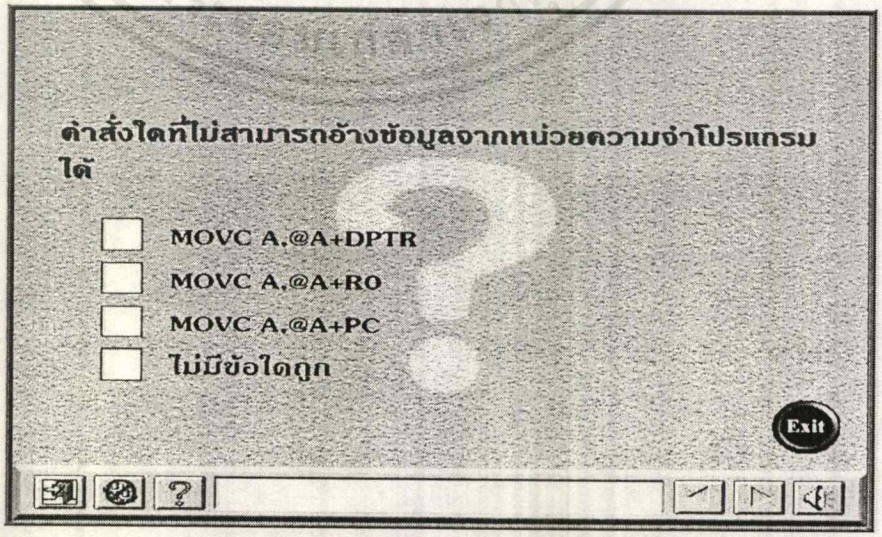


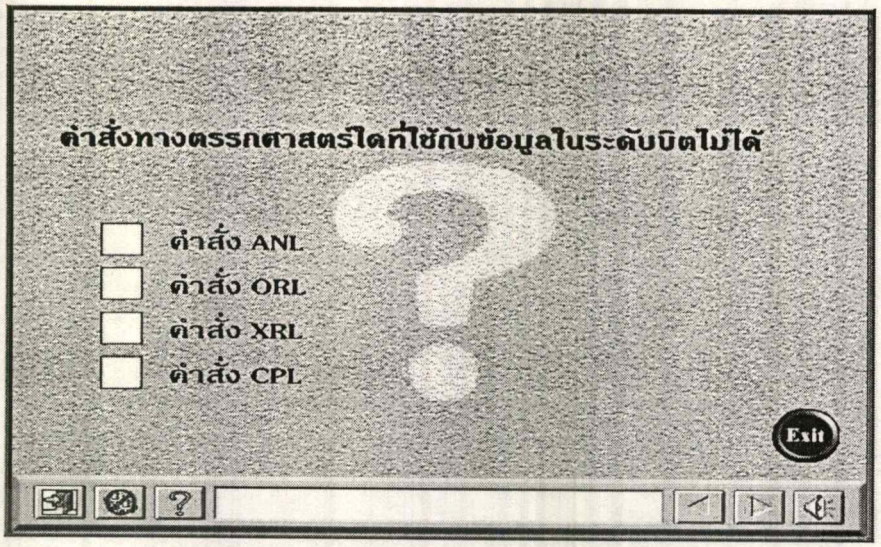


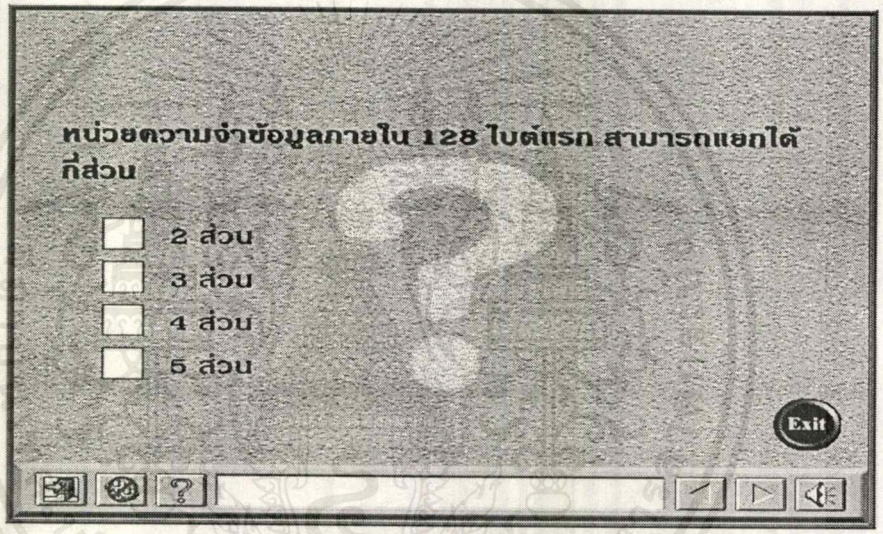


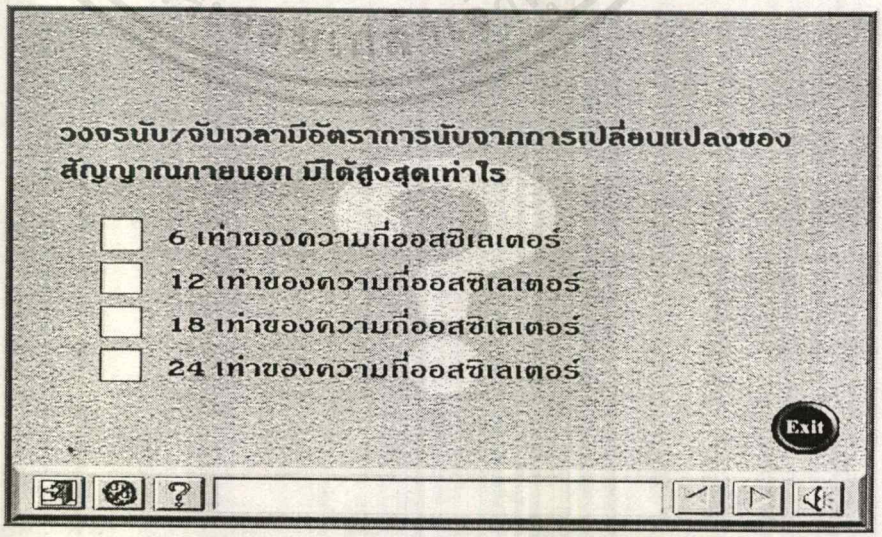


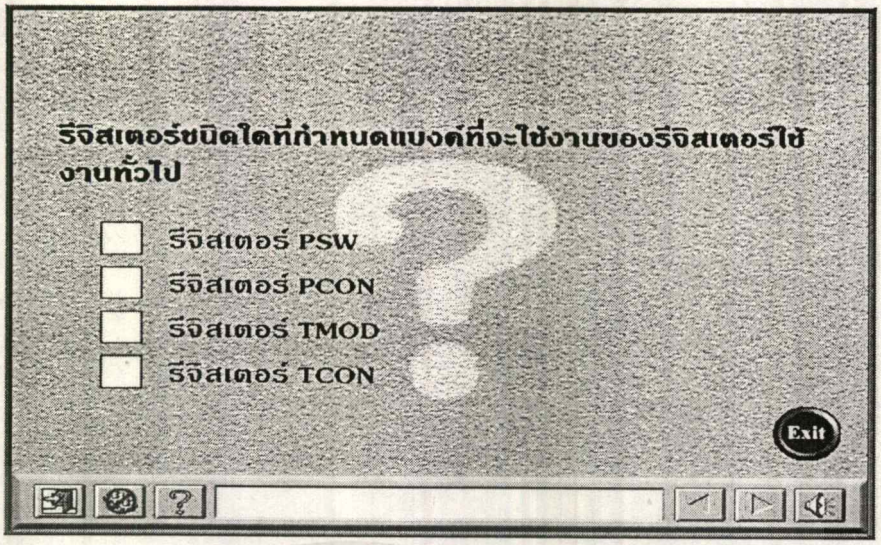


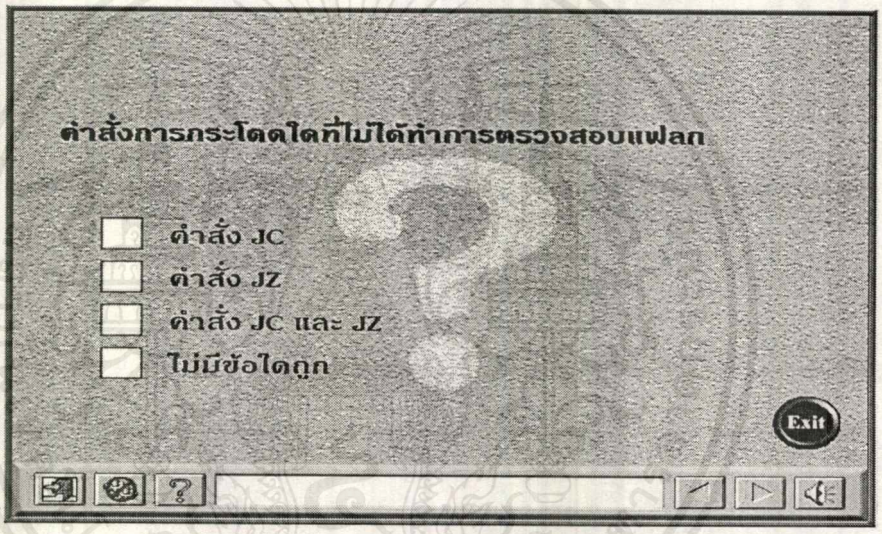


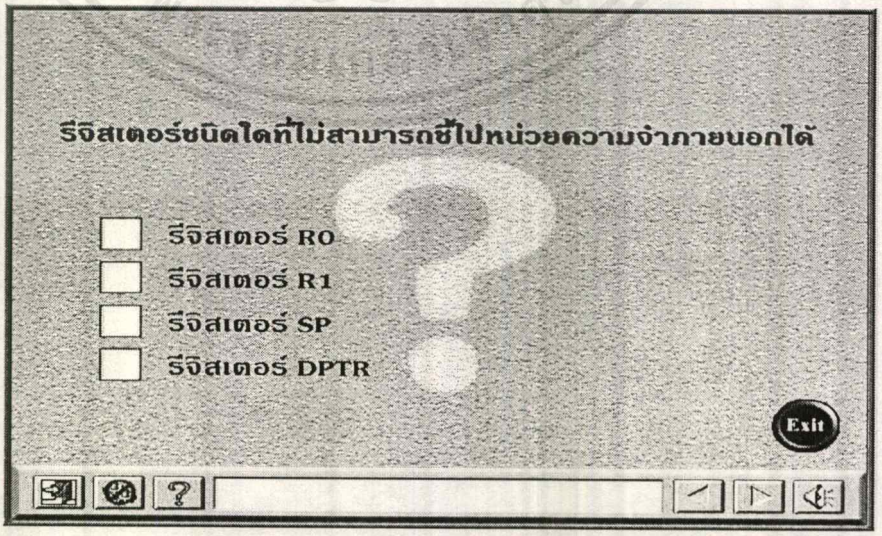


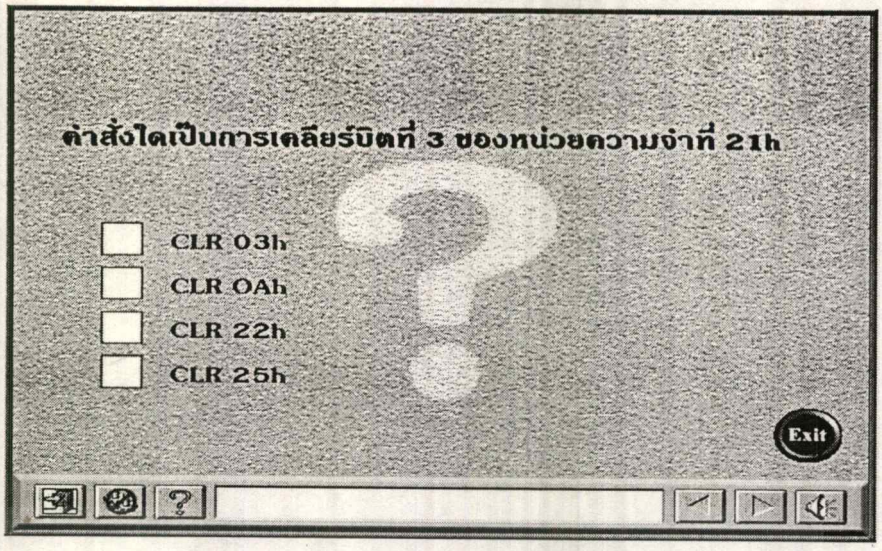


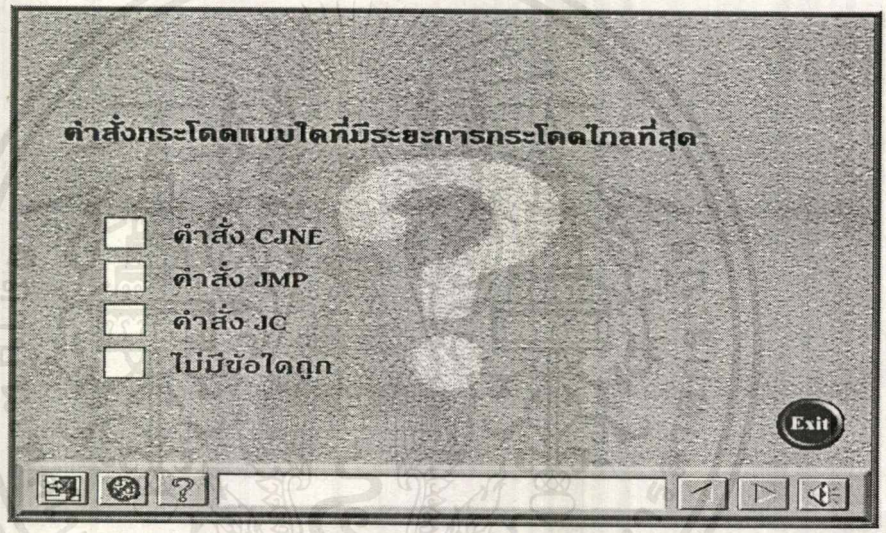


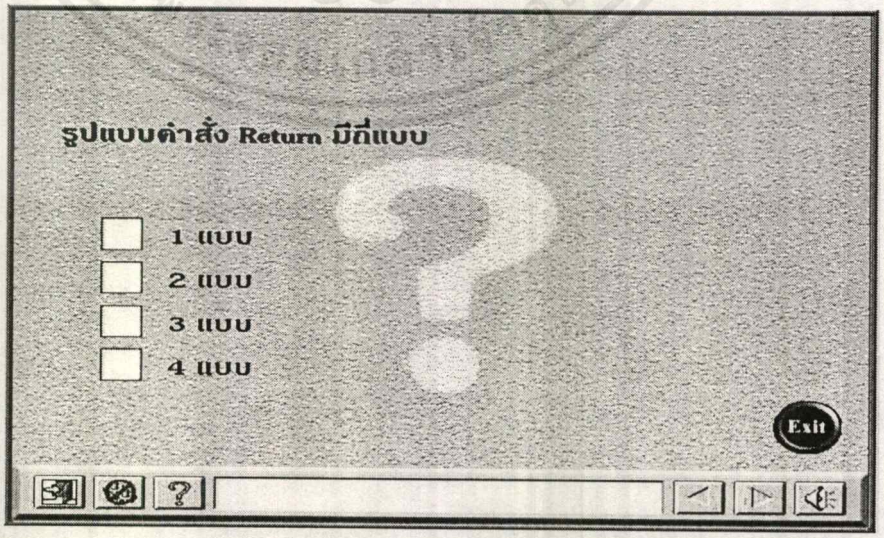


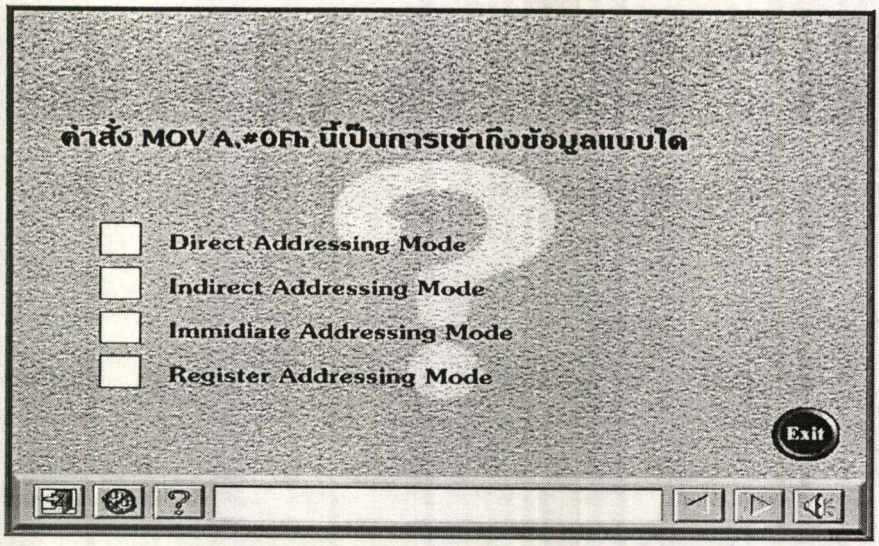


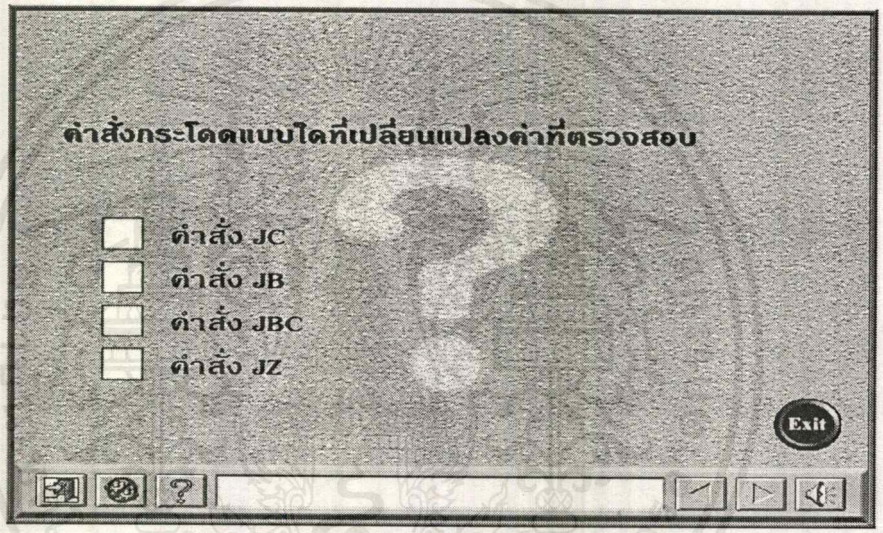


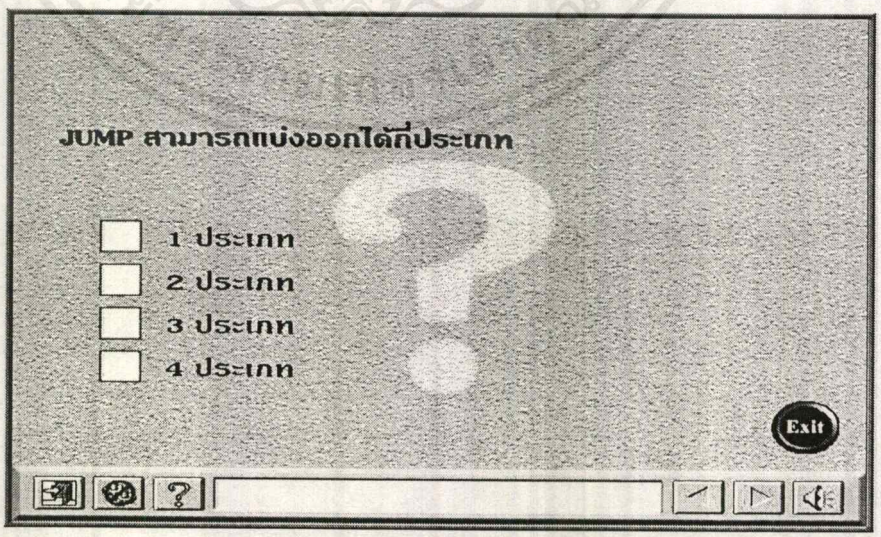


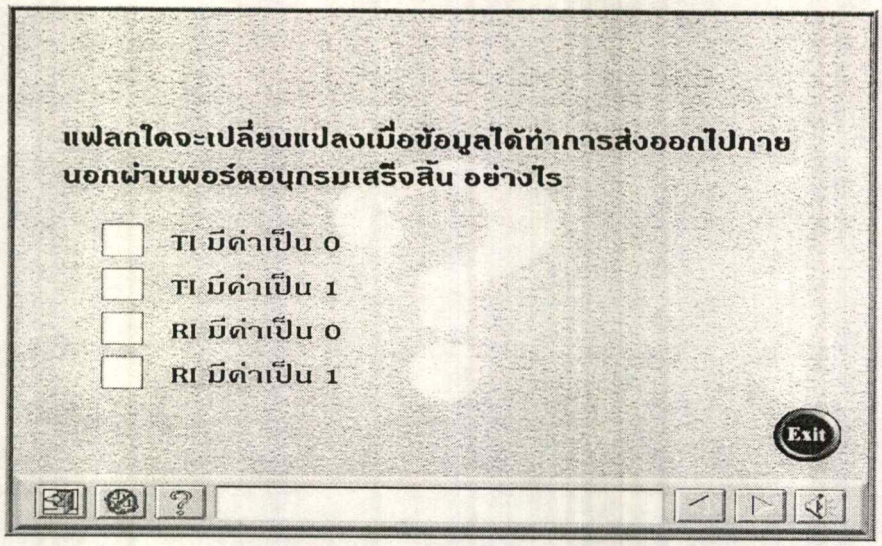












สรุป

จากการทดสอบการใช้งานของระบบคอมพิวเตอร์ช่วยสอนของคณะผู้จัดทำ สรุปได้ว่าทั้งเนื้อหาและรูปแบบการนำเสนอ มีความเหมาะสมที่จะให้ผู้สนใจได้ศึกษาหาความรู้ ได้มากพอสมควรทางคณะผู้จัดทำหวังเป็นอย่างยิ่งว่า บทเรียนช่วยสอนนี้จะสามารถให้ความรู้แก่ผู้สนใจเป็นอย่างดี

บทที่ 6
สรุปและวิจารณ์

จากการทดสอบการใช้งานของชิงเกิลบอร์ดทั้งทางฮาร์ดแวร์ ,และซอฟแวร์ ที่ผ่านมาแล้วนั้น ทำให้ผู้ใช้งานสามารถนำชิงเกิลบอร์ดในโครงงานนี้ไปใช้ประโยชน์ได้โดยไม่ต้องอาศัยเครื่องมืออื่น ๆ มากนัก เพราะนอกจากจะมีเครื่องมือ หรือ ทูล ที่จำเป็น เหมือนชิงเกิลบอร์ดทั่วไป ในท้องตลาดแล้ว ยังมีเครื่องมือช่วยในการเขียนโปรแกรมภาษาแอสเซมบลีด้วยในตัว ทั้งเอดิเตอร์, แอสเซมเบลอ อีกทั้งมีการจัดวางคีย์ให้คล้ายคีย์บอร์ดบนคอมพิวเตอร์ส่วนบุคคลทั่วไปให้มากที่สุด เพื่อเพิ่มความสะดวกในการใช้งานและค้นหาตำแหน่งของคีย์ ทำให้การใช้งานแบบ Stand alone มีประสิทธิภาพมากยิ่งขึ้น

ส่วนการสร้าง CAI ก็เหมาะและเป็นประโยชน์สำหรับผู้ที่มี PC แต่ไม่มีความรู้ทางด้าน MCS-51 ก็สามารถเรียนด้วยตนเองได้

อย่างไรก็ตามเนื่องจากเวลาในการทำโครงงานนี้มีจำกัด ในขณะที่จำนวนเนื้องานที่ต้องทำก็มีมาก ทั้งการศึกษาข้อมูลเกี่ยวกับ MCS-51 ทั้งหมดเพื่อจัดลงในเนื้อหาของ CAI, การคิดหาวิธีและรูปแบบในการสอนของเนื้อหาใน CAI แต่ละหน้าซึ่งมีจำนวนมากพอสมควร และขั้นตอนในการสร้าง CAI จัด และตกแต่งแต่ละหน้าต้องใช้เวลามาก รวมทั้งในขั้นตอนการนำแต่ละหน้ามาเชื่อมเข้าด้วยกันจะมีความยุ่งยากและปัญหามาก จึงเป็นไปได้ที่อาจจะมีข้อผิดพลาดบางประการ ซึ่งทางผู้จัดทำต้องขออภัยล่วงหน้าและจะได้หาโอกาสดำเนินการแก้ไขต่อไป

แนวทางการพัฒนา

ปัญหาที่เกิดขึ้น คือ การทำเสียงที่เครื่อนไหวตามเสียงโดยใช้ โปรแกร Authorware จะมีปัญหามากเนื่องจาก

1. ในกรณีมีการใช้เสียง wave จากการกดปุ่ม จะหยุดเสียง wave ของการสอนโดยอัตโนมัติ แต่การเคลื่อนที่ของ cursor ยังทำงานอยู่จึงไม่สำพันธกัน

2. ในกรณีที่เรียนจบแล้วรอจะเปลี่ยนหน้า แต่กำหนดให้สามารถเล่นใหม่อีกครั้งทำให้การกดอีกครั้งภาพที่แสดงอาจหายไปทำให้ การสอนไม่ถูกต้อง แก้โดยการจะมีการรออยู่ในหน้าที่สอนนั้น แต่เมื่อมีการกดเปลี่ยนโหมดการเปลี่ยนหน้าอัตโนมัติไปมา จะทำให้มีการเลื่อนหน้าเร็วหลาดหน้ากว่าจะทำงานได้ปกติ

การแก้ไข ถ้าต้องการจะทำให้มีการเลื่อนของ cursor จะทำได้โดยสร้างให้เป็น movie file ในการสอนแทนจะแก้ปัญหารนี้ได้ แต่เนื่องจากเวลาน้อยในการทำงานจึงไม่สามารถศึกษาการสร้าง movie file เพื่อมาแก้ไขได้ทัน

โปรแกรมที่ใช้สร้างจะเป็นโปรแกรมประเภทดังนี้

1. Macromedia Director , 2. Adobe Premier

โดย Authorware จะสามารแสดงข้อมูลของ Movie file ได้ดังนี้

Macromedia Director files (DIR, DXR)

Microsoft Video for Windows (AVI) files

Macintosh QuickTime files

QuickTime for Windows (MOV) files

Autodesk Animator and Autodesk Animator Pro (FLC, FLI, CEL) files

MPEG files

Bitmap Sequence files

การสร้างคำถามโดยผู้ใช้

เราต้องกำหนดก่อนว่าจะให้ผู้ใช้ตั้งถามได้ยาวสูงสุดเท่าใด จากนั้นกำหนดรูปแบบที่จะใช้เก็บบน text file โดยบันทัดแรกเป็นคำถาม โดยมีเลขคำถามอยู่ด้านหน้ากำกับไว้(โดยไม่ต้องมีจุด) บันทัดต่อไปเป็นคำตอบ 4 ข้อโดยแต่ละข้อใช้บันทัดเดียว และมีเครื่องหมายกำกับข้างหน้า

เครื่องหมาย "-" จะเป็นคำตอบที่ผิด , เครื่องหมาย "+" จะเป็นคำตอบที่ถูก

จากนั้นเราก็สร้าง Authorware ที่อ่าน file ที่กำหนด และใช้เก็บข้อมูลลงในตัวแปลเพื่อนำออกแสดง

เอกสารอ้างอิง

1. แผนกหนังสือพิเศษด้านอิเลคทรอนิคส์ , "เข้าใจ/สร้าง/เล่น ไมโครโปรเซสเซอร์ เล่ม 1" , บ. ซีเอ็ดยูเคชัน จำกัด (มหาชน) , 219 หน้า , 2538

2. ผศ. สมยศ จุณณะปิยะ , " การประยุกต์ใช้งานไมโครคอนโทรลเลอร์ตระกูล MCS-51 " ภาควิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2537

3. สุนทร วิทูสุรพจน์ , "การใช้งานไมโครคอนโทรลเลอร์ ตระกูล 8051" , บ. ซีเอ็ดยูเคชัน จำกัด (มหาชน) , 180 หน้า , 2537

4. ปรเมษฐ์ ประนยานันทน์ , "คู่มือและการประยุกต์ใช้งานไมโครคอนโทรลเลอร์ MCS-51" บ. ซีเอ็ดยูเคชัน จำกัด (มหาชน) , 380 หน้า , 2536

5. ทีมงาน ETT , "ET-BOARD V4.0 New Generation USER' S MANUAL" , บริษัท อีทีที จำกัด , 282 หน้า , 2535

6. Kenneth J. Ayala , "The 8051 Microcontroller Architecture,Programming and Application" , West Publishing company , 241 p. , 1991