บทที่ 4

ระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ด

4.1 แนวความคิดในการออกแบบระบบ

จากแนวความคิดที่พยายามจะลดเวลาที่ใช้ในการออกแบบระบบไมโครโปรเซสเซอร์ ทำให้เกิดแนวความคิดที่ว่า น่าจะมีระบบไมโครโปรเซสเซอร์ที่เป็น Universal ซึ่งสามารถใช้ได้ทั้งในด้านงานวิจัยพัฒนาระบบและในด้านการใช้งานจริง ทั้งยังขยายฟังก์ชันการทำงานได้มากและมีความยืดหยุ่นสูง จึงได้สร้างระบบไมโครโปรเซสเซอร์ในรูปแบบของระดับบอร์ดขึ้น โดยออกแบบสร้างบอร์ดต่าง ๆ ไว้เป็นมาตรฐานล่วงหน้า และปราศจากจุดบกพร่อง (Bug) ผู้ใช้เพียงแต่ออกแบบระบบด้วยการเลือกบอร์ดที่ต้องการใช้ มาประกอบเป็นวงจรบนแผ่นสัญญาณด้านหลังซึ่งยึดติดอยู่กับการ์ดแร็ก (Card Rack) แล้วสามารถพัฒนาซอฟต์แวร์บนระบบนี้ได้ทันที โดยระบบที่สร้างขึ้นนี้จะมีอุปกรณ์ช่วยในการดีบักซอฟต์แวร์รวมอยู่ด้วย เราจึงเรียกระบบที่สร้างขึ้นนี้ว่า ระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ด หรือ Board-Level Microprocessor Development System ซึ่งหลังจากพัฒนาฮาร์ดแวร์และซอฟต์แวร์จนเป็นที่แน่ใจแล้ว ผู้ใช้สามารถนำระบบไปใช้งานจริงได้

ระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดที่สร้างขึ้นได้ออกแบบอิงกับ STD บัสขนาด 8 บิต หรือ IEEE-961 บัสเพื่อใช้เป็นบัสมาตรฐานในการเชื่อมการติดต่อระหว่างบอร์ดต่าง ๆ และได้เลือกใช้ไมโครโปรเซสเซอร์เบอร์ Z-80 และ 8085 เป็นซีพียูของระบบ การที่เลือก Z-80 และ 8085 มาพัฒนานั้น เนื่องจากนิยมใช้กันในระบบควบคุมขนาดเล็กและขนาดกลาง และยังเป็นเบอร์ที่นิยมศึกษาเป็นพื้นฐานเบื้องต้นของผู้ที่เริ่มสนใจระบบไมโครโปรเซสเซอร์ นอกจากนี้ยังมีซอฟต์แวร์สนับสนุนการพัฒนาโปรแกรมของระบบที่ใช้ Z-80 และ 8085 อีกมากมาย เช่น A8MZ80, ASM8085, PL/M85 และ MCC85 เป็นต้น

4.2 เหตุที่เลือกใช้ STD บัสในระบบพัฒนา

ระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดประกอบด้วยบอร์ดต่าง ๆ ซึ่งแต่ละบอร์ดในระบบจะเชื่อมต่อสัญญาณกันโดยอาศัยบอร์ดแม่ หรือแผ่นสัญญาณด้านหลังที่นิยมออกแบบให้อิงกับบัสมาตรฐานใดมาตรฐานหนึ่ง ในงานวิจัยครั้งนี้ได้เลือก STD บัส (Simple To Designed Bus) เป็นบัสของระบบ โดยมีเหตุผลดังต่อไปนี้

1. STD บัสเป็นบัสที่นิยมใช้กันอย่างแพร่หลายในงานควบคุมทางอุตสาหกรรม และการรวบรวมข้อมูล โดยเฉพาะพวก Embedded Control System

2. มีผู้ผลิตบอร์ดมาตรฐานมากกว่า 2,000 ชนิดที่ออกแบบอิงกับ STD บัสซึ่งสามารถรองรับการใช้งานได้เกือบทุกฟังก์ชัน ผู้ใช้สามารถใช้บอร์ดที่ผลิตจากผู้ผลิตต่างรายกันมาทำงานร่วมกันในระบบหนึ่ง ๆ ได้โดยไม่จำเป็นต้องผูกติดกับผู้ผลิตรายใดเลย

3. STD บัสเป็นบัสที่ใช้ง่าย เพราะมีการกำหนดคุณสมบัติทั้งทางกล และทางไฟฟ้าค่อนข้างแน่นอน

4. มี Cost-Per-Function ต่ำที่สุด ประมาณ 1/3 ถึง 1/2 เท่าของบอร์ดทางอุตสาหกรรมที่มีสมรรถนะเหมือนกันแต่อิงกับระบบบัสอื่น ๆ

5. เป็น "Open Bus" คือ ใครก็สามารถออกแบบสร้างบอร์ดที่อิงกับ STD บัสได้โดยไม่ต้องมีใบอนุญาตหรือไม่ต้องซื้อลิขสิทธิ์ใด ๆ

6. STD บัสมีความยืดหยุ่นสูง สามารถขยายระบบได้ง่ายและใช้กับบอร์ดจำนวนมากได้ทั้งยังมีบอร์ดสนับสนุนให้ผู้ใช้สร้างฟังก์ชันทำงานเฉพาะ (Special function)ได้เอง

7. STD บอร์ด มีขนาดเล็ก และถูกตรึงไว้ทั้ง 3 ด้าน จึงไม่บิดงอหรือแตกหักง่ายจากการกระแทกและการสั่นสะเทือน (Vibration)

8. ทนต่อการช็อค (Shock) ทางไฟฟ้า และไม่ถูกสัญญาณอื่นรบกวนง่ายเหมือนคอมพิวเตอร์ทั่วไป

9. STD บัส มีการพัฒนาให้ใช้กับไมโครโปรเซสเซอร์ขนาด 16 บิต และ 32 บิตแล้วยังใช้กับระบบมัลติโปรเซสเซอร์ (Multiprocessor) ได้อีกด้วย

4.3 <u>รูปแบบของระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ด</u>

ดังที่ได้กล่าวมาแล้วว่าระบบพัฒนาไมโครโปรเซสเซอร์ (MDS) ประกอบด้วยส่วนที่เป็นฮาร์ดแวร์และส่วนที่เป็นซอฟต์แวร์ ส่วนที่เป็นฮาร์ดแวร์ของระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดที่สร้างขึ้นนี้ ได้แก่

1. ส่วนที่เป็นอุปกรณ์พื้นฐาน ในการสร้างระบบไมโครโปรเซสเซอร์ที่ผู้ใช้ต้องการจะพัฒนา (Target System) ซึ่งก็คือบอร์ดต่าง ๆ ในระบบเช่น บอร์ดซีพียูและหน่วยความจำ บอร์ดไอ/โอ เป็นต้น โดยอาจจะเป็นบอร์ดที่สร้างขึ้นในงานวิจัยครั้งนี้ หรือเป็นบอร์ดมาตรฐานต่าง ๆที่มีขายในท้องตลาดซึ่งอิงกับ STD บัส

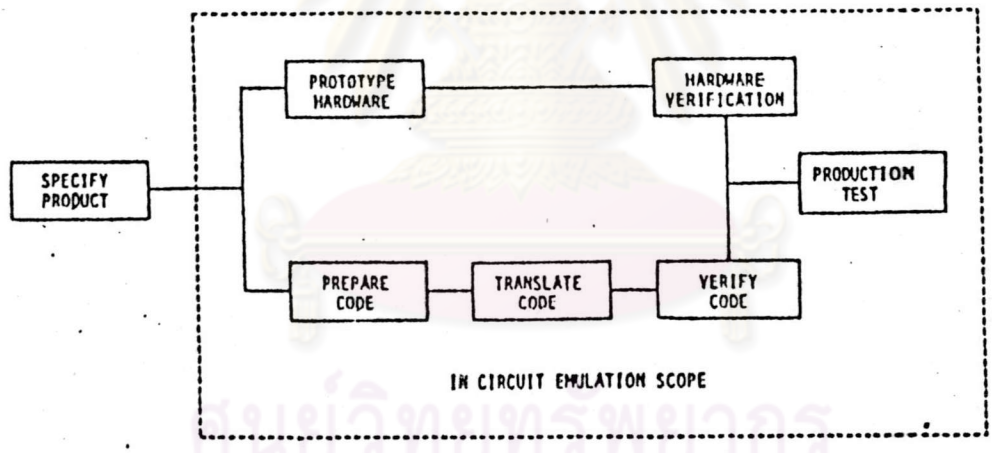
2. ส่วนที่เป็นอุปกรณ์ช่วยในการพัฒนาระบบไมโครโปรเซสเซอร์ เพื่อให้ระบบที่สร้างขึ้นมีประสิทธิภาพในการใช้งานมากยิ่งขึ้น อุปกรณ์ช่วยในการพัฒนาระบบไมโครโปร

เซสเซอร์ที่มักใช้กันบ่อย ๆ ได้แก่ ออสซิโลสโคป และลอจิกโพรบ อุปกรณ์เหล่านี้เป็นอุปกรณ์ช่วยในการพัฒนาในด้านฮาร์ดแวร์คือใช้ตรวจจับสัญญาณ แต่ก็ไม่สามารถตรวจสอบสถานะของบัสข้อมูล บัสแอดเดรสและสัญญาณบางอย่างได้จึงต้องอาศัยเครื่องมือที่มีประสิทธิภาพสูงอันได้แก่ เครื่องวิเคราะห์ลอจิก (Logic Analyzer) ซึ่งมีราคาแพงมากและใช้งานได้ยาก ส่วนอุปกรณ์ช่วยในการพัฒนาทางด้านซอฟต์แวร์ ได้แก่ รอมอีมูเลเตอร์ (ROM Emulator) เครื่องไมโครคอมพิวเตอร์แผ่นวงจรพิมพ์เดี่ยว(Single-Board Microcomputer) และอินเซอร์กิตอีมูเลเตอร์ (ICE) โดยเครื่องไมโครคอมพิวเตอร์แผ่นวงจรพิมพ์เดี่ยว และ ICE สามารถใช้ดีบักได้ทั้งฮาร์ดแวร์และซอฟต์แวร์ เราสามารถเปรียบเทียบความสามารถในการพัฒนาซอฟต์แวร์โดยใช้อุปกรณ์ต่าง ๆ ได้ดังตารางที่ 4.1 [27]

ตารางที่ 4.1 แสดงข้อเปรียบเทียบความสามารถในการใช้ Single-Board, ROM Emulator และ ICE ในการพัฒนาซอฟต์แวร์

ข้อพิจารณา	Single Board	ROM Emulator	ICE
1.การติดต่อกับหน่วยความจำได้ทั้ง 64 K	ไม่ได้	ได้	ได้
2.การติดต่อกับพอร์ตได้ทั้ง 256 พอร์ต	ไม่ได้	ได้	ได้
3.การอ่านเขียนหน่วยความจำจำโดยตรง	ได้	ไม่ได้	ได้
4.การอ่านค่าและตั้งค่ารีจีสเตอร์	ได้	ไม่ได้	ได้
5.การใช้งานสัญญาณควบคุม เช่น NMI,INT,BUSRQ	ไม่ได้	ไม่ได้	ได้
6.การ Run Real Time	ได้	ได้	ได้
7.การกำหนด Breakpoint	ได้	ไม่ได้	ได้
8.การทำ Assembler	ไม่ได้	ได้	ได้
9.การทำ Disassembler	ไม่ได้	ไม่ได้	ได้

จะเห็นว่า รอมอีมูเลเตอร์ และเครื่องไมโครคอมพิวเตอร์แผ่นวงจรพิมพ์เดี่ยวนั้น มีขีดความสามารถค่อนข้างจำกัดเช่น ความสามารถของซีพียูและหน่วยความจำของระบบที่กำลังพัฒนาถูกนำมาใช้ได้ไม่เต็มที่ การหาความผิดพลาดของโปรแกรมทำได้จำกัดโดยเฉพาะในส่วนที่เกี่ยวกับการหยุดทำงานอย่างมีเงื่อนไข (Conditional Breakpoint) และ Real Time Trace นอกจากนี้ยังมีขีดจำกัดในการหาความผิดพลาดของวงจรอีก ส่วน ICEเป็นเครื่องมือช่วยในการดีบักที่มีประสิทธิภาพมากที่สุดเพราะสามารถพัฒนาฮาร์ดแวร์และซอฟต์แวร์ควบคู่กันไปได้เลย เดิม ICE เป็นระบบค่อนข้างใหญ่ทำให้มีราคาแพง ต่อมาได้มีการนำเอาเครื่อง PC (Personal Computer) ซึ่งมีราคาถูกมาประยุกต์ใช้งานด้วยทำให้ ICE มีราคาถูกลง ICE ทำหน้าที่เสมือนซีพียูของระบบไมโครโปรเซสเซอร์ที่ต้องการพัฒนา ซึ่งผู้ใช้สามารถสั่งให้ทำงานหรือหยุดดูค่าต่าง ๆ ได้ทั้งยังสามารถดีบักโปรแกรมในระดับภาษาแอสเซมบลี้ได้อีกด้วย ปัจจุบันผู้ใช้สามารถพัฒนาซอฟต์แวร์บน ICE ได้โดยที่ไม่ต้องสร้าง Target System มาก่อน[7],[8] [27] ฉะนั้นเราสามารถแสดงขั้นตอนการพัฒนาซอฟต์แวร์ทั่ว ๆ ไปที่ใช้ ICE ได้ดังรูปที่ 4.1



Development cycle using in-circuit emulation module

รูปที่ 4.1 แสดงขั้นตอนการพัฒนาซอฟต์แวร์ทั่ว ๆ ไปที่ใช้ ICE

ICE มีความสามารถพื้นฐานในการดีบัก ดังนี้

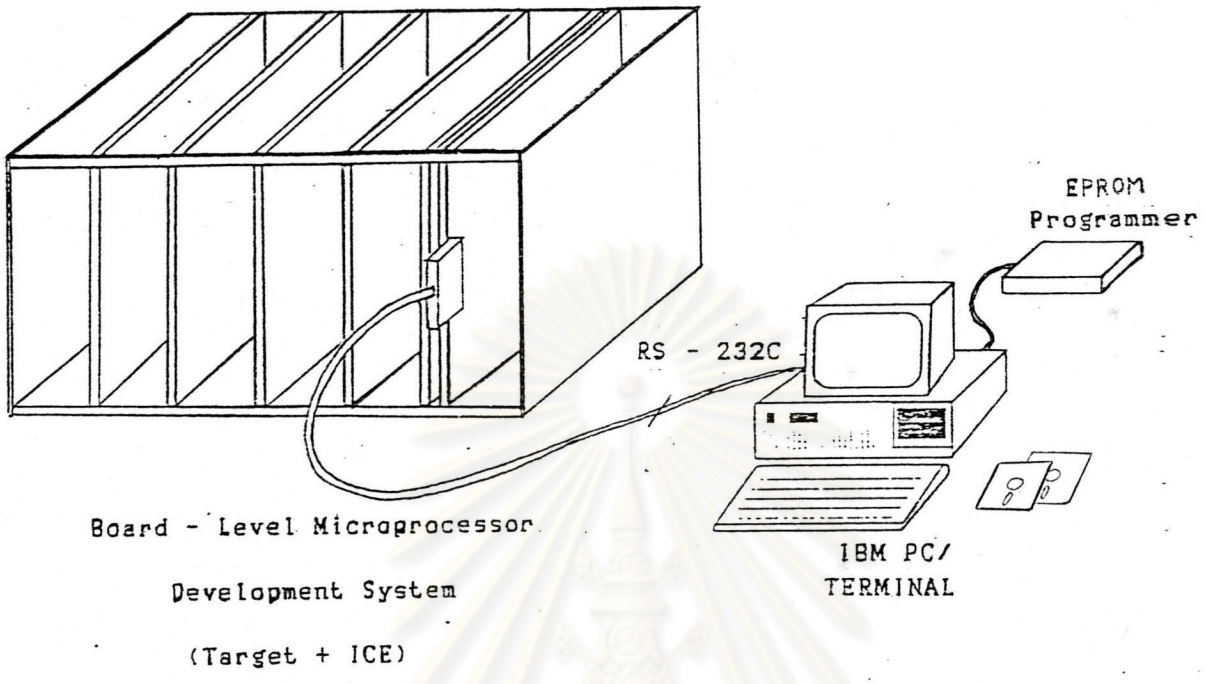
1. ตรวจสอบและแก้ไขข้อมูลในหน่วยความจำ
2. ตรวจสอบและแก้ไขข้อมูลในรีจีสเตอร์
3. ทำงานตามเวลาจริง (Real Time Execute)
4. กำหนดจุดหยุดการทำงานได้ (Breakpoint)
5. ทำงานทีละคำสั่ง (Single Step Execute)
6. อ่านและเขียนพอร์ตไอ/โอได้
7. ทำแอสเซมเบลอร์/ดิสแอสเซมเบลอร์ได้

รูปแบบโครงสร้างของ ICE โดยทั่วไปแบ่งได้เป็น 2 ประเภท [16] คือ

1. แบบซีพียูเดียว (Single Processor) ซีพียูของ ICE จะทำหน้าที่เป็นทั้งซีพียูของระบบที่จะพัฒนาและทำหน้าที่ควบคุมการทำงานของ ICE ด้วย ซีพียูดังกล่าวนี้ต้องเป็นซีพียูชนิดเดียวกับที่ใช้ในระบบที่จะพัฒนา ซึ่งปกติมักจะมีความสามารถในการดีบักโดยตัวเองอยู่ด้วย ICE แบบนี้ จะมีฟังก์ชันการทำงานน้อยกว่าแบบหลายซีพียูเพราะถูกจำกัดโดยการทำงานของซีพียูเอง และ ICE แบบนี้นิยมใช้ในระบบที่เป็นระดับบอร์ด

2. แบบหลายซีพียู (Multi Processor) ซีพียูตัวหนึ่งจะทำหน้าที่เป็นซีพียูของระบบที่จะพัฒนา ส่วนซีพียูตัวอื่น ๆ ทำหน้าที่ควบคุมการทำงานของ ICE โครงสร้าง ICE แบบนี้มีข้อดีหลายประการ เช่น การออกแบบทำได้ง่าย ผู้ใช้สามารถติดต่อกับICE ได้โดยไม่จำเป็นต้องหยุดการทำงานของระบบที่จะพัฒนา และสามารถบันทึกสถานะของบัสขณะซีพียูทำงานตามเวลาจริง(Real Time Trace)ได้ เป็นต้น ICE แบบนี้นิยมทำเป็นแบบ Stand Alone

ในงานวิจัยครั้งนี้ เลือกสร้าง ICE แบบซีพียูเดียวประกอบเข้ากับระบบไมโครโปรเซสเซอร์ระดับบอร์ด แม้ว่าจะมีฟังก์ชันการทำงานน้อยกว่าแบบหลายซีพียูแต่ก็มีความสามารถในการดีบักพอสมควร โดยสร้าง ICE ที่เป็นระดับบอร์ด ซึ่งเราเรียกว่า บอร์ดอีมูเลชัน (Emulation Board) และได้เลือกออกแบบให้ใช้งานร่วมกับเครื่องไมโครคอมพิวเตอร์ IBM PC ซึ่งช่วยให้ผู้ใช้สามารถดีบักซอฟต์แวร์ในระดับภาษาที่สูงกว่าระดับภาษาเครื่อง เหตุที่เลือกใช้กับเครื่องไมโครคอมพิวเตอร์ IBM PCเนื่องจากเป็นที่นิยมใช้กันแพร่หลายและมีระบบพัฒนาไมโครโปรเซสเซอร์ส่วนที่เป็นซอฟต์แวร์มากมาย เช่น แอสเซมเบลอร์ คอมไพล์เลอร์ ลิงก์เกอร์และซีมูเลเตอร์ เป็นต้น ปกติ ICE นอกจากจะใช้ได้กับเครื่องไมโครคอมพิวเตอร์แล้วยังใช้ได้กับเทอร์มินัล แต่การดีบักโดยใช้เครื่องไมโครคอมพิวเตอร์สะดวกกว่าเนื่องจากมีอุปกรณ์สนับสนุนมากมายทั้งหน่วยความจำของเครื่องไมโครคอมพิวเตอร์ จานเก็บบันทึกข้อมูล (Disk) และโปรแกรมสนับสนุนต่าง ๆ ระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดนี้ได้เลือกออกแบบส่วน ICE ให้ทำงานร่วมกับเครื่องไมโครคอมพิวเตอร์IBM PC หรือเครื่อง IBM PC Compatible และยังออกแบบให้ใช้กับเทอร์มินัล VT-100ได้ด้วย ผู้ใช้เพียงแต่มีระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดก็เสมือนมีทั้ง Target System และ ICE ซึ่งสามารถออกแบบและพัฒนาระบบที่ใช้ไมโครโปรเซสเซอร์ Z-80 หรือ 8085 ได้ นอกจากนี้ระบบพัฒนาไมโครโปรเซสเซอร์ยังต้องมีเครื่องมือในการโปรแกรมอีพรอม เพื่อนำโปรแกรมที่พัฒนาเรียบร้อยแล้วไปใช้ใน Target System และเครื่องพิมพ์ เพื่อการเก็บบันทึกข้อมูลข่าวสารต่าง ๆ เราสามารถแสดงระบบทั้งหมดได้ดังรูปที่ 4.2



รูปที่ 4.2 แสดงองค์ประกอบทั้งหมดของระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ด

4.4 คุณสมบัติของระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ด

ระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดที่จัดสร้างขึ้น จะออกแบบให้มีคุณสมบัติดังต่อไปนี้

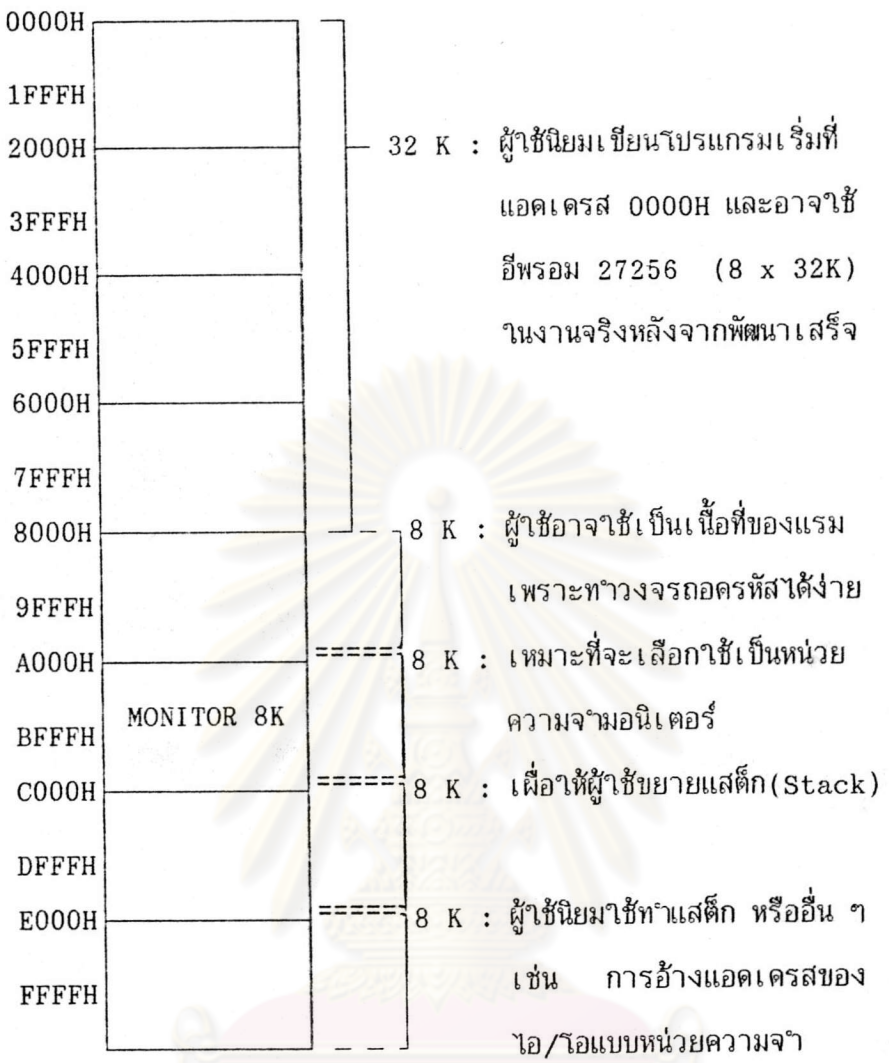
1. มีหน่วยความจำของผู้ใช้ 64 กิโลไบต์เต็ม
2. สามารถเลือกหน่วยความจำรอม/แรมได้ทุกขนาด
3. มี ICE ระดับบอร์ด ที่มีความสามารถในการดีบักร่วมกับเครื่องไมโครคอมพิวเตอร์ IBM PC

4.4.1 หน่วยความจำของผู้ใช้ 64 K เต็ม

จากแนวความคิดที่เลือกสร้าง ICE แบบซีพียูเดียว ประกอบเข้ากับระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ด ทำให้ซีพียูของระบบต้องสลับกันทำงานระหว่างโปรแกรมของผู้ใช้และโปรแกรมมอนิเตอร์ซึ่งเป็นโปรแกรมช่วยในการดีบัก การอ้างแอดเดรสหน่วยความจำของโปรแกรมมอนิเตอร์และหน่วยความจำของผู้ใช้นั้นถูกออกแบบให้อยู่บนหน่วยความจำ

ของระบบเพจ (Page)เดียวกันหรือคนละเพจขึ้นอยู่กับขนาดของโปรแกรมและขบวนการใช้งาน ปกติแล้วระบบไมโครโปรเซสเซอร์ขนาด 8 บิต สามารถอ้างแอดเดรสหน่วยความจำได้ 64 K (หรือ 2^{16} bytes) และอ้างแอดเดรสพอร์ตไอ/โอได้ถึง 256 พอร์ต (หรือ 2^{8}) แต่ส่วนใหญ่การประยุกต์ใช้งานระบบไมโครโปรเซสเซอร์ที่ใช้ Z-80 หรือ 8085 เป็นซีพียูจะมีขนาดโปรแกรมของผู้ใช้ประมาณ 8-10 กิโลไบต์ [10] จึงทำให้เครื่องมือช่วยในการดีบักอย่างเครื่องไมโครคอมพิวเตอร์แผ่นวงจรพิมพ์เดี่ยว หรือ ICE บางรุ่นแบ่งเนื้อที่หน่วยความจำส่วนหนึ่งของระบบมาเป็นโปรแกรมมอนิเตอร์ ทำให้ผู้ใช้ไม่สามารถเขียนโปรแกรมที่เนื้อที่นั้นได้ อย่างเช่น เครื่องไมโครคอมพิวเตอร์แผ่นวงจรพิมพ์เดี่ยว MPF-1 [17] หรือ SDA-85[18] มีโปรแกรมมอนิเตอร์อยู่ที่ 0000H-17FFH และให้โปรแกรมของผู้ใช้เริ่มที่แอดเดรส 1800H ส่วน ET Board [19] ออกแบบให้โปรแกรมผู้ใช้อยู่ที่ 1000H-1F7FH เท่านั้นซึ่งมีเนื้อที่จำกัดมากและเมื่อนำโปรแกรมไปใช้ในวงจรจริง อาจจะต้องทำการออกแบบฮาร์ดแวร์เพิ่มเพื่อให้เมื่อเปิดเครื่องระบบเริ่มทำงานที่โปรแกรมแอดเดรสนั้น ๆ แทนที่จะเริ่มทำงานที่แอดเดรส 0000H ตามซีพียู และในปัจจุบันมีแนวโน้มว่าขนาดโปรแกรมของผู้ใช้มีขนาดใหญ่ขึ้น เนื่องจากนิยมออกแบบโปรแกรมเป็นโมดูล (Module) และเขียนโปรแกรมด้วยภาษาชั้นสูงเพราะหน่วยความจำมีราคาถูกลง การเขียนโปรแกรมไม่จำเป็นต้องมีขนาดเล็กแต่เน้นว่าโปรแกรมต้องไม่มีบักและตรวจสอบแก้ไขได้ง่าย ดังนั้นขนาดโปรแกรมผู้ใช้เมื่อแปลมาเป็นภาษาเครื่องจึงมีขนาดใหญ่ หากเครื่องมือช่วยในการดีบักใช้โปรแกรมมอนิเตอร์อยู่บนหน่วยความจำเพจเดียวกับโปรแกรมผู้ใช้อาจมีเนื้อที่ไม่เพียงพอสำหรับโปรแกรมผู้ใช้ จึงคิดว่าโปรแกรมของผู้ใช้คงมีขนาดใหญ่ไม่เต็ม 64 K ถ้าหากแบ่งเนื้อที่ 8 K ให้เป็นโปรแกรมมอนิเตอร์เหลืออีก 56 K สำหรับโปรแกรมผู้ใช้ก็จะมีวิธีทำได้ 2 แบบ คือ

แบบที่ 1 ให้โปรแกรมมอนิเตอร์มีขนาด 8 K และกำหนดแอดเดรสเริ่มต้นของโปรแกรม 8 K นี้ให้แน่นอน แต่จะไม่ใช้ที่ 0000H เพื่อเปิดโอกาสให้ผู้ใช้สามารถออกแบบพัฒนาโปรแกรมของตนเองเริ่มที่แอดเดรส 0000H ได้ และจะได้ไม่ต้องทำฮาร์ดแวร์ส่วนที่บังคับให้ซีพียูเริ่มทำงานตามโปรแกรมในตำแหน่งอื่นที่ไม่ใช่แอดเดรส 0000H เมื่อเริ่มเปิดเครื่องหรือถูกรีเซต จากการวิจัยหาข้อมูลสรุปได้ว่า เนื้อที่ 8 K ที่แอดเดรส A000H-BFFFH เหมาะสมที่จะเป็นตำแหน่งของโปรแกรมมอนิเตอร์ดังแสดงในรูปที่ 4.3



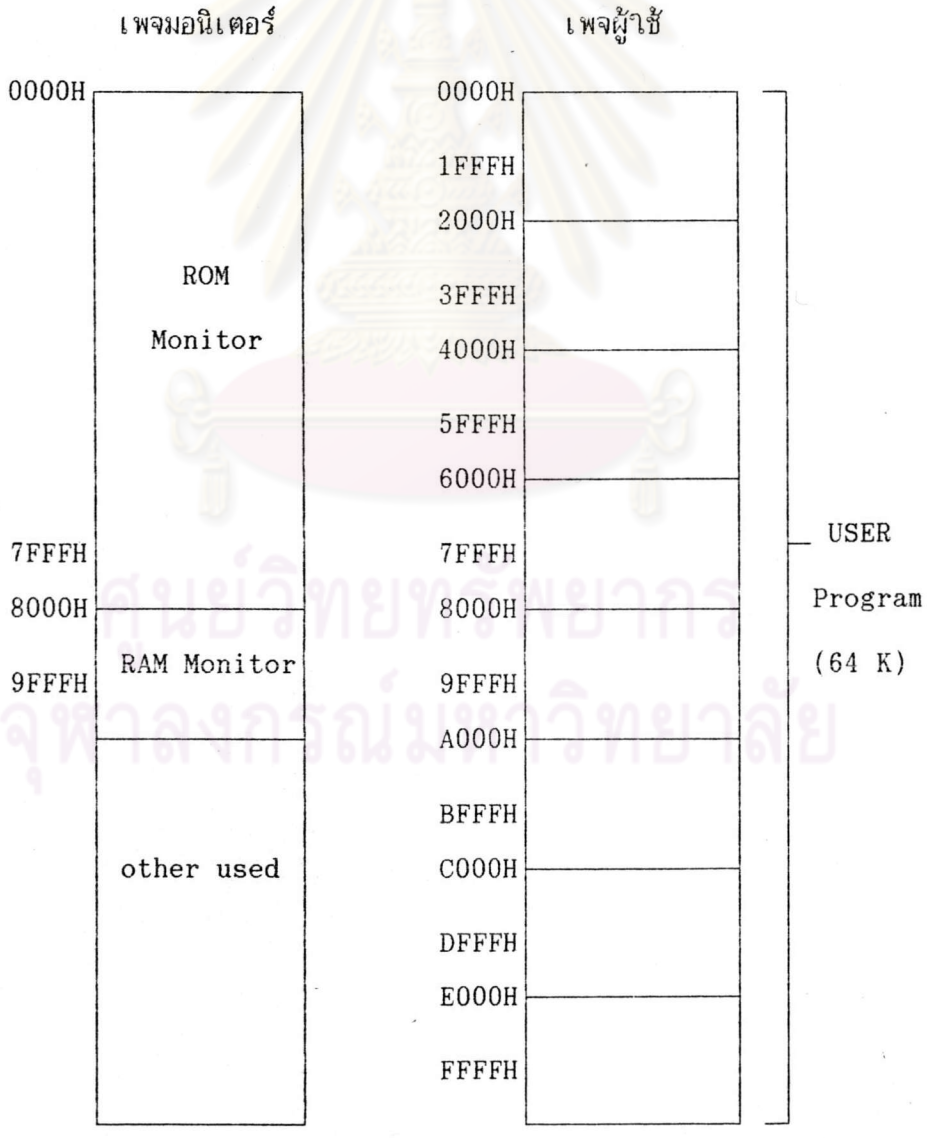
รูปที่ 4.3 แสดงการออกแบบตำแหน่งมอนิเตอร์ที่ A000H - BFFFH

แบบที่ 2 ให้โปรแกรมมอนิเตอร์มีขนาด 8 K และเปลี่ยนตำแหน่งเริ่มต้นของโปรแกรมมอนิเตอร์ได้ด้วยการเลือกดิปสวิตช์(DIP Switch) เพื่อความสะดวกของผู้ใช้ในการออกแบบโปรแกรม แต่วิธีนี้มีความยุ่งยากในการออกแบบโปรแกรมมอนิเตอร์เป็นอย่างมาก เพราะค่าต่าง ๆที่อ้างอิงในโปรแกรมต้องเป็นค่าแบบสัมพัทธ์ ทั้งยังเหลือเนื้อที่หน่วยความจำให้ผู้ใช้ใช้เพียง 56 K จึงไม่เลือกทำวิธีนี้

การที่ขนาดโปรแกรมของผู้ใช้มีแนวโน้มว่าจะขยายใหญ่ขึ้นเรื่อย ๆ จึงเกิดความคิดใหม่ว่า ควรออกแบบให้หน่วยความจำของผู้ใช้มีขนาดได้ 64 K เต็มตามความสามารถของไมโครโปรเซสเซอร์ขนาด 8 บิตที่สามารถอ้างแอดเดรสของหน่วยความจำได้ ส่วนโปรแกรมมอนิเตอร์ให้ออกแบบไว้เป็นอีกเพจหนึ่งซ้อนอยู่ในตำแหน่งเริ่มต้นที่ 0000H เช่นกัน ซึ่งทำให้เราสามารถขยายขนาดของโปรแกรมมอนิเตอร์ออกไปได้อย่างไม่มีข้อจำกัด เพราะเดิมเรา

ออกแบบโปรแกรมมอนิเตอร์ให้มีขนาด 8K และให้โปรแกรมในการติดต่อกับผู้ใช้อีกส่วนหนึ่งอยู่บนเครื่องไมโครคอมพิวเตอร์

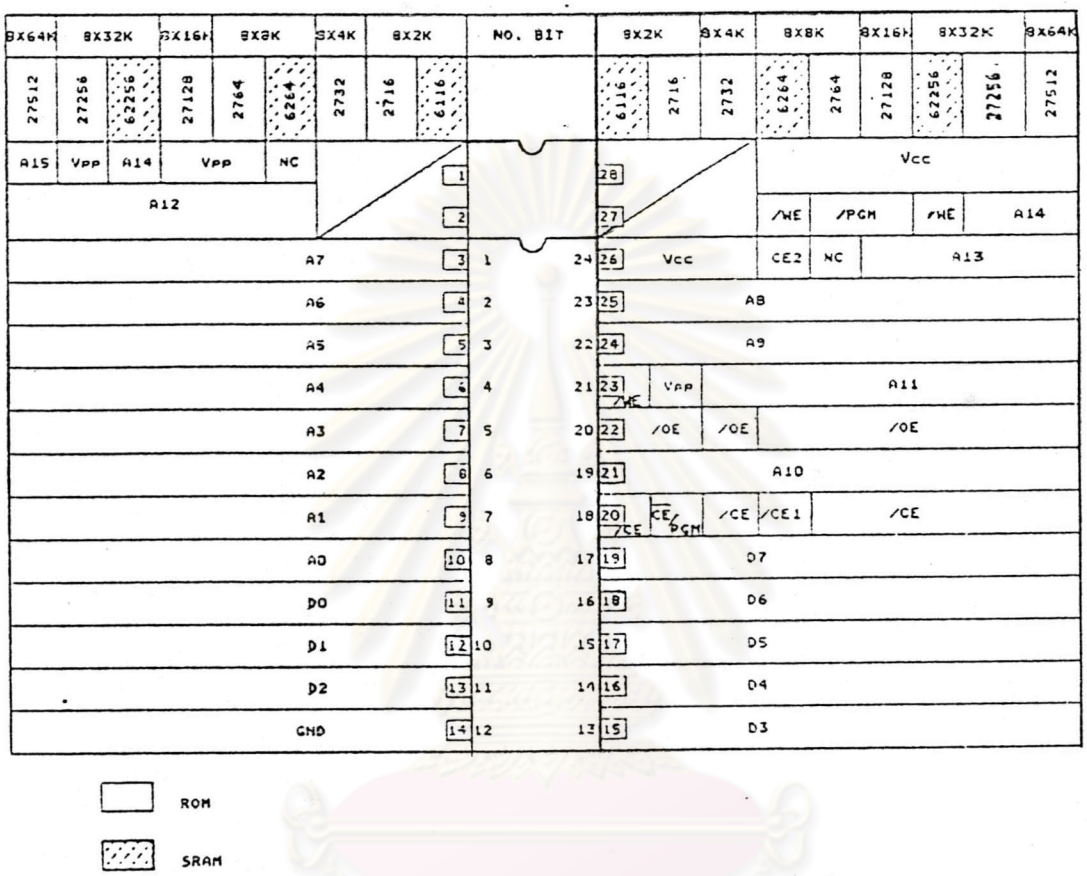
ฉะนั้น สำหรับระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดนี้ได้ออกแบบหน่วยความจำของระบบเป็น 2 เพจ เพจหนึ่งสำหรับโปรแกรมของผู้ใช้ซึ่งอ้างแอดเดรสได้ 64 K ส่วนอีกเพจหนึ่งเป็นเนื้อที่สำหรับโปรแกรมมอนิเตอร์ซึ่งจะอยู่ในส่วนของการอีมูเลชันดังรูปที่ 4.4 เมื่อใช้ระบบพัฒนานี้ช่วยในการดีบักโปรแกรม เมื่อเปิดเครื่อง ซีพียูจะเริ่มทำงานที่โปรแกรมมอนิเตอร์หลังจากนั้นจึงมีการสลับเพจจากเพจมอนิเตอร์ไปยังเพจผู้ใช้ตามคำสั่ง และอาจมีการสลับเพจกลับมาที่เพจมอนิเตอร์อย่างเดิมก็ได้ เทคนิคในการสลับเพจจะกล่าวรายละเอียดในบทที่ 5



รูปที่ 4.4 แสดงการออกแบบหน่วยความจำของระบบพัฒนา เพจมอนิเตอร์และเพจผู้ใช้

4.4.2 การเลือกใช้รอม/แรมได้ทุกขนาดในหน่วยความจำผู้ใช้

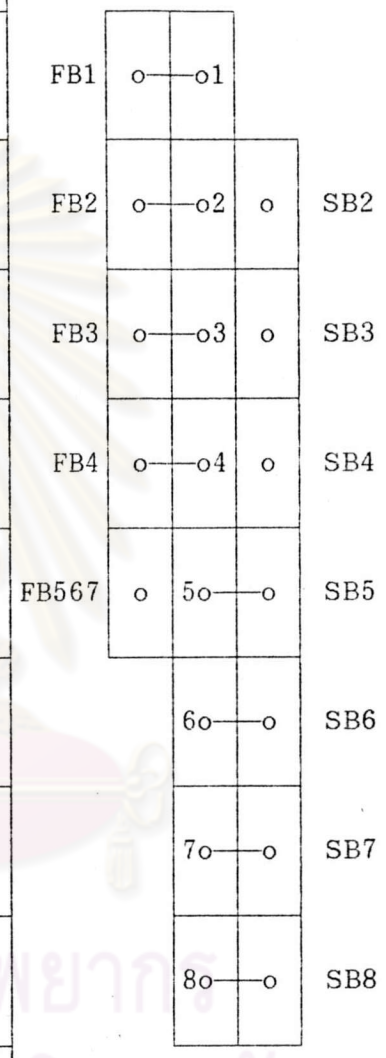
โดยทั่วไปหน่วยความจำรอม (ROM) และแรม (SRAM) ที่มักใช้คู่กับซีพียูขนาด 8 บิต มีขนาดต่าง ๆ ดังแสดงในรูปที่ 4.5 และมีตำแหน่งขาต่างกันบ้างตามรูป



รูปที่ 4.5 แสดงตำแหน่งขาต่าง ๆ ของรอม/แรม ที่มักใช้คู่กับซีพียูขนาด 8 บิต

จะเห็นว่ารอม/แรมขนาดเล็กที่สุดคือ 2 K ใหญ่ขึ้นมาก็เป็น 4 K แต่ระบบพัฒนาได้ออกแบบแบ่งหน่วยความจำของผู้ใช้ออกเป็นบล็อกย่อยที่สุด บล็อกละ 8 K เนื่องจากหน่วยความจำขนาดใหญ่มีราคาถูกลง ซอฟต์แวร์ที่ประยุกต์ใช้งานมีแนวโน้มว่าขนาดจะใหญ่ขึ้น และหน่วยความจำที่มีความจุต่ำถูกผลิตออกมาน้อยลง แต่ถึงกระนั้นผู้ใช้ก็ยังสามารถใช้รอม/แรมขนาด 2 K หรือ 4 K ในระบบพัฒนาได้ โดยการอ้างแอดเดรสยังคงเป็นบล็อกละ 8 K อยู่ ระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดนี้ออกแบบให้หน่วยความจำของผู้ใช้สามารถอ้างแอดเดรสได้เต็ม 64 K และให้ใช้หน่วยความจำรอม/แรมได้ทุกขนาดบนซอกเก็ต(Socket) 2 อัน ทั้งยังเลือกขนาด/แอดเดรสของหน่วยความจำได้ด้วยการต่อ Jumper โดย Jumper ของหน่วยความจำผู้ใช้มี 2 ชุด ชุดแรกสำหรับเลือกขนาดและแอดเดรสเริ่มต้นของแต่ละซอกเก็ต โดยออกแบบไว้ป้องกันการเลือกแอดเดรสของรอม/แรมซ้ำกันด้วย ดังแสดงในรูปที่ 4.6

Jumper	เลือกใช้ซอกเก็ต/แอดเดรส
ขา 1- FB1	Socket#1/(0000H-1FFFH)
ขา 2- FB2	Socket#1/(2000H-3FFFH)
ขา 3- FB3	Socket#1/(4000H-5FFFH)
ขา 4- FB4	Socket#1/(6000H-7FFFH)
ขา 5- FB567	Socket#1/(8000H-DFFFH)
ขา 2- SB2	Socket#2/(2000H-3FFFH)
ขา 3- SB3	Socket#2/(4000H-5FFFH)
ขา 4- SB4	Socket#2/(6000H-7FFFH)
ขา 5- SB5	Socket#2/(8000H-9FFFH)
ขา 6- SB6	Socket#2/(A000H-BFFFH)
ขา 7- SB7	Socket#2/(C000H-DFFFH)
ขา 8- SB8	Socket#2/(E000H-FFFFH)



รูปที่ 4.6 แสดงการต่อ Jumper เลือกขนาด/แอดเดรสของหน่วยความจำผู้ใช้

เหตุที่ออกแบบให้ซอกเก็ตที่ 1 ใช้ได้ทั้งรอมและแรม ส่วนซอกเก็ตที่ 2 ให้ใช้ได้แต่แรมนั้นเนื่องจากผู้ใช้นิยมออกแบบใช้รอมเริ่มต้นที่แอดเดรส 0000H ซึ่งก็คือ บล็อกแรก ส่วนบล็อกต่อ ๆ ไปอาจจะใช้รอมหรือแรมก็ได้จึงออกแบบ Jumper ให้เลือกได้ 2 ทาง แต่ถ้าผู้ใช้ใช้รอมขนาด 64 K ในซอกเก็ตที่ 1แล้วผู้ใช้จะใช้เขียนโปรแกรมได้เพียง 56 กิโลไบต์เท่านั้น ส่วนอีก 8 กิโลไบต์จะเป็นเนื้อที่แรมในซอกเก็ตที่ 2 ซึ่งจะเป็นแอดเดรสบล็อกใดก็ได้ขึ้นอยู่กับการต่อ Jumper ดังแสดงรูปที่ 4.6 ปกติเราจะใช้แรม 32 K ทั้ง 2 ซอกเก็ตเพื่อผู้ใช้สามารถโหลดโปรแกรมลงหน่วยความจำขณะทำการดีบักได้จึงเลือกต่อ Jumper ดังรูป

ส่วน Jumper ชุดที่สองสำหรับสลับขาสัญญาณของไอซีที่ใช้ในแต่ละซอกเก็ต เนื่องจากขาของไอซีรอมและแรมแต่ละเบอร์ทำหน้าที่ต่างกัน ดังแสดงในรูปที่ 4.5 จะเห็นว่าขาที่มีสัญญาณต่างกันได้แก่ (นับอิงกับขาของซอกเก็ต 28 ขา)

ขาที่ 1 เป็นได้ทั้งขาสัญญาณแอดเดรส A15, A14 และขา Vpp ซึ่งปกติต่อกับไฟเลี้ยง +5 โวลต์ จึงมี Jumper ให้เลือกได้ 3 ทาง สำหรับซอกเก็ตที่ 1 ส่วนซอกเก็ตที่ 2 นั้นให้ใช้ได้เฉพาะแรมจึงไม่จำเป็นต้องมี Jumper เลือก โดยให้ต่อสัญญาณแอดเดรส A14 เข้าขาที่ 1 ของซอกเก็ตโดยตรง

ขาที่ 20 เป็นได้ทั้งขาสัญญาณ /CE (Chip Enable, actives Low) และ PGM (Programmable) ของหน่วยความจำ แต่เนื่องจากเราไม่สามารถโปรแกรมรอมบนระบบพัฒนานี้ได้จึงต่อสัญญาณ /CE เข้าขาที่ 20 ของซอกเก็ตโดยตรง

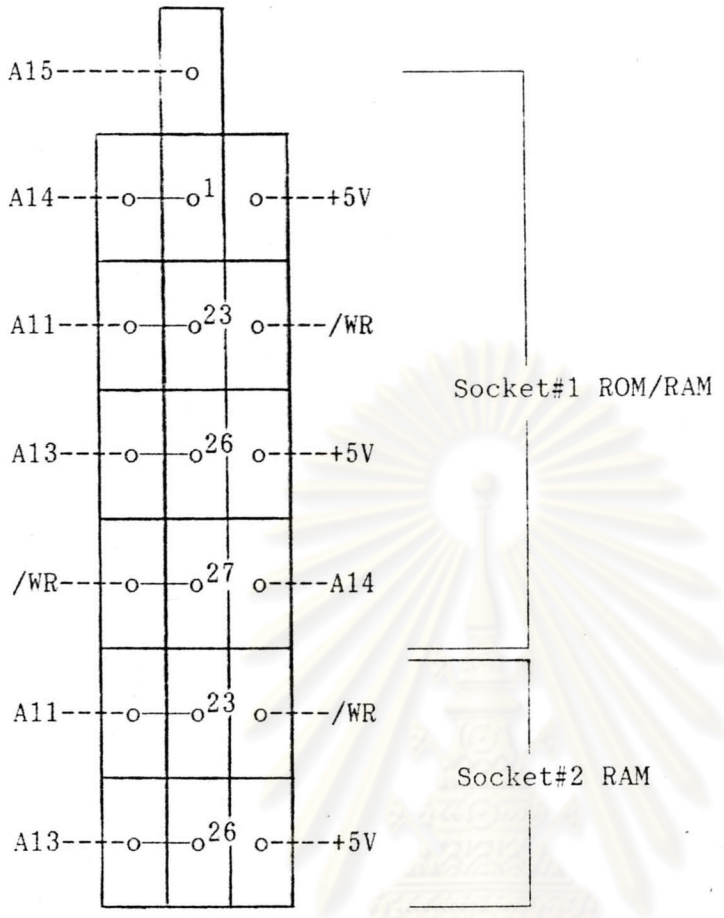
ขาที่ 22 เป็นได้ทั้งขาสัญญาณ/OE (Output Enable,actives Low) และขา Vpp ซึ่งปกติเรามักใช้สัญญาณ /RD (Read,active Low) ของซีพียูเป็นสัญญาณ /OE แต่เนื่องจากเราไม่สามารถโปรแกรมรอมบนระบบพัฒนาได้ จึงต่อสัญญาณ /RD ของซีพียูเข้าที่ขา 22 นี้โดยตรง

ขาที่23 เป็นได้ทั้งขาสัญญาณ /WE (Write Enable, actives Low) ขา Vpp และขาแอดเดรส A11 ปกติเรามักใช้สัญญาณ /WR (Write,active Low)ของซีพียูเป็นสัญญาณ /WE ของหน่วยความจำ และเนื่องจากเราไม่สามารถโปรแกรมรอมบนระบบพัฒนาได้จึงออกแบบ Jumper ให้เลือกต่อได้เพียง 2 ทางคือ ต่อกับสัญญาณ /WR หรือต่อกับแอดเดรส A11 สำหรับทั้ง 2 ซอกเก็ต

ขาที่ 26 เป็นได้ทั้งขาสัญญาณ CE2(Chip Select2,actives High) ขาแอดเดรส A13 และขาไฟเลี้ยง +5 โวลต์ ปกติเรามักให้สัญญาณ CE2 แอคทีฟตลอดเวลาโดยต่อกับ +5 โวลต์ ดังนั้น Jumper จึงถูกออกแบบให้เลือกต่อได้ 2 ทาง คือ จะต่อกับแอดเดรส A13 หรือไฟเลี้ยง +5 โวลต์ สำหรับทั้ง 2 ซอกเก็ต

ขาที่ 27 เป็นได้ทั้งขาสัญญาณ /WE , PGM และขาแอดเดรส A14 แต่เนื่องจากเราไม่สามารถโปรแกรมรอมบนระบบพัฒนาได้จึงเหลือแค่ 2 ทางเลือกและเนื่องจากซอกเก็ตที่ 1 ใช้ได้ทั้งรอมและแรมจึงต้องออกแบบ Jumper ให้เลือกได้ 2 ทางคือ จะต่อขาที่ 27 เข้ากับสัญญาณ /WR หรือจะต่อเข้ากับแอดเดรส A14 ส่วนซอกเก็ตที่ 2 เราให้ใช้ได้เฉพาะแรมดังนั้นเราจึงออกแบบให้ต่อขาที่ 27 ของซอกเก็ตที่ 2 กับสัญญาณ /WR โดยตรง

สรุปได้ว่า Jumperชุดที่สองใช้สำหรับการสลับขาสัญญาณที่เข้าสู่ขาซอกเก็ตของรอมและแรมเพื่อให้เหมาะกับชนิดและขนาดของหน่วยความจำที่ผู้ใช้เลือก ซึ่งผู้ใช้สามารถเลือกต่อ Jumper ได้ตามรูปที่ 4.7 และจะเห็นว่า Jumper ชุดที่สองนี้ประกอบด้วย Jumper ที่เลือก 3 ทาง 1 ตัว และเป็น Jumper ที่เลือก 2 ทางอีก 5 ตัววางเรียงกัน ดังแสดงในรูปที่ 4.7 โดย 4 ตัวบนเป็น Jumper สลับขาของซอกเก็ตที่ 1 ส่วน 2 ตัวล่างเป็น Jumper สลับขาของซอกเก็ตที่ 2



ชนิดของหน่วยความจำ	เบอร์	Socket#1				Socket#2	
		ขา 1	ขา 23	ขา 26	ขา 27	ขา 23	ขา 26
ROM	2716(2K)	-	/WE	+5V	-	X	X
	2732(4K)	-	A11	+5V	-	X	X
	2764(8K)	+5V	A11	-	/WE	X	X
	27128(16K)	+5V	A11	A13	/WE	X	X
	27256(32K)	+5V	A11	A13	A14	X	X
	27512(64K)	A15	A11	A13	A14	X	X
RAM	6116(2K)	-	/WE	+5V	-	/WE	+5V
	6264(8K)	-	A11	+5V	/WE	A11	+5V
	62256(32K)	A14	A11	A13	/WE	A11	A13

หมายเหตุ X:Socket#2 ใช้ได้เฉพาะแรมเท่านั้น , -:เสียบหรือไม่เสียบ Jumper ก็ได้

รูปที่ 4.7 แสดงการต่อ Jumper เพื่อสลับขาสัญญาณที่เข้าสู่ขาซอกเก็ตหน่วยความจำ

4.4.3 ความสามารถในการดีบักของ ICE ในระบบพัฒนานี้

ระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดมีจุดเด่นอยู่ที่ การมีส่วนช่วยในการดีบักเพิ่มเติมเข้ามาในระบบซึ่งทำให้ระบบมีประสิทธิภาพมากขึ้น เราได้ออกแบบ ICE ระดับบอร์ด หรือบอร์ดอีมูเลชันให้มีความสามารถในการดีบักใกล้เคียงกับอินเซอร์กิตอีมูเลเตอร์ และดีกว่าเครื่องไมโครคอมพิวเตอร์แผ่นวงจรพิมพ์เดี่ยว ซึ่งมีขีดความสามารถค่อนข้างจำกัด ความสามารถในการดีบักของบอร์ดอีมูเลชัน ในระบบพัฒนาไมโครโปรเซสเซอร์ระดับบอร์ดถูกกำหนดไว้ดังนี้

1. ความสามารถในการจัดการเกี่ยวกับหน่วยความจำ คือ สามารถโปรแกรม/เรียกดูหรือแก้ไขค่าในหน่วยความจำได้ทั้ง 64 กิโลไบต์

2. ความสามารถในการจัดการเกี่ยวกับรีจีสเตอร์ คือ สามารถเรียกดู/แก้ไขค่าในรีจีสเตอร์ได้

3. ความสามารถในการแสดงข้อมูลในหน่วยความจำเป็นโปรแกรมภาษาแอสเซมบลี (Unassemble)

4. ความสามารถในการป้อนโปรแกรมภาษาแอสเซมบลีทีละบรรทัด โดยผ่านทางแป้นพิมพ์และจอภาพของเครื่องไมโครคอมพิวเตอร์ IBM PC หรือเทอร์มินัล VT-100 (Line Assemble)

5. ความสามารถในการจัดการเกี่ยวกับพอร์ตไอ/โอ คือ สามารถสั่งให้อ่านหรือเขียนข้อมูลกับพอร์ตไอ/โอได้ ซึ่งเป็นส่วนช่วยในการดีบักฮาร์ดแวร์ด้วย

6. ความสามารถในการสั่งให้ซีพียูทำงานทีละคำสั่ง เพื่อหยุดดูค่าต่าง ๆ (Single Step)

7. ความสามารถในการทำงานตามโปรแกรมผู้ใช้ (Execute User Program) โดยเริ่มต้นจากตำแหน่งหน่วยความจำปัจจุบัน หรือเริ่มต้นจากตำแหน่งที่กำหนดท้ายคำสั่งก็ได้

8. ความสามารถในการกำหนดจุดหยุดปฏิบัติงาน (Breakpoint) คือ สามารถกำหนดจุดหยุดปฏิบัติงานเพื่อตรวจสอบค่าต่าง ๆ หลังจากทำงานตามโปรแกรมผู้ใช้มาช่วงหนึ่งแล้ว และยังสามารถจะสั่งให้ทำงานต่อจากจุดนั้นได้อย่างถูกต้อง

9. ความสามารถในการถ่ายข้อมูลจากเครื่องไมโครคอมพิวเตอร์IBM PC ลงสู่หน่วยความจำของระบบ (Download)

10. ความสามารถในการเก็บข้อมูลหรือโปรแกรมจากหน่วยความจำของระบบ ลงสู่ดิสก์ (Upload)