บทที่ 3

หลักการทำงานของอินเซอร์กิตอีมูเลเตอร์ที่ออกแบบขึ้น

จากแนวคิดและทฤษฎีที่กล่าวถึงแล้วในบทที่ 2 สรุปหลักการที่ใช้ดังนี้

1. ใช้ไมโครโปรเซสเซอร์สำหรับควบคุมแบบแยกจากไมโครโปรเซสเซอร์ที่ใช้อีมูเลชัน

2. การติดต่อกับผู้ใช้ทางฮาร์ดแวร์ ใช้ RS-232C ซึ่งเป็นระบบที่มาตรฐานทำให้เขียนซอฟต์แวร์ในการควบคุมเครื่องคอมพิวเตอร์ได้ง่าย

3. รูปแบบการส่งข้อมูลใช้การติดต่อกับเทอร์มินอลคือรับข้อมูลที่ผู้ใช้พิมพ์มาทั้งหมดวิเคราะห์และส่งสัญญาณออกในรูปแบบที่จะแสดงผลบนจอภาพโดยตรง โปรแกรมทางด้านเครื่องคอมพิวเตอร์ใช้โปรแกรมมาตรฐานในการเลียนแบบเทอร์มินอล นอกจากนี้ยังสามารถรับส่งข้อมูลเป็นกลุ่มสำหรับโปรแกรมที่จะพัฒนาในอนาคตด้วย

4. รูปแบบการติดต่อกับผู้ใช้ เป็นคำสั่งที่คล้ายกับคำสั่งในโปรแกรม debug.com

5. การควบคุมการทำงานของไมโครโปรเซสเซอร์ ใช้สัญญาณ WAIT ร่วมกับการตรวจสอบการเริ่มต้นของคำสั่ง

6. การกำหนดจุดหยุดใช้วงจรเปรียบเทียบแอดเดรสจากซีพียู กับค่าที่ตั้งไว้

7. มีการติดตามการทำงานในเวลาจริง (real time trace)

8. มีหน่วยความจำอีมูเลชันที่สามารถเลือกใช้ได้ด้วยคำสั่ง

<u>ข้อกำหนดรายละเอียดของอินเซอร์กิตอีมูเลเตอร์ที่ออกแบบ</u>

1. ใช้สำหรับไมโครโปรเซสเซอร์ Z-80 และ 8085

2. การเชื่อมต่อกับเครื่องไมโครคอมพิวเตอร์ใช้ RS-232C

3. โปรแกรมที่ใช้บนเครื่องไมโครคอมพิวเตอร์คือโปรแกรมเลียนแบบเทอร์มินอล PROCOMM PLUS 1.1B

4. มีหน่วยความจำอีมูเลชัน ขนาด 64 กิโลไบต์ ซึ่งเป็นขนาดสูงสุดที่ Z-80 และ 8085 ติดต่อได้โดยตรง แบ่งเป็นบล็อกขนาด 2 กิโลไบต์

5. มีคำสั่งใช้งานดังนี้

5.1 คำสั่งที่เกี่ยวกับการกำหนดการใช้หน่วยความจำในช่วงแอดเดรสที่กำหนด ถ้ามีการทำงานผิดจากที่กำหนดไว้จะเกิดการหยุด

กำหนดว่ามีหน่วยความจำหรือไม่

กำหนดว่าใช้จากระบบ (หน่วยความจำอีมูเลชัน) หรือใช้จากระบบเป้าหมาย

กำหนดว่าเป็นที่เก็บโปรแกรมและข้อมูลหรือเก็บเฉพาะข้อมูล

กำหนดว่าเป็นหน่วยความจำอ่านได้อย่างเดียวหรือเขียนได้ด้วย

5.2 คำสั่งเกี่ยวกับหน่วยความจำ

แสดงข้อมูลในหน่วยความจำเป็นบล็อก

ใส่ค่าในหน่วยความจำเป็นบล็อก

เปรียบเทียบข้อมูลในหน่วยความจำ 2 ช่วง

ย้ายข้อมูลจากหน่วยความจำช่วงหนึ่งไปยังอีกช่วงหนึ่ง

ทดสอบอ่านเขียนหน่วยความจำในช่วงที่กำหนดว่าเป็นแรมทั้งหมดหรือไม่

ค้นหาข้อมูลที่กำหนดในช่วงแอดเดรสที่กำหนด

5.3 คำสั่งเกี่ยวกับอินพุตเอาต์พุต

อ่านข้อมูลจากอินพุตที่กำหนด

ส่งข้อมูลไปยังเอาต์พุตที่กำหนด

5.4 การจัดการข้อมูลเป็นภาษาแอสเซมบลี

รับคำสั่งภาษาแอสเซมบลีแปลเป็นภาษาเครื่องใส่ในแอดเดรสที่กำหนด

อ่านข้อมูลภาษาเครื่องจากช่วงแอดเดรสที่กำหนดแปลงเป็นภาษาแอสเซมบลี

5.5 การอ่านเขียนแฟ้มข้อมูลในรูปแบบ Intel hex

อ่านแฟ้มข้อมูลมาเขียนลงในหน่วยความจำ

อ่านข้อมูลในหน่วยความจำเขียนเป็นแฟ้มข้อมูล

5.6 การแสดงและแก้ไขค่ารีจิสเตอร์

5.7 การทำงานในโปรแกรมของผู้ใช้ทีละคำสั่ง

ทำงานทีละคำสั่ง แสดงค่ารีจิสเตอร์หลังจากทำแต่ละคำสั่งจนครบจำนวนที่ระบุ

ทำงานทีละคำสั่ง แสดงเฉพาะคำสั่ง หยุดรอระหว่างคำสั่ง

ทำงานทีละแมชชีนไซเคิล หยุดรอระหว่างแมชชีนไซเคิล

5.8 การกำหนดจุดหยุด

การกำหนดจุดหยุดเป็นแอดเดรสที่ซีพียูอ่านคำสั่งได้ 1 แอดเดรส

กำหนดให้หยุดเมื่อซีพียู HALT

กำหนดให้หยุดเมื่อสัญญาณภายนอกเป็นสถานะตรงที่กำหนด

5.9 การให้ซีพียูทำงานตามเวลาจริงจนกว่าจะพบจุดหยุด

5.10 การติดตามการทำงานของซีพียูในเวลาจริง

ติดตามตั้งแต่ซีพียูเริ่มพบจุดหยุด ให้ซีพียูหยุดเมื่อบัฟเฟอร์เต็ม

ติดตามตั้งแต่ซีพียูเริ่มทำงานจนพบจุดหยุด

ติดตามตั้งแต่ซีพียูเริ่มทำงานจนพบจุดหยุดแล้วติดตามต่ออีก 1024 แมชชีนไซเคิล

แสดงผลการติดตามเป็นแอดเดรส, สถานะ, ข้อมูลและแปลเป็นภาษาแอสเซมบลี

5.11 คำสั่งควบคุมการรับสัญญาณควบคุมซีพียูจากระบบเป้าหมาย

5.12 คำสั่งรีเซตซีพียู

5.13 คำสั่งเรียกดูตารางคำสั่งทั้งหมด

การกำหนดรายละเอียดที่กล่าวมามีเหตุผลที่อธิบายในเรื่องแนวความคิดและทฤษฎีแล้ว และมีเหตุผลสำหรับการเลือกรายละเอียดบางข้ออีกดังนี้

การจัดการเลือกใช้หน่วยความจำเป็นแอดเดรสช่วงละ 2 กิโลไบต์ ทำให้ผู้ใช้สามารถกำหนดการใช้หน่วยความจำได้ตามที่เป็นจริง เพราะปัจจุบันขนาดเล็กสุดของหน่วยความจำที่มีผู้นิยมใช้ คือขนาด 2 กิโลไบต์

การกำหนดว่ามีหรือไม่มีหน่วยความจำในช่วงที่กำหนด ทำให้เวลาโปรแกรมทำงานผิดพลาด มีการอ่านเขียนหน่วยความจำจากบริเวณนั้น โปรแกรมจะหยุดทำงานซึ่งมีประโยชน์เหมือนการต้องการสัญญาณตอบรับของซีพียูที่ใช้ระบบ Asynchronous Buses ที่กล่าวถึงในบทที่ 2

การกำหนดว่าหน่วยความจำเป็นชนิดอ่านอย่างเดียวได้ ทำให้หน่วยความจำที่เก็บโปรแกรม หรือข้อมูลที่สำคัญไม่ถูกทำลายจากโปรแกรมที่ผิดพลาด แต่ยังไม่สามารถเขียนข้อมูลด้วยคำสั่งของอินเซอร์กิตอีมูเลเตอร์ได้ตามปรกติ

การกำหนดว่าเป็นหน่วยความจำที่เก็บเฉพาะข้อมูลไม่ใช่เก็บโปรแกรม ทำให้ข้อผิดพลาดที่เกิดจากซีพียูทำงานบริเวณที่ไม่ใช่เก็บโปรแกรมจะถูกพบได้โดยเร็วก่อนที่จะเกิดความเสียหายมากขึ้น

การกำหนดจุดหยุดได้จุดเดียวทำให้การใช้งานทั่วไป ทำได้ง่ายไม่สับสน โดยเพิ่มการกำหนดให้หยุดได้เมื่อโปรแกรม HALT หรือจากสัญญาณภายนอกที่สามารถประยุกต์ใช้งานได้กรณีที่ต้องการทำงานซับซ้อน

การมีความสามารถในการทดสอบหน่วยความจำแรม นอกจากจะอ่านเขียนได้แล้วยังสามารถตรวจสอบซ่อมปัญหาที่เกิดจากสายแอดเดรสและข้อมูลขาดหรือซ้ำซ้อนได้ด้วย

นอกจากคำสั่งที่แสดงไว้แล้วอินเซอร์กิตอิมูเลเตอร์ ยังต้องมีการรับส่งข้อมูลเป็นกลุ่มในกรณีที่มีโปรแกรมจัดการข้อมูลบนเครื่องคอมพิวเตอร์แล้ว โดยเราให้ลักษณะของข้อมูลมีส่วนตรวจสอบข้อผิดพลาด ซึ่งจะเลือกการใช้งานคล้ายรูปแบบของแฟ้มข้อมูล Intel hex แต่การรับส่งจะทำทีละบรรทัด เพื่อให้เข้ากันได้กับระบบการรับคำสั่งจากเทอร์มินอลที่เป็นโปรแกรมปรกติ

การออกแบบระบบเป็นระดับบล็อก

1. การเลือกใช้ไมโครโปรเซสเซอร์

จากหลักการและข้อกำหนดของระบบที่กล่าวมาแล้ว เราจะนำมาใช้ในการออกแบบในส่วนของฮาร์ดแวร์และซอฟต์แวร์ร่วมกัน โดยในเครื่องนี้จะประกอบด้วยซีพียู 2 ตัว ทำหน้าที่แยกกันโดยมีส่วนหนึ่งเป็นระบบไมโครโปรเซสเซอร์ปรกติเรียกว่าคอนโทรลซีพียู มีพอร์ต RS-232C สำหรับติดต่อภายนอก และมีอุปกรณ์อินพุตเอาต์พุต สำหรับควบคุมการทำงานของซีพียูอีกตัวหนึ่ง การเลือกไมโครโปรเซสเซอร์ที่ทำหน้าที่คอนโทรลซีพียูจึงเลือกที่มีระบบพัฒนาไมโครโปรเซสเซอร์ครบสมบูรณ์ที่สุด เพราะการเขียนซอฟต์แวร์จะเขียนให้ใช้กับซีพียูตัวนี้ จึงเลือกใช้ 8085 เพราะในห้องปฏิบัติการมีอินเซอร์กิตอิมูเลเตอร์ที่ใช้งานได้หลายเครื่อง เพื่อเปลี่ยนขณะเกิดปัญหาในการใช้ และยังมีซอฟต์แวร์ภาษาระดับสูง คือ ภาษา PL/M(Caine, Farber & Gordon, Inc., 1986) จากการทดลองสามารถใช้งานได้สะดวก เขียนโปรแกรมได้ง่ายขึ้น ขนาดโปรแกรมไม่ใหญ่มาก และเหมาะสมกับการใช้งานที่ไม่ต้องมีการคำนวณแบบตัวเลขทศนิยม จึงเข้ากับงานวิจัยนี้ได้ดี

2. โครงสร้างทางฮาร์ดแวร์

การออกแบบในขั้นต้นจะได้โครงสร้างทางฮาร์ดแวร์แสดงในรูป 3.1

โดยในเครื่องนี้จะประกอบด้วยซีพียู 2 ตัว คือ คอนโทรลซีพียู 8085 และอีมูเลชันซีพียู Z-80 หรือ 8085

ระบบของคอนโทรลซีพียูประกอบด้วยหน่วยความจำรอม, แรม, วงจรตั้งเวลา พอร์ต RS-232C สำหรับติดต่อภายนอก และมีอุปกรณ์อินพุตเอาต์พุตสำหรับควบคุมการทำงานของอีมูเลชันซีพียู

ระบบของอีมูเลชันซีพียู มีส่วนประกอบสำคัญ คือ

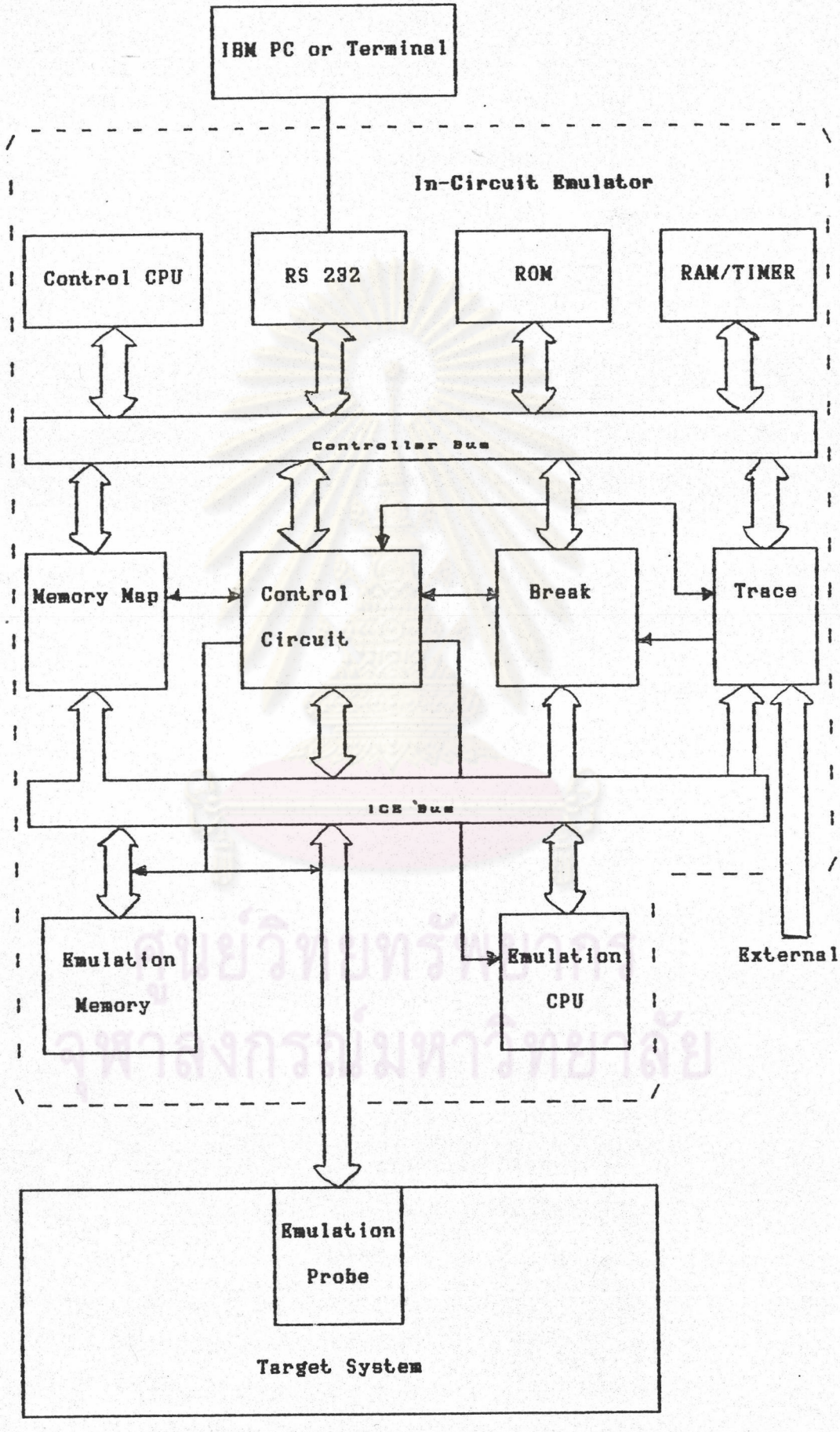
หน่วยความจำอีมูเลชัน (Emulation Memory) ขนาด 64 กิโลไบต์สำหรับผู้ใช้กำหนดให้เป็นรอมหรือแรมได้ช่วงละ 2 กิโลไบต์ มีการต่อบัฟเฟอร์สำหรับบัสข้อมูลเพื่อป้องกันข้อมูลจากหน่วยความจำอีมูเลชันออกไปรบกวนกับสัญญาณภายนอก สัญญาณควบคุมการทำงานของบัฟเฟอร์นี้จะเกิดจากอุปกรณ์ที่ควบคุมการทำงานของหน่วยความจำอีมูเลชัน คือวงจรเลือกหน่วยความจำ

วงจรเลือกหน่วยความจำ (Memory Map) มีการรับแอดเดรสจากบัสของอีมูเลชันซีพียูมาใช้ถอดรหัสเลือกหน่วยความจำ โดยโปรแกรมค่าได้จากคอนโทรลซีพียู จึงมีลักษณะเป็นพอร์ตอินพุตเอาต์พุตของระบบคอนโทรลซีพียู และ เอาต์พุตที่ได้นอกจากจะควบคุมการใช้หน่วยความจำแล้วยังส่งไปกำหนดการหยุดซีพียูด้วย

วงจรที่ใช้กำหนดจุดหยุดจะมีลักษณะเป็นวงจรเปรียบเทียบ ที่ด้านหนึ่งต่อกับแอดเดรสบัสของอีมูเลชันซีพียู อีกด้านหนึ่งเป็นพอร์ตเอาต์พุตของคอนโทรลซีพียูเพื่อรับค่าที่ผู้ใช้กำหนด นอกจากนี้ยังรับสัญญาณจากวงจรอื่นที่สามารถหยุดการทำงานของซีพียูได้แล้ววงจรจะสร้างสัญญาณส่งไปควบคุมซีพียู

Trace คือ วงจรติดตามการทำงานของซีพียูในเวลาจริง ประกอบด้วยหน่วยความจำที่มีส่วนของข้อมูลติดต่อกับบัสของอีมูเลชันซีพียูและต่อกับสัญญาณเข้าจากภายนอกด้วยและมีส่วนแอดเดรสต่อกับวงจรนับที่โปรแกรมได้จากคอนโทรลซีพียู สัญญาณที่ได้จากบัฟเฟอร์เต็มจะส่งไปหยุดซีพียูด้วย

Control Circuit คือ วงจรควบคุมการทำงานของอีมูเลชันซีพียูจากพอร์ตเอาต์พุตของคอนโทรลซีพียูและมีพอร์ตอินพุตตรวจสอบสถานะของอีมูเลชันซีพียู สัญญาณจากวงจรกำหนดจุดหยุดร่วมกับสัญญาณควบคุมจากคอนโทรลซีพียูจะใช้สร้างสัญญาณ WAIT ในการควบคุมอีมูเลชันซีพียู



รูป 3.1 โครงสร้างทางฮาร์ดแวร์

ในการออกแบบครั้งนี้จะมีส่วนที่แตกต่างกันระหว่างอิมูเลชันซีพียู Z-80 กับอิมูเลชันซีพียู 8085 อยู่ในด้านระดับและช่วงเวลาของสัญญาณ ในการออกแบบฮาร์ดแวร์เราจะใช้ PAL (Programmable Array Logic) เป็นหลัก ซึ่งสามารถรับสัญญาณอินพุตและกลับระดับสัญญาณอินพุตได้ ในส่วนสัญญาณควบคุมที่สร้างจากคอนโทรลซีพียูเราจึงสามารถคงระดับและหน้าที่ของสัญญาณไว้ตามเดิม เพื่อให้ซอฟต์แวร์ในส่วนควบคุมในจุดที่สำคัญมีลักษณะเหมือนกันทั้ง 2 รุ่น

3. <u>โครงสร้างทางซอฟต์แวร์</u> ประกอบด้วยข้อกำหนดรายละเอียดดังนี้

3.1 โปรแกรมจะรับข้อมูลจากผู้ใช้ที่พิมพ์คำสั่งทางแป้นพิมพ์ของเครื่องไมโครคอมพิวเตอร์แล้วรับเข้ามาในอินเซอร์กิตอิมูเลเตอร์ผ่านทาง RS-232C

3.2 โปรแกรมจะตอบรับตัวอักษรทุกตัวที่ผู้ใช้พิมพ์เข้ามากลับไปแสดงทางจอภาพ หลังจากผู้ใช้พิมพ์จบคำสั่งแล้ว จะส่งผลการทำงานไปที่จอภาพ จนเสร็จคำสั่งจึงรอรับคำสั่งใหม่

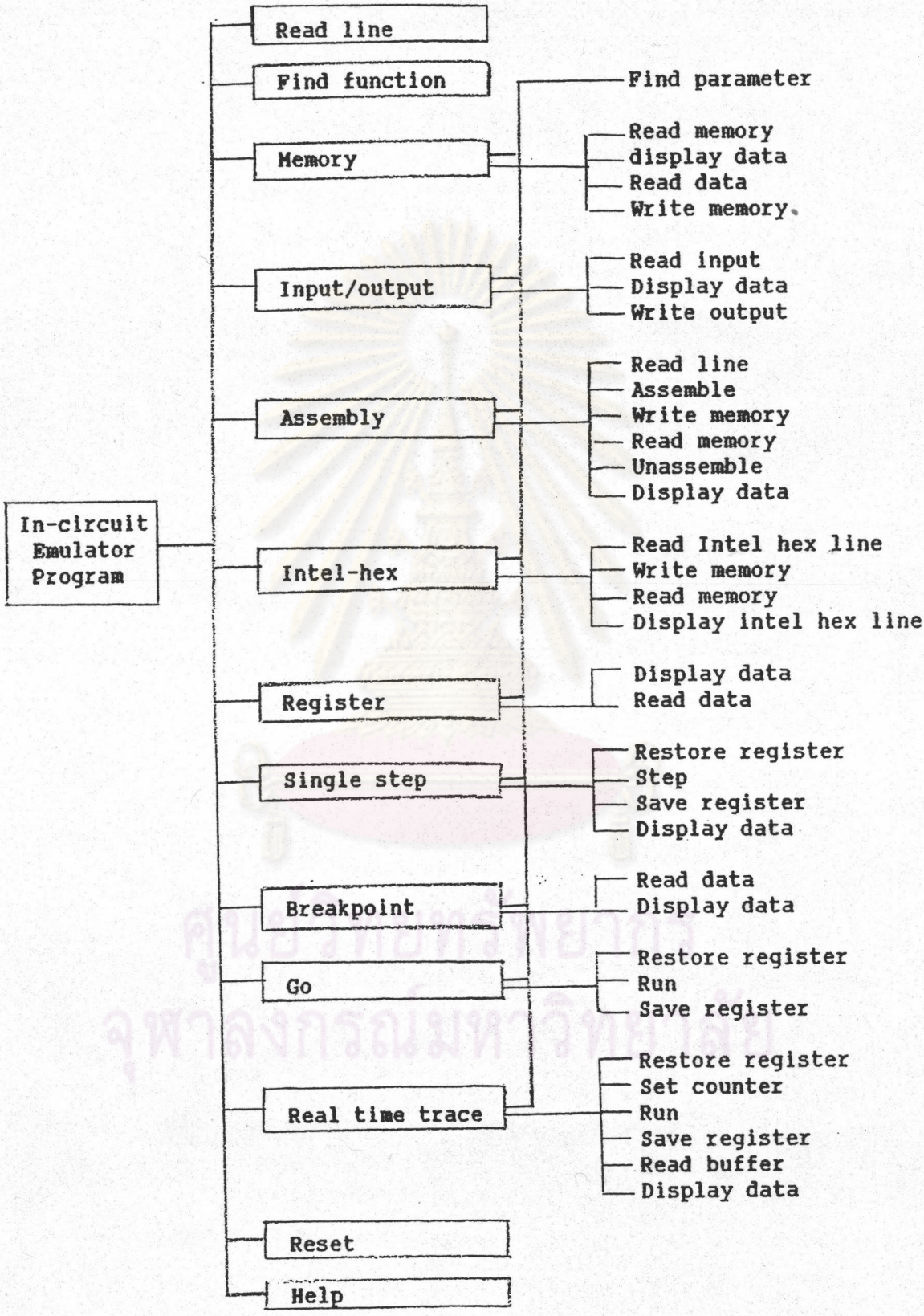
3.3 กระบวนการจัดข้อมูลจะประกอบด้วยการวิเคราะห์คำสั่งและการทำงานตามคำสั่งนั้นๆ การทำงานของแต่ละคำสั่งจะมีส่วนที่ควบคุมการทำงานของฮาร์ดแวร์และส่วนแสดงผลหรือรับข้อมูลเพิ่มเติม

เมื่อดูจากคำสั่งที่ใช้ทั้งหมด เราสามารถเขียนโครงสร้างของซอฟต์แวร์ได้ดังรูป 3.2

<u>เครื่องมือที่ใช้ในงานวิจัย</u>

งานวิจัยนี้เป็นการพัฒนาระบบไมโครโปรเซสเซอร์จึงต้องการเครื่องมือเป็นระบบพัฒนาไมโครโปรเซสเซอร์ ซึ่งประกอบด้วยอุปกรณ์ดังนี้

1. IBM PC หรือเครื่องคอมพิวเตอร์แบบที่เข้ากันได้
2. อินเซอร์กิตอิมูเลเตอร์ สำหรับ 8085 (ESA ICE-I)
3. โปรแกรมบน IBM PC สำหรับติดต่ออินเซอร์กิตอิมูเลเตอร์ (ice85.exe)
4. โปรแกรมแปลภาษา PL/M สำหรับ 8085 (80ds.lib)
5. โปรแกรมเลียนแบบเทอร์มินอลบน IBM (PROCOMM PLUS)
6. โปรแกรมสำหรับออกแบบวงจรด้วย PAL (PALASM2)



รูป 3.2 โครงสร้างทางซอฟต์แวร์

ขั้นตอนการทำงาน

1. สร้างเครื่องต้นแบบที่ประกอบด้วยซีพียู 8085 หน่วยความจำ ตัวจับเวลา และ RS-232C (ในที่นี้ใช้อินเซอร์กิตอีมูเลเตอร์ที่ออกแบบโดย เศรษฐา พันธ์เพ็ง (2532) เป็นเครื่องต้นแบบเพราะมีวงจรอินเซอร์กิตอีมูเลเตอร์ที่ใช้หลักการ WAIT ในการควบคุมอีมูเลชันซีพียู Z-80 โดยมีคอนโทรลซีพียูเป็น 8085)

2. ทดลองเขียนโปรแกรมด้วยภาษา PL/M ให้เครื่องต้นแบบสามารถติดต่อกับ IBM ให้มีการรับและวิเคราะห์คำสั่งได้ การแก้จุดบกพร่องอาศัยอินเซอร์กิตอีมูเลเตอร์สำหรับ 8085 ESA ICE-I (Electro Systems Associates Pvt Ltd)

3. ทดลองเขียนโปรแกรมควบคุมการทำงานของวงจร WAIT ที่ใช้ในเครื่องต้นแบบ ศึกษาการหาจุดเริ่มต้นคำสั่งของ Z-80 จากการทดลองใช้งาน นำหลักการทั้ง 2 มาใช้ออกแบบวงจรที่ทำการวิจัย

4. ออกแบบวงจรอินเซอร์กิตอีมูเลเตอร์สำหรับ Z-80 แล้วสร้างบนแผ่นวงจรพิมพ์อเนกประสงค์ ทดลองเขียนโปรแกรมควบคุมให้สามารถทำงานและกำหนดจุดหยุดได้

5. ออกแบบวงจรส่วนติดตามการทำงานในเวลาจริง แล้วออกแบบแผ่นวงจรพิมพ์สำหรับวงจรทั้งหมด

6. ออกแบบวงจรอินเซอร์กิตอีมูเลเตอร์ สำหรับ 8085 แล้วออกแบบแผ่นวงจรพิมพ์