บทที่ 5

การทำงานของโปรแกรม

การใช้โปรแกรม PROCOMM PLUS ได้แสดงในเรื่องคู่มือการใช้งานแล้ว ในบทนี้จะอธิบายการทำงานของโปรแกรมบนอินเซอร์กิตอีมูเลเตอร์ ได้แก่

การทำงานของโปรแกรมหลัก

โปรแกรมหลักทำหน้าที่เรียกใช้โปรแกรมย่อยเพื่อทำงานดังนี้

1. เมื่อเริ่มเปิดเครื่องหรือกดปุ่มรีเซต

   โปรแกรมจะตั้งค่าตัวแปรและกำหนดค่าเริ่มต้นให้ฮาร์ดแวร์

2. รับคำสั่งจากผู้ใช้ 1 บรรทัด

   ผู้ใช้นิมน์คำสั่งทางแป้นนิมน์ของเครื่องไมโครคอมพิวเตอร์ผ่านทาง RS-232C โปรแกรมจะตอบรับตัวอักษรทุกตัวที่ผู้ใช้นิมน์เข้ามากลับไปแสดงทางจอภาพ และเก็บข้อมูลไว้ในบัฟเฟอร์ คำสั่งต่างๆจะจบเมื่อกด Enter แล้ว

3. วิเคราะห์คำสั่ง

   หลังจากผู้ใช้นิมน์จบคำสั่งแล้วโปรแกรมจะวิเคราะห์คำสั่งโดยดูตัวอักษรตัวแรก

4. การทำงานตามคำสั่ง

   การทำงานของแต่ละคำสั่งจะมีส่วนที่ควบคุมการทำงานของฮาร์ดแวร์และส่วนแสดงผลหรือรับข้อมูลเพิ่มเติม จนเสร็จคำสั่งจึงกลับไปรอรับคำสั่งใหม่

ข้อกำหนดในการรับส่งข้อมูลกับโปรแกรม PROCOMM PLUS

การใช้ PROCOMM PLUS ได้แสดงในเรื่องคู่มือการใช้งานแล้ว ในที่นี้จะอธิบายความหมายของการติดตั้งระบบที่เกี่ยวกับซอฟต์แวร์ บนอินเซอร์กิตอีมูเลเตอร์ ได้แก่

1. การกำหนดรูปแบบอัตราการรับข้อมูล

   วงจรส่วนนี้ประกอบด้วย ไอซี 8251 ที่สามารถโปรแกรมรูปแบบการรับส่งข้อมูลได้หลายแบบแต่ในที่นี้จะกำหนดเพียงรูปแบบเดียวเพื่อไม่ให้ผู้ใช้สับสน และอัตราการรับส่งข้อมูลที่

วงจรออกแบบไว้ให้สามารถโปรแกรมได้ถูกต้องเป็นค่าตายตัวเช่นเดียวกันรูปแบบและอัตราการรับส่งข้อมูลที่ใช้ในอินเซอร์กิตอีมูเลเตอร์ คือ

BAUD RATE 9600

PARITY NONE

DATA BITS 8

STOP BITS 1

ค่าที่กำหนดนี้เป็นค่าที่นิยมใช้งานทั่วไปสำหรับซอฟต์แวร์บนเครื่องไมโครคอมพิวเตอร์ที่ติดต่อกับระบบพัฒนาไมโครโปรเซสเซอร์ที่มีใช้ในปัจจุบัน เช่น คอมพิวเตอร์แผ่นพิมพ์เดี่ยวที่ใช้ 8052 AH-BASIC สามารถทำงานกับโปรแกรมที่มีรูปแบบและอัตราการรับส่งขนาดนี้ได้พอดี ทำให้ผู้ใช้งานสามารถทำเครื่องไมโครคอมพิวเตอร์ให้เป็นระบบพัฒนาไมโครโปรเซสเซอร์สำหรับระบบต่างกันได้โดยไม่ยุ่งยาก

2. การกำหนดควบคุมการรับส่งสัญญาณ

แม้ว่าการรับส่งข้อมูลแบบอนุกรมจะมีความเร็วต่ำแต่เมื่อเปรียบเทียบกับความสามารถของอุปกรณ์บางอย่าง เช่น จอภาพ หรือฟลอปปีดิสค์ ซึ่งมีจังหวะที่ต้องรอเป็นเวลานาน ถ้าเราไม่มีการควบคุมการรับส่งข้อมูล ทางพอร์ตอนุกรมจะมีข้อมูลสูญหายไปในระหว่างที่หยุดรอ วิธีการแก้ไขคือต้องตกลงวิธีการส่งสัญญาณควบคุมระหว่างโปรแกรมบนไมโครคอมพิวเตอร์ที่เป็นเทอร์มินอลกับเครื่องอินเซอร์กิตอีมูเลเตอร์ว่าในช่วงที่ไม่สามารถรับข้อมูลได้จะต้องส่งสัญญาณให้อีกฝ่ายหนึ่งหยุดส่งข้อมูล มาตรฐานที่ใช้กันในระบบทั่วไปมี 2 วิธีคือ การควบคุมด้วยฮาร์ดแวร์และการควบคุมด้วยซอฟต์แวร์

การควบคุมด้วยฮาร์ดแวร์ต้องการสายรับส่งสัญญาณเพิ่มเติมจากสัญญาณข้อมูลแบบอนุกรม ที่เดิมจำเป็นต้องใช้เพียงสามเส้นคือ สายสัญญาณส่งข้อมูล รับข้อมูล และสายดินสัญญาณเพิ่มเติมจะมีไว้ควบคุมการส่งข้อมูลของอุปกรณ์ที่ติดต่อด้วยว่าในช่วงที่ไม่มีสัญญาณอนุญาตให้ส่งข้อมูลมาไม่ได้ สำหรับไอซี 8251 มีการจัดสัญญาณเหล่านี้ไว้เรียบร้อย เพียงต่อสายให้ถูกต้องตามที่กำหนดก็จะควบคุมการส่งข้อมูลจากอินเซอร์กิตอีมูเลเตอร์ไม่ให้ไปยังเทอร์มินอลได้แต่ในการใช้งานทั่วไปกลับนิยมใช้การต่อสัญญาณควบคุมเหล่านี้ป้อนกลับให้ตัวเองทำให้ส่งข้อมูลออกมาได้ตลอดเวลา เพราะในโปรแกรมติดต่อสื่อสารที่ทำงานบนคอมพิวเตอร์บางโปรแกรมไม่ยอมทำงานต่อ เมื่อไม่มีสัญญาณตอบรับจากพอร์ตสื่อสาร เช่น PROCOMM PLUS เมื่อตั้งการเลือกใช้การควบคุมแบบฮาร์ดแวร์(RTS/CTS) เวลาเรียกใช้โปรแกรมจากดอส โปรแกรมจะหยุดทำงานและกลับไปดอสไม่ได้ จนกว่าจะต่อพอร์ตสื่อสารให้มีสัญญาณเหล่านี้ครบ การเลือกแบบนี้จึงมี

โอกาสทำให้เครื่องใช้งานปกติไม่ได้ต้องเปิดเครื่องใหม่ ไม่สะดวกและอาจมีข้อมูลสูญหายได้ง่าย

การควบคุมแบบซอฟต์แวร์ มีข้อจำกัดตรงที่ผู้พัฒนาซอฟต์แวร์ต้องเขียนโปรแกรมเพิ่มเติมเพื่อรับรู้ข้อมูลที่ตกลงกันว่าเป็นตัวควบคุมอุปกรณ์ ข้อมูลที่นิยมใช้กันทั่วไปคือตัวอักษรในรหัสแอสกีเป็น 13 หรือ Ctrl-S จากแป้นพิมพ์ ในการนำมาใช้งานเรียกชื่อว่า XOFF ทำหน้าที่เป็นตัวหยุดคือในการสื่อสารระหว่างอุปกรณ์ 2 ตัว ถ้าตัวรับไม่สามารถจะรับข้อมูลได้ทันทีตัวส่งก็จะใช้รหัส XOFF นี้เป็นสัญญาณบอกตัวส่งให้หยุดชั่วขณะ สิ่งที่เราพบบ่อยๆคือ ถ้าการแสดงผลทางจอภาพเร็วมากจนอ่านไม่ทันเราจะกดปุ่ม Ctrl-S จากแป้นพิมพ์ เพื่อหยุดการแสดงไว้ชั่วคราว ในโปรแกรม PROCOMM PLUS การกด Ctrl-S จะส่งรหัสนี้ไปยังอินเซอร์กิตอีมูเลเตอร์โดยตรง เพื่อสั่งให้หยุดส่งข้อมูลมา การหยุดจอภาพควรใช้ปุ่ม Ctrl-Numlock หรือ Pause แทนจะหยุดจอภาพพร้อมกับส่งสัญญาณ XOFF ไปด้วย

สำหรับรหัสควบคุมที่ใช้คู่กับ XOFF คือ XON ซึ่งเป็นรหัสแอสกีหมายเลข 11 หรือ Ctrl-Q บนแป้นพิมพ์ ในการใช้ปุ่ม Pause จะมีการส่งสัญญาณนี้ออกมาเองเมื่อกดปุ่มใดๆ

จากการพิจารณาในเหตุผลข้างต้นเราจึงต้องเขียนโปรแกรมในการส่งข้อมูลไปยังเครื่องคอมพิวเตอร์ (ซึ่งก็คือการส่งไปยังจอภาพในการใช้งานปกติ และการส่งไปยังดิสค์ในการสร้างแฟ้มข้อมูล) ให้มีการตรวจสอบสัญญาณ XOFF ก่อนจะส่งข้อมูลออกไป และถ้าพบ XOFF ต้องหยุดส่งข้อมูลแล้วรอรับสัญญาณจนกว่าจะพบ XON จึงเข้าสู่การทำงานปกติของการส่งได้

ในทางตรงข้ามจะพิจารณาว่าโปรแกรมในการรับข้อมูลจำเป็นต้องควบคุมการส่งสัญญาณของเครื่องคอมพิวเตอร์หรือไม่ พบว่าในการใช้งานปกติข้อมูลที่คอมพิวเตอร์ส่งมาคือข้อมูลที่ผู้ใช้พิมพ์ผ่านเข้าแป้นพิมพ์ที่มีความเร็วต่ำและโปรแกรมจะจัดการข้อมูลได้เร็วมากจึงไม่น่าต้องใช้วิธีนี้ หลังจากการพัฒนาโปรแกรมมาถึงส่วนที่ต้องรับข้อมูลที่คอมพิวเตอร์อ่านจากดิสค์ส่งมาก็ยังสามารถจัดการข้อมูลได้เร็วพอโดยทดลองใช้เครื่องคอมพิวเตอร์ 386-33 อ่านข้อมูลจากแรมดิสค์ส่งมาด้วยอัตรา 19200 บิตต่อวินาที โปรแกรมที่รับข้อมูลจากดิสค์ใส่หน่วยความจำของอีมูเลชันซีพียูยังคงทำงานได้โดยไม่ผิดพลาด จึงไม่จำเป็นต้องใช้การควบคุมในส่วนนี้

3. ในการใช้งานร่วมกับดิสค์เพื่อให้สามารถรับส่งแฟ้มข้อมูลแบบ Intel hex ได้จะต้องกำหนดตัวเลือกรูปแบบของแฟ้มข้อมูลที่จะส่ง ในที่นี้ Intel hex เป็นแฟ้มข้อมูลแบบแอสกี คือมีเฉพาะตัวอักษรที่อ่านได้ร่วมกับรหัสควบคุมตำแหน่งแสดงผล การรับส่งข้อมูลจึงสามารถทำงานร่วมกับการใช้งานเทอร์มินอลธรรมดาที่ใช้ จอภาพ และแป้นพิมพ์ ได้ง่ายเราจึงเลือกใช้รูปแบบการรับส่งแฟ้มข้อมูลเป็นแบบแอสกี

ในการพิจารณารับส่งแฟ้มข้อมูลสิ่งที่สำคัญกว่าความเร็วในการรับส่งคือความถูกต้องของข้อมูล โดยเฉพาะข้อมูลที่สำคัญ คือการส่งแฟ้มข้อมูลภาษาเครื่อง สำหรับไมโครโปรเซสเซอร์มาทดลองทำงานบนอินเซอร์กิตอีมูเลเตอร์ควรจะมีการตรวจสอบข้อผิดพลาดด้วย ในโปรแกรม PROCOMM PLUS มีตัวเลือกเกี่ยวกับการส่งแฟ้มข้อมูลแบบแอสกี ที่สำคัญคือ pace character (ดูจากรูป 8.4) ในการส่งข้อมูลจากแฟ้มข้อแอสกีซึ่งแบ่งเป็นบรรทัด เมื่อส่งข้อมูลจบบรรทัดแล้ว จะรอรับตัวอักษรตัวนี้ให้ได้ก่อนจึงส่งข้อมูลบรรทัดต่อไป ปกติถ้าไม่มีการตกลงกันจะไม่ใช้ความสามารถนี้โดยใส่เป็นรหัสแอสกีหมายเลข 0 แล้วเพิ่มเวลารอระหว่างตัวอักษรและระหว่างบรรทัดแทน ซึ่งจะลดข้อผิดพลาดที่เกิดจากความเร็วของอุปกรณ์ได้ แต่ความผิดพลาดด้านต่างๆที่อาจเกิดขึ้นได้จะไม่ได้รับการจัดการในรหัสแอสกี จึงรหัสควบคุมการสื่อสารอีกกลุ่มหนึ่งเพื่อใช้ตอบรับความถูกต้องในการสื่อสาร (ต่างกับ XON/XOFF ที่ควบคุมด้านความเร็ว) คือ ACK (Acknowledge,06,Ctrl-F)ใช้สำหรับผู้รับส่งกลับไปให้ผู้ส่งเมื่อได้รับข้อมูลที่ถูกต้อง และ NAK (Negative Acknowlegde, 15,(Ctrl-U) เป็นรหัสที่ผู้รับส่งให้กับผู้ส่ง เพื่อแสดงว่าได้รับข้อมูลที่ไม่ถูกต้องมา

ในแฟ้มข้อมูลแบบ Intel hex มีการตรวจสอบผลรวมของข้อมูลในแต่ละบรรทัด จึงใช้ความสามารถนี้ร่วมกับการส่งสัญญาณ ACK ได้แต่ PROCOMM PLUS มีการตรวจสอบได้เฉพาะสัญญาณตอบรับว่าถูกต้องเท่านั้น เมื่อเกิดความผิดพลากจึงได้เพียงหยุดรอรับ ACK เท่านั้น แม้จะส่ง NAK ออกไปก็ไม่สามารถทำให้ PROCOMM PLUS จัดการข้อผิดพลาดแบบอัตโนมัติได้ ต้องให้ผู้มาจัดการต่อ

ในการใช้ ACK นี้ นอกจากประโยชน์ในการตรวจสอบความถูกต้องของการส่ง Intel hex แล้ว การพิมพ์คำสั่งเข้าจากแป้นพิมพ์ก็มีรูปแบบเดียวกัน เราจึงสามารถเขียนคำสั่งที่จะใช้อินเซอร์กิตอีมูเลเตอร์ไว้เป็นแฟ้มข้อมูลได้แล้วส่งแฟ้มข้อมูลนี้ไปให้อินเซอร์กิตอีมูเลเตอร์ทีละบรรทัด ถ้าคำสั่งทำงานได้ถูกต้องหรือทำงานเสร็จแล้วพร้อมรับข้อมูลบรรทัดต่อไปจึงส่ง ACK ออกมาซึ่งจะเหมือนกับเครื่องหมาย prompt ที่ผู้ใช้ต้องรอก่อนจะพิมพ์คำสั่งใหม่ได้ สำหรับกรณีที่ผู้ใช้ใส่ข้อมูลผิดพลาดแล้วมีข้อความบอกความผิดพลาดเครื่องจะส่ง NAK ออกมาแทน เป็นการหยุดการส่งคำสั่งจากแฟ้มข้อมูลรอให้ผู้ใช้มาจัดการ

ในการส่งข้อมูลมาเก็บเป็นแฟ้มข้อมูลที่คอมพิวเตอร์ ในรูปแบบ Intel hex ซึ่งมีการส่งข้อมูลตรวจสอบข้อผิดพลาดในแต่ละบรรทัดมาด้วย แต่ PROCOMM PLUS ไม่มีความสามารถในการจัดการส่วนนี้เราจึงส่งข้อมูลมาเก็บไว้โดยไม่มีการตรวจสอบก่อน แต่ข้อมูลจะถูกตรวจสอบอีกครั้งในเวลาที่แฟ้มข้อมูลนี้ถูกส่งกลับมาให้อินเซอร์กิตอีมูเลเตอร์

การทำงานของโปรแกรมติดต่อสื่อสารกับอินเซอร์กิตอีมูเลเตอร์

วิธีการที่นิยมในการควบคุมพอร์ตสื่อสารมี 2 วิธีคือ การใช้สัญญาณอินเตอร์รัพต์เพื่อให้โปรแกรมสื่อสารเป็นงานในส่วนโปรแกรมตอบรับอินเตอร์รัพต์ กับอีกวิธีหนึ่งคือการรับส่งข้อมูลแบบ polling จะติดต่อกับโปรแกรมสื่อสารในเวลาที่เตรียมไว้สำหรับงานนี้โดยวนตรวจสอบความพร้อมของไอซีที่ใช้ควบคุมพอร์ตสื่อสาร

วิธีแรกมีความยุ่งยากในการเขียนโปรแกรมและการแก้จุดบกพร่องมาก แต่สามารถรับข้อมูลได้ต่อเนื่องครบถ้วนกว่า

ในโปรแกรมของเราเลือกใช้วิธี Polling ด้วยเหตุผลของความง่ายในการพัฒนาและผลเสียที่พบเป็นสิ่งที่ยอมรับได้ในการทำงานในระบบนี้ เพราะช่วงเวลาในระหว่างที่ไม่รับคำสั่งเป็นช่วงเวลาไม่จำเป็นต้องรับการควบคุมจากผู้ใช้ แต่กรณีที่จำเป็นเราก็สามารถใช้วิธี Polling จัดการได้เช่นกัน

การทำงานของโปรแกรมรับข้อมูลรับข้อมูล 1 ตัวอักษร

1. อ่านสถานะของ 8251

   จนสถานะแสดงว่ารับข้อมูลได้

2. ส่งข้อมูลกลับไปยังโปรแกรมที่เรียกใช้

การทำงานของโปรแกรมรับข้อมูลรับข้อมูล 1 ตัวอักษรจากโปรแกรมที่เรียกใช้

1. อ่านสถานะของ 8251
2. ถ้าสถานะแสดงว่ารับข้อมูลได้ ดูว่าข้อมูลเป็น XOFF หรือไม่
3. ถ้าเป็น XOFF อ่านสถานะของ 8251

   จนสถานะแสดงว่ารับข้อมูลได้ และ อ่านข้อมูลได้ XON
4. กลับไปทำข้อ 1 จนสถานะของ 8251 บอกว่าพร้อมส่งข้อมูล
5. ส่งข้อมูลให้ 8251

รูปแบบคำสั่งใช้งาน

คำสั่งต่างๆมีรูปแบบดังนี้

คำสั่งเรียกดูรูปแบบคำสั่งทั้งหมด

Help	H

คำสั่งเกี่ยวกับการกำหนดการใช้หน่วยความจำ

Memory Map	MM [range [T|S][P|D][RW|RO]|[N]]

คำสั่งเกี่ยวกับหน่วยความจำ

Dump	D [address|range]
Enter	E address [list]
Fill	F range list
Compare	C range address
Move	M range address
Search	S range list

คำสั่งเกี่ยวกับอินพุตเอาต์พุต

Input	I address
Output	O address byte

คำสั่งเกี่ยวกับการจัดการข้อมูลเป็นภาษาแอสเซมบลี

Assemble	A [address]
Unassemble	U [address|range]

คำสั่งเกี่ยวกับการจัดการข้อมูลแบบ Intel hex

Load	L
Write	W range
display user memory	J Intel hex record
display control memory	Y Intel hex record

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

enter user memory K Intel hex record

enter control memory Z Intel hex record

คำสั่งควบคุมการรับสัญญาณควบคุมซีพียูจากระบบเป้าหมาย

Pin P

คำสั่งรีเซตซีพียู

reset X

คำสั่งเกี่ยวกับการแสดงและแก้ไขค่ารีจิสเตอร์

Register R [register]

คำสั่งเกี่ยวกับการทำงานในโปรแกรมของผู้ใช้ทีละคำสั่ง

Trace T [value|I|M]

คำสั่งเกี่ยวกับการกำหนดจุดหยุด

Breakpoint B [P[S address [value]|C]]|[H[S|C]]|[X[H|L|C]]

คำสั่งเกี่ยวกับการให้ซีพียูทำงานในเวลาจริงจนกว่าจะพบจุดหยุด

Go G [address]

คำสั่งเกี่ยวกับการติดตามการทำงานของซีพียูในเวลาจริง

Trace Realtime TR B|C|F

Trace History TH [range[range[F][R][W][I][O][A]]]

ในการใช้งานให้พิมพ์อักษรตัวย่อคำสั่งตามด้วยพารามิเตอร์ต่างๆโดยมีสัญลักษณ์ของพารามิเตอร์ดังนี้

[ ] สิ่งที่อยู่ในวงเล็บนี้เป็นตัวเลือกที่จะใช้หรือไม่ใช้ก็ได้

| สิ่งที่อยู่หน้าหรือหลังเครื่องหมายนี้ให้เลือกตัวใดตัวหนึ่ง

address ตัวเลขฐานสิบหกไม่เกิน 4 หลัก เป็นแอดเดรสของหน่วยความจำ

value ตัวเลขฐานสิบหกไม่เกิน 4 หลักใช้เป็นตัวนับจำนวนครั้งในการทำงาน 0 หมายถึง 65536 ครั้ง

range มี 2 ลักษณะคือ

address1 address2

โดย address2 ต้องมากกว่าหรือเท่ากับ address1

หรือ

address1 L value

จะได้ address2 = address1 + value - 1

โดย address2 ต้องมากกว่าหรือเท่ากับ address1

byte เลขฐานสิบหกไม่เกิน 2 หลัก

list ข้อมูลเป็นไบต์ได้ 1 ไบต์ หรือหลายไบต์ คั่นด้วย , หรือตัวว่าง และยังใช้เป็นตัวอักษรได้โดยอยู่ในเครื่องหมายคำพูด คู่ใดคู่หนึ่งคือ " " หรือ ' ' ถ้าเริ่มด้วยเครื่องหมายใดจะถือว่าหมดข้อความเมื่อพบเครื่องหมายนั้นโดยไม่สนใจอีกเครื่องหมายหนึ่ง เราจึงสามารถเขียนข้อความเหล่านี้ได้ " I'm a boy."

หรือ 'This "literal"is correct.'

สามารถใช้ข้อความปนกับข้อมูลเป็นไบต์ได้

register ชื่อรีจิสเตอร์ของแต่ละซีพียู ดูรายละเอียดในคำสั่ง R

Intel hex record ดูรายละเอียดในคำสั่ง L

พารามิเตอร์อื่นที่แสดงอักษรด้วยตัวพิมพ์ใหญ่ให้ใช้อักษรนั้นเป็นพารามิเตอร์

ตัวอักษรทั้งหมดสามารถใช้ได้ทั้งตัวพิมพ์เล็กและตัวพิมพ์ใหญ่ โดยมีความหมายเหมือนกันยกเว้นข้อความในเครื่องหมายคำพูด ใน list

คำสั่งต่างๆจะมีผลเมื่อกด Enter แล้วเท่านั้น

สามารถใช้ปุ่ม Back Space ลบตัวอักษรที่ป้อนไปแล้วได้

ในระหว่างพิมพ์คำสั่ง ถ้าต้องการยกเลิกกด Esc

ในการทำงานของบางคำสั่งจะแสดงผลจำนวนมากจนดูผลบนจอไม่ทัน สามารถใช้ปุ่ม Ctrl-Numlock หรือ Pause เพื่อหยุดจอชั่วคราวได้ และกดปุ่มใดๆเมื่อต้องการให้แสดงผลต่อ ถ้าต้องการหยุดการทำงานกด Esc

การใช้งานคำสั่งและการแสดงผลได้แสดงในเรื่องคู่มือการใช้งานแล้ว ในบทนี้จะอธิบายหลักการของซอฟต์แวร์ในแต่ละคำสั่งเฉพาะส่วนที่เกี่ยวกับข้อมูล และการนำข้อมูลมาใช้ในการควบคุมฮาร์ดแวร์เท่านั้น

ส่วนขั้นตอนการทำงานของแต่ละคำสั่ง เราสามารถอธิบายดังต่อไปนี้

<u>คำสั่งเรียกดูรูปแบบตารางคำสั่งทั้งหมด</u>

Help H

โปรแกรมย่อยแสดงข้อความที่บอกรูปแบบการใช้งานคำสั่งต่าง ๆ บนจอภาพ

<u>คำสั่งเกี่ยวกับการกำหนดการใช้หน่วยความจำ</u>

Memory Map MM [range [T|S][P|D][RW|RO]|[N]]

คำสั่งนี้ ใช้แสดงและแก้ไขการกำหนดการใช้หน่วยความจำของอินเซอร์กิตอีมูเลเตอร์ ถ้าไม่ใส่พารามิเตอร์จะแสดงค่าที่กำหนดไว้เดิม ในช่วงแอดเดรสทั้งหมด ถ้ากำหนด range จะแสดงเฉพาะช่วงแอดเดรสที่กำหนดเป็นช่วงละ 2 กิโลไบต์ เช่น 0 ถึง 7FFH คือ 2 กิโลไบต์แรก ถ้ากำหนดเป็น 0 ถึง 8 ก็จะเป็นช่วง 2 กิโลไบต์แรกเช่นกัน

ถ้ากำหนดพารามิเตอร์อื่นต่อท้ายจะเป็นการกำหนดสถานะใหม่โดยตัวอักษรแต่ละตัวที่อยู่คู่กันใน [ ] จะใช้พร้อมกันไม่ได้ ความหมายของตัวอักษรมีดังนี้

T ใช้หน่วยความจำจากระบบเป้าหมาย

S ใช้หน่วยความจำจากภายในอินเซอร์กิตอีมูเลเตอร์

P กำหนดให้เป็นหน่วยความจำสำหรับเก็บทั้งโปรแกรมและข้อมูล

D กำหนดให้เป็นหน่วยความจำสำหรับเก็บเฉพาะข้อมูล

RW กำหนดให้เป็นหน่วยความจำชนิดอ่านเขียนได้

RO กำหนดให้เป็นหน่วยความจำชนิดอ่านอย่างเดียว

N กำหนดว่าช่วงแอดเดรสนั้นไม่มีหน่วยความจำ

หลังจากแก้ไขจะแสดงสถานะที่กำหนดใหม่ ในช่วงแอดเดรสนั้นทั้งหมด

การทำงาน

1. หาพารามิเตอร์ โดยการเปรียบเทียบข้อมูลที่ผู้ใช้พิมพ์เข้ามาเก็บไว้ในบัฟเฟอร์กับพารามิเตอร์ ได้ช่วงแอดเดรสและสถานะ

2. อ่านข้อมูลจากวงจรถอดรหัสแอดเดรสแบบโปรแกรมได้ที่กำหนดการใช้หน่วยความจำของอินเซอร์กิตอีมูเลเตอร์ในช่วงแอดเดรสที่ผู้ใช้กำหนด มีขั้นตอนดังนี้

2.1 ทำให้อีมูเลชันซีพียู อยู่ในสถานะ WAIT ไม่ติดต่อหน่วยความจำ

2.2 ให้อีมูเลชันซีพียูอ่านคำสั่ง

```
LD HL,address
LD A,(HL)
```

2.3 อีมูเลชันซีพียูจะส่งพารามิเตอร์ address ออกมาที่บัสแอดเดรส วงจรถอดรหัสจะส่งข้อมูลสำหรับช่วงแอดเดรสนั้นออกมาให้อ่านได้จากพอร์ตอินพุต

3. เปลี่ยนข้อมูลตามพารามิเตอร์สถานะที่กำหนดใหม่

4. เขียนข้อมูลตามพารามิเตอร์สถานะที่กำหนดใหม่ในหน่วยความจำที่กำหนด การใช้หน่วยความจำของอินเซอร์กิตอีมูเลเตอร์ในช่วงแอดเดรสที่ผู้ใช้กำหนด มีขั้นตอนดังนี้

4.1 ทำให้อีมูเลชันซีพียู อยู่ในสถานะ WAIT ไม่ติดต่อหน่วยความจำ

4.2 ให้อีมูเลชันซีพียูอ่านคำสั่ง

```
LD HL,address
LD A,(HL)
```

4.3 อีมูเลชันซีพียูจะส่งพารามิเตอร์ address ออกมาที่บัสแอดเดรส คอนโทรลซีพียูจะส่งข้อมูลให้วงจรถอดรหัสแบบพอร์ตเอาต์พุต

5. แสดงช่วงแอดเดรสและสถานะที่กำหนดใหม่ทางจอภาพ

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

<u>คำสั่งเกี่ยวกับหน่วยความจำ</u>

Dump D [address|range]

คำสั่งนี้ใช้ดูค่าข้อมูลในหน่วยความจำของผู้ใช้โดยแสดงแอดเดรสเริ่มต้นของบล็อกต่อด้วยข้อมูลบรรทัดละ 16 ไบต์ ในรูปเลขฐาน 16 แล้วต่อด้วยข้อมูลในรูปตัวอักษรถ้าข้อมูลใดแทนด้วยตัวอักษรไม่ได้จะแสดงด้วยจุด

ถ้ากำหนด address คือ แอดเดรสเริ่มต้น ที่จะดูค่าข้อมูล จะแสดง 128 ไบต์

ถ้ากำหนด range คือ ช่วงแอดเดรสที่จะดูค่าข้อมูล

ถ้าไม่กำหนดพารามิเตอร์จะเริ่มแอดเดรสแรกที่ค่าต่อจากที่แสดงไว้ครั้งก่อนจำนวน 128 ไบต์

การทำงาน

1. หาพารามิเตอร์ โดยการเปรียบเทียบข้อมูลที่ผู้ใช้พิมพ์เข้ามาเก็บไว้ในบัฟเฟอร์กับพารามิเตอร์

2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้นและแอดเดรสสุดท้ายที่จะแสดง

3. อ่านข้อมูลจากหน่วยความจำของอีมูเลชันซีพียูในแอดเดรสที่กำหนด มีขั้นตอนดังนี้

3.1 ทำให้อีมูเลชันซีพียู อยู่ในสถานะ WAIT ไม่ติดต่อหน่วยความจำ

3.2 ให้อีมูเลชันซีพียูอ่านคำสั่ง

```
LD HL,address
LD A,(HL)
```

3.3 ให้อีมูเลชันซีพียูอยู่ในสถานะ BUSRQ ติดต่อหน่วยความจำ อีมูเลชันซีพียูเตรียมอ่านหน่วยความจำในแมชชีนไซเคิลต่อไป

3.4 ยกเลิกสถานะ BUSRQ ให้อีมูเลชันซีพียูทำงาน 1 แมชชีนไซเคิล แล้วกลับเป็นสถานะ BUSRQ อีมูเลชันซีพียูอ่านหน่วยความจำในแมชชีนไซเคิลนี้ เก็บข้อมูลไว้ในรีจิสเตอร์ A

3.5 ทำให้อีมูเลชันซีพียู อยู่ในสถานะ WAIT ไม่ติดต่อหน่วยความจำ

3.6 ให้อีมูเลชันซีพียูอ่านคำสั่ง

```
LD (HL),A
```

3.7 อีมูเลชันซีพียูจะส่งข้อมูลในรีจิสเตอร์ A ออกมาที่บัสข้อมูล คอนโทรลซีพียูจะอ่านข้อมูลนี้จากพอร์ตอินพุต

3.8 ยกเลิกสัญญาณ WAIT ให้อีมูเลชันซีพียูจบงานในแมชชีนไซเคิลนี้แล้วให้สัญญาณ WAIT อีกครั้ง

4. แสดงข้อมูลที่ได้ทางจอภาพ

5. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 3 จนจบแอดเดรสสุดท้าย

```
Enter          E address [list]
```

คำสั่งนี้ใช้แก้ไขค่าข้อมูลในหน่วยความจำ

ถ้ากำหนด list จะเป็นการใส่ข้อมูลใน list ไปที่หน่วยความจำเริ่มจากที่กำหนด address

ถ้ากำหนดเฉพาะ address จะใช้แก้ไขข้อมูลทีละไบต์ ถ้าไม่ต้องการแก้ไขให้กดแป้นอักขระว่างจะเป็นการแก้ไขข้อมูลแอดเดรสถัดไป ถ้าต้องการกลับแก้ไขแอดเดรสก่อนหน้ากดเครื่องหมายลบ (-) ถ้าต้องการเลิกการทำงานกด Enter ในขณะที่ไม่มีการแก้ไขข้อมูล

การทำงาน

1. หาพารามิเตอร์

2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้น

3. อ่านข้อมูลจากหน่วยความจำของอีมูเลชันซีพียูในแอดเดรสที่กำหนด มีขั้นตอนเช่นเดียวกับที่แสดงในคำสั่ง D

4. แสดงข้อมูลที่ได้ทางจอภาพ

5. เขียนข้อมูลให้หน่วยความจำของอีมูเลชันซีพียูในแอดเดรสที่กำหนด มีขั้นตอนดังนี้

5.1 ทำให้อีมูเลชันซีพียู อยู่ในสถานะ WAIT ไม่ติดต่อหน่วยความจำ

5.2 ให้อีมูเลชันซีพียูอ่านคำสั่ง

```
LD HL,address
LD (HL),byte
```

5.3 ให้อีมูเลชันซีพียูอยู่ในสถานะ BUSRQ ติดต่อหน่วยความจำ อีมูเลชันซีพียูเตรียมเขียนหน่วยความจำในแมชชีนไซเคิลต่อไป

5.4 ยกเลิกสถานะ BUSRQ ให้อีมูเลชันซีพียูทำงาน 1 แมชชีนไซเคิล แล้วกลับเป็นสถานะ BUSRQ อีมูเลชันซีพียูเขียนหน่วยความจำในแมชชีนไซเคิลนี้

5.5 ยกเลิกสัญญาณ BUSRQ ให้อีมูเลชันซีพียูอยู่ในสถานะ WAIT อีกครั้ง

6. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 3 จนจบ

Fill F range list

คำสั่งนี้ใช้ป้อนค่าข้อมูลใน list ลงสู่หน่วยความจำ ถ้า range ที่กำหนดยาวกว่าข้อมูลใน list ข้อมูลจะถูกใส่ซ้ำๆกันจนกว่าจะเต็ม ถ้า range สั้นกว่าข้อมูลที่เกินจะถูกตัดทิ้ง

การทำงาน

1. หาพารามิเตอร์

2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้นและแอดเดรสสุดท้าย

3. เขียนข้อมูลให้หน่วยความจำของอีมูเลชันซีพียูในแอดเดรสที่กำหนด

มีขั้นตอนเช่นเดียวกับที่แสดงในคำสั่ง E

4. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 3 จนจบ

Compare C range address

เปรียบเทียบข้อมูลในช่วงที่ระบุด้วยrange กับ ข้อมูลในช่วงที่เริ่มต้นด้วย address ถ้าข้อมูลต่างกันจะแสดงผลที่แตกต่างให้เห็นถ้าเหมือนกันทั้งหมดจะไม่แสดงผลใดๆ

การทำงาน

1. หาพารามิเตอร์

2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้น,แอดเดรสสุดท้ายและแอดเดรสเริ่มต้นของข้อมูลที่จะเปรียบเทียบด้วย

3. อ่านข้อมูลจากหน่วยความจำของอีมูเลชันซีพียูในแอดเดรสที่กำหนด

4. อ่านข้อมูลจากหน่วยความจำของอีมูเลชันซีพียูในแอดเดรสที่จะเปรียบเทียบด้วย

5. เปรียบเทียบข้อมูลที่ได้จากข้อ 3 และข้อ 4 ถ้าไม่เท่ากันแสดงผลบนจอภาพ

6. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 3 จนจบ

Move M range address

ย้ายข้อมูลจากหน่วยความจำช่วงหนึ่งไปยังอีกช่วงหนึ่ง

การทำงาน

1. หาพารามิเตอร์

2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้น,แอดเดรสสุดท้ายของต้นทางและแอดเดรสเริ่มต้น,แอดเดรสสุดท้ายของปลายทาง

3. ถ้าแอดเดรสเริ่มต้นของต้นทางน้อยกว่าแอดเดรสเริ่มต้นของปลายทางให้ย้ายข้อมูลจากแอดเดรสสุดท้ายไปก่อนเพื่อป้องกันข้อมูลต้นทางสูญหายก่อนเคลื่อนย้ายเสร็จ

4. ถ้าแอดเดรสเริ่มต้นของต้นทางมากกว่าแอดเดรสเริ่มต้นของปลายทางให้ย้ายข้อมูลจากแอดเดรสเริ่มต้นไปก่อนเพื่อป้องกันข้อมูลต้นทางสูญหายก่อนเคลื่อนย้ายเสร็จ

5. การเคลื่อนย้ายข้อมูลใช้การอ่านและเขียนข้อมูลในหน่วยความจำของอีมูเลชันซีพียูแบบเดียวกับที่แสดงแล้วในคำสั่ง D และ E

Memory Test MT range

ทดสอบอ่านเขียนหน่วยความจำในช่วงที่กำหนด เพื่อตรวจสอบว่าเป็นหน่วยความจำแรมที่ถูกต้องหรือไม่ถ้าไม่ถูกต้องจะแสดงแอดเดรสเริ่มต้นที่อ่านได้ไม่ถูกต้องจากที่เขียนไว้

การทำงาน

1. หาพารามิเตอร์
2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้น,แอดเดรสสุดท้าย
3. เขียนข้อมูลด้วยค่าที่เกิดจาก แอดเดรสไบต์สูง XOR แอดเดรสไบต์ต่ำ
4. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 3 จนจบ
5. อ่านข้อมูลเปรียบเทียบกับค่า แอดเดรสไบต์สูง XOR แอดเดรสไบต์ต่ำ ถ้าไม่เท่ากันให้แสดงข้อผิดพลาด
6. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 5 จนจบ
7. เขียนข้อมูลด้วยค่าที่เกิดจาก แอดเดรสไบต์สูง XNOR แอดเดรสไบต์ต่ำ
8. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 7 จนจบ
9. อ่านข้อมูลเปรียบเทียบกับค่า แอดเดรสไบต์สูง XNOR แอดเดรสไบต์ต่ำ ถ้าไม่เท่ากันให้แสดงข้อผิดพลาด
10. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 9 จนจบ
11. ถ้าพบข้อผิดพลาดในหลายแอดเดรสให้แสดงข้อผิดพลาดที่แอดเดรสต่ำสุด

Search S range list

หาค่าที่กำหนดใน list ในช่วงที่กำหนดโดย range

การทำงาน

1. หาพารามิเตอร์
2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้น,แอดเดรสสุดท้าย
3. อ่านข้อมูลจากหน่วยความจำของอีมูเลชันซีพียูในแอดเดรสที่กำหนด เปรียบเทียบกับข้อมูลใน list ถ้าเท่ากันอ่านแอดเดรสต่อไปมาเปรียบเทียบกับข้อมูลต่อไปของ list จนหมดข้อมูลของ list ถ้าเท่ากันทั้งหมดแสดงผลบนจอภาพด้วยแอดเดรสเริ่มต้นที่เปรียบเทียบได้ตรง
4. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 3 จนจบ

<u>คำสั่งเกี่ยวกับอินพุตเอาต์พุต</u>

Input I address

คำสั่งให้อ่านค่าจากพอร์ตแอดเดรสที่กำหนด

แอดเดรสของพอร์ตสำหรับ Z-80 จะมีได้ตั้งแต่ 0 ถึง FFFF

สำหรับ 8085 มีค่าได้ไม่เกิน FF

การทำงาน

1. หาพารามิเตอร์

2. อ่านข้อมูลจากพอร์ตอินพุตของอีมูเลชันซีพียูในแอดเดรสที่กำหนด มีขั้นตอนดังนี้

2.1 ทำให้อีมูเลชันซีพียู อยู่ในสถานะ WAIT ไม่ติดต่อหน่วยความจำ

2.2 ให้อีมูเลชันซีพียูอ่านคำสั่ง

สำหรับ Z-80

```
LD BC,address
IN A,(C)
```

สำหรับ 8085

```
IN address
```

2.3 ให้อีมูเลชันซีพียูอยู่ในสถานะ BUSRQ ติดต่อหน่วยความจำ อีมูเลชันซีพียูเตรียมอ่านพอร์ตอินพุตในแมชชีนไซเคิลต่อไป

2.4 ยกเลิกสถานะ BUSRQ ให้อีมูเลชันซีพียูทำงาน 1 แมชชีนไซเคิล แล้วกลับเป็นสถานะ BUSRQ อีมูเลชันซีพียูอ่านพอร์ตอินพุตในแมชชีนไซเคิลนี้ เก็บข้อมูลไว้ในรีจิสเตอร์ A

2.5 ทำให้อีมูเลชันซีพียู อยู่ในสถานะ WAIT ไม่ติดต่อหน่วยความจำ

2.6 ให้อีมูเลชันซีพียูอ่านคำสั่ง

```
LD (HL),A
```

2.7 อีมูเลชันซีพียูจะส่งข้อมูลในรีจิสเตอร์ A ออกมาที่บัสข้อมูล คอนโทรลซีพียูจะอ่านข้อมูลนี้จากพอร์ตอินพุต

2.8 ยกเลิกสัญญาณ WAIT ให้อีมูเลชันซีพียูจบงานในแมชชีนไซเคิลนี้แล้วให้สัญญาณ WAIT อีกครั้ง

3. แสดงข้อมูลที่ได้ทางจอภาพ

Output O address byte

คำสั่งให้ส่งค่าข้อมูลที่กำหนดออกไปที่พอร์ตแอดเดรสที่กำหนด

การทำงาน

1. หาพารามิเตอร์

2. เขียนข้อมูลให้พอร์ตเอาต์พุตของอีมูเลชันซีพียูในแอดเดรสที่กำหนด มีขั้นตอนดังนี้

2.1 ทำให้อีมูเลชันซีพียู อยู่ในสถานะ WAIT ไม่ติดต่อหน่วยความจำ

2.2 ให้อีมูเลชันซีพียูอ่านคำสั่ง

สำหรับ Z-80

```
LD BC,address
LD A,byte
OUT (C),A
```

สำหรับ 8085

```
MVI A,byte
OUT address
```

2.3 ให้อีมูเลชันซีพียูอยู่ในสถานะ BUSRQ ติดต่อหน่วยความจำ อีมูเลชันซีพียูเตรียมเขียนพอร์ตเอาต์พุตในแมชชีนไซเคิลต่อไป

2.4 ยกเลิกสถานะ BUSRQ ให้อีมูเลชันซีพียูทำงาน 1 แมชชีนไซเคิล แล้วกลับเป็นสถานะ BUSRQ อีมูเลชันซีพียูเขียนพอร์ตเอาต์พุตในแมชชีนไซเคิลนี้

2.5 ยกเลิกสัญญาณ BUSRQ ให้อีมูเลชันซีพียูอยู่ในสถานะ WAIT อีกครั้ง

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

<u>คำสั่งเกี่ยวกับการจัดการข้อมูลเป็นภาษาแอสเซมบลี</u>

Assemble A [address]

คำสั่งนี้ทำให้ผู้ใช้สามารถป้อนโปรแกรมภาษาแอสเซมบลีลงสู่หน่วยความจำ ทีละบรรทัดได้และจะทำการแปลเป็นภาษาเครื่องแสดงไว้ท้ายบรรทัดด้วย

การทำงาน

1. หาพารามิเตอร์

2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้น

3. แสดงแอดเดรสเริ่มต้นที่ต้องการป้อนโปรแกรม และรอรับโปรแกรมเป็นภาษาแอสเซมบลี จนผู้ใช้กด Enter

4. นำคำสั่งที่เขียนเป็นนิวโมนิกนั้นไปเปิดตารางเทียบออฟโค้ด แล้วแปลโปรแกรมออกมาในรูปของภาษาเครื่อง

5. นำมาแสดงต่อท้ายบรรทัดเดิม และนำค่าที่ได้ไปเก็บในหน่วยความจำของอิมูเลชันซีนียู หากมีการป้อนคำสั่งผิดรูปแบบก็ให้แสดง Error บนจอ

6. เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 3 จนจบเมื่อผู้ใช้กด Enter โดยที่ไม่ได้ป้อนโปรแกรมในบรรทัดนั้น

Unassemble U [address[range]

คำสั่งให้แสดงข้อมูลในหน่วยความจำออกมาเป็นโปรแกรมภาษาแอสเซมบลี บนหน้าจอ หากไม่กำหนดแอดเดรสท้าย คำสั่งจะแสดงทีละ 11 บรรทัด ถ้าไม่กำหนดพารามิเตอร์จะเริ่มแอดเดรสแรกที่ค่าต่อจากที่แสดงไว้ครั้งก่อน

การทำงาน

1. หาพารามิเตอร์

2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้น

3. อ่านข้อมูลจากหน่วยความจำเพจผู้ใช้ในแอดเดรสที่กำหนด แล้วเทียบเป็นภาษาแอสเซมบลีด้วยการเปิดตาราง หากเป็นคำสั่งที่ใช้มากกว่า 1 ไบต์ก็จะอ่านข้อมูลในแอดเดรสถัดไปมาใช้

4. หลังจากแปลเป็นคำสั่งสมบูรณ์แล้วก็แสดงบนหน้าจอ

5 เปลี่ยนแอดเดรสเป็นค่าต่อไปแล้วกลับไปทำงานในข้อ 3 จนจบ

คำสั่งเกี่ยวกับการจัดการข้อมูลรูปแบบ Intel hex

ข้อมูลรูปแบบ Intel hex มีลักษณะดังนี้

Intel HEX-ASCII format

```
: B C A A A A T T H H ... H H C C
  |_| |_____| |_| |_| |_| |_| |_|
    \     \     \   \   \   \   \__ checksum    : checksum + sum = 0
      \     \     \   \___\___\____ data bytes  : 2 characters per byte
        \     \     \_______________ record type: 00=data,01=end of file
          \     \___________________ address     : first address
            \_____________________ byte count : number of data bytes
```

data used ASCII 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

NUL (00) and DEL (7F) are ignored.

example

```
:10000000316502216502360003A6502FE01D21B000D
:10001000E04D39021650234C3080021650236FFF7
:10002000A6502FE01D233003E05D39021650234C9
:1000300C32000216502360003A6502FE02D24B0061
:10004000E06D39021650234C33800216502360094
:10005000A6502FE01D263003E04D390216502346A
:05006000C35000FB7617
:00000001FF
```

*** byte count in end of file record must be 00

Load L

คำสั่งรับแฟ้มข้อมูลในรูปแบบ Intel hex จากเครื่องไมโครคอมพิวเตอร์ ไปยังหน่วยความจำของระบบที่พัฒนา ด้วยคำสั่ง L

การทำงาน

1. รอรับข้อมูลในรูปแบบของ Intel hex จากเครื่องไมโครคอมพิวเตอร์ IBM PC โดยรอรับ ":" ถ้าพบตัวอักษรอื่นก่อนจะแสดงข้อความ Load error แล้วเลิกการทำงาน

2. เมื่อพบ ":" แล้วรับตัวอักษรตัวต่อๆไปจะต้องเป็นตัวอักษร 0 ถึง F ถ้าพบตัวอักษรอื่นจะแสดงข้อความ Load error แล้วเลิกการทำงาน

3. ตัวอักษร 0 ถึง F 2 ตัวแรกจะนำมาแปลงเป็นเลขฐานสิบหก 2 หลักเพื่อบอกจำนวนข้อมูล

4. ตัวอักษร 4 ตัวถัดไปใช้บอกเลขแอดเดรสเริ่มต้น

5. ตัวอักษร 2 ตัวถัดไปจะต้องเป็น 00 หรือ 01 เท่านั้น ถ้าเป็น 00 จำนวนข้อมูลจะต้องเป็น 0 ด้วย

6. ตัวอักษรถัดไปจะถูกจัดเป็นคู่เพื่อแทนเลขฐานสิบหก 2 หลักเก็บไว้เป็นข้อมูลจนกว่าจะครบจำนวนข้อมูลที่กำหนด

7. หลังจากครบจำนวนข้อมูลแล้วผลรวมของเลขฐานสิบหก 2 หลักตั้งแต่เริ่มต้นเมื่อรวมกับเลขฐานสิบหก 2 หลักถัดจากนี้จะต้องเป็น 0 ถ้าไม่ใช่จะแสดงข้อความ Load error แล้วเลิกการทำงาน

8. นำข้อมูลทั้งหมดที่ได้ไปเขียนในหน่วยความจำเริ่มจากแอดเดรสที่กำหนด แล้วส่ง ACK ไปให้เครื่องคอมพิวเตอร์ที่ติดต่อด้วย

9. ถ้าข้อมูลที่รับได้ไม่ได้แสดงว่าเป็นบรรทัดสุดท้ายของแฟ้มข้อมูลให้กลับไปเริ่มที่ข้อ 1 ถ้าจบแฟ้มข้อมูล เลิกการทำงาน

Write W range

คำสั่งส่งข้อมูลจากหน่วยความจำในช่วงแอดเดรสที่กำหนดลงสู่ดิสค์ของเครื่องไมโครคอมพิวเตอร์ เป็นแฟ้มข้อมูลในรูปแบบ Intel hex

การทำงาน

1. หาพารามิเตอร์
2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้นและแอดเดรสสุดท้าย
3. รอให้ผู้ใช้เตรียมเครื่องคอมพิวเตอร์สำหรับรับข้อมูลเมื่อพร้อมแล้วผู้ใช้กดปุ่มใดๆ
4. อ่านข้อมูลจากหน่วยความจำของอีมูเลชันชีพียูในแอดเดรสที่กำหนด
5. แสดงข้อมูลที่ได้ทางจอภาพในรูปแบบ Intel hex จำนวนบรรทัดละไม่เกิน 16 ไบต์
6. เปลี่ยนแอดเดรสเพิ่มขึ้นแล้วกลับไปทำข้อ 4 จนกว่าจะครบตามพารามิเตอร์

การส่งรับข้อมูล วิธีที่จัดรูปแบบการส่งข้อมูลที่มีการตรวจสอบข้อผิดพลาดที่ดีและมีโปรแกรมจัดการข้อมูลอยู่ทั้งด้านอินเซอร์กิตอีมูเลเตอร์และด้านเครื่องคอมพิวเตอร์ซึ่งจะแสดงผลออกมาในรูปแบบต่างๆกันได้ตามความต้องการ และทางด้านอินเซอร์กิตอีมูเลเตอร์ก็จะมีงานในการแปลคำสั่งและจัดการข้อมูลน้อยลง แต่วิธีการนี้ต้องมีงานการออกแบบซอฟแวร์ของเครื่องคอมพิวเตอร์อีกมาก และควรออกแบบให้ใช้งานได้กว้างกว่าการใช้กับอินเซอร์กิตอีมูเลเตอร์รุ่นนี้เท่านั้น ในงานวิจัยนี้จะเสนอเฉพาะรูปแบบข้อมูลและโปรแกรมทางด้านอินเซอร์กิตอีมูเลเตอร์เท่านั้นซึ่งแสดงเป็นคำสั่งสำหรับอินเซอร์กิตอีมูเลเตอร์ได้ดังนี้

display user memory	J Intel hex record
display control memory	Y Intel hex record
enter user memory	K Intel hex record
enter control memory	Z Intel hex record

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

คำสั่ง J และ Y เป็นคำสั่งให้อินเซอร์กิตอีมูเลเตอร์ส่งข้อมูลที่อยู่ในหน่วยความจำไปให้เครื่องคอมพิวเตอร์ที่มีโปรแกรมรับข้อมูลอยู่ คำสั่งที่ใช้จะมีรูปแบบแตกต่างจากที่ใช้ในการติดต่อกับเทอร์มินอลคือ พารามิเตอร์อยู่ในรูปแบบ Intel hex จึงมีการตรวจสอบความผิดพลาดของการส่งรับข้อมูลได้ดีขึ้น พารามิเตอร์นี้จะมี 2 ตัว คือแอดเดรสเริ่มต้นอยู่ในส่วนแอดเดรสเริ่มต้นของรูปแบบ Intel hex และจำนวนข้อมูลที่ต้องการให้แสดงจะอยู่ในส่วนข้อมูลที่มีเพียงตัวเดียว ตัวอย่างเช่น

J:01000000FF00

คือคำสั่งให้แสดงข้อมูลในหน่วยความจำที่ติดต่อกับอีมูเลชันชีพียูตั้งแต่แอดเดรส 0 เป็นจำนวน 255 ไบต์

การทำงาน

1. หาพารามิเตอร์ จะต้องอยู่ในรูป Intel hex ที่มีข้อมูลเพียง 1 ตัว สำหรับบอกจำนวนข้อมูลที่ต้องการให้แสดงผล

2. จากพารามิเตอร์ กำหนดแอดเดรสเริ่มต้นและจำนวนข้อมูล

3. สำหรับคำสั่ง J

อ่านข้อมูลจากหน่วยความจำของอีมูเลชันชีพียูในแอดเดรสที่กำหนด

สำหรับคำสั่ง Y

อ่านข้อมูลจากหน่วยความจำของคอนโทรลชีพียูในแอดเดรสที่กำหนด

4. แสดงข้อมูลที่ได้ทางจอภาพในรูปแบบ Intel hex

5. เปลี่ยนแอดเดรสเพิ่มขึ้นแล้วกลับไปทำข้อ 3 จนกว่าจะครบจำนวนข้อมูล

คำสั่ง K และ Z เป็นคำสั่งให้อินเซอร์กิตอีมูเลเตอร์รับข้อมูลไปใส่ในหน่วยความจำ คำสั่งที่ใช้จะมีรูปแบบคล้ายการรับข้อมูล Intel hex ในคำสั่ง L

การทำงาน

1. ตรวจสอบพารามิเตอร์จากบัฟเฟอร์ที่เก็บข้อมูลคำสั่งที่รับมา ถ้าไม่ใช่ ":" จะแสดงข้อความ error แล้วเลิกการทำงาน

2. เมื่อพบ ":" แล้วตัวอักษรตัวต่อๆไปจะต้องเป็นตัวอักษร 0 ถึง F ถ้าพบตัวอักษรอื่นจะแสดงข้อความ error แล้วเลิกการทำงาน

3. ตัวอักษร 0 ถึง F 2 ตัวแรกจะนำมาแปลงเป็นเลขฐานสิบหก 2 หลักเพื่อบอกจำนวนข้อมูล

4. ตัวอักษร 4 ตัวถัดไปใช้บอกเลขแอดเดรสเริ่มต้น

5. ตัวอักษร 2 ตัวถัดไปจะต้องเป็น 00 หรือ 01 เท่านั้น ถ้าเป็น 00 จำนวนข้อมูลจะต้องเป็น 0 ด้วย

6. ตัวอักษรถัดไปจะถูกจัดเป็นคู่เพื่อแทนเลขฐานสิบหก 2 หลักเก็บไว้เป็นข้อมูลจนกว่าจะครบจำนวนข้อมูลที่กำหนด

7. หลังจากครบจำนวนข้อมูลแล้วผลรวมของเลขฐานสิบหก 2 หลักตั้งแต่เริ่มต้นเมื่อรวมกับเลขฐานสิบหก 2 หลักถัดจากนี้จะต้องเป็น 0 ถ้าไม่ใช่จะแสดงข้อความ Load error แล้วเลิกการทำงาน

8. นำข้อมูลทั้งหมดที่ได้ไปเขียนในหน่วยความจำเริ่มจากแอดเดรสที่กำหนด โดยเขียนในหน่วยความจำของอีมูเลชันซีพียูสำหรับคำสั่ง K และหน่วยความจำของคอนโทรลซีพียูสำหรับคำสั่ง Z แล้วส่ง ACK ไปให้เครื่องคอมพิวเตอร์ที่ติดต่อด้วย

<u>คำสั่งควบคุมการรับสัญญาณควบคุมซีพียูจากระบบเป้าหมาย</u>

Pin P

สัญญาณควบคุมซีพียูจากระบบเป้าหมายแบ่งเป็น 4 กลุ่ม คือ

RESET (Z-80,8085)

BUSRQ (Z-80) หรือ HOLD (8085)

NMI (Z-80) หรือ TRAP (8085)

INT (Z-80) หรือ INTR, RST5.5, RST6.5, RST7.5

การใช้งานจะแสดงการกำหนดเดิมและให้แก้ไขด้วยตัวอักษร D (Disable) หรือ E (Enable)

การทำงาน

1. จะแสดงบนจอภาพเป็นชื่อขาสัญญาณและสถานะที่ตั้งไว้

2. รอรับตัวอักษรจากผู้ใช้เพื่อเปลี่ยนแปลงสถานะ

3. เก็บสถานะที่เปลี่ยนแปลงใหม่ไว้เป็นข้อมูลที่จะใช้เมื่อเปลี่ยนโมดเป็นอีมูเลชัน

คำสั่งรีเซตซีพียู

reset X

คำสั่งรีเซตให้ซีพียูกลับไปทำงานที่แอดเดรส 0000H

การทำงาน

1. คอนโทรลซีพียูส่งสัญญาณ C_SOD ให้เป็น 0
2. ตรวจสอบสถานะของอีมูเลชันซีพียูจนพบว่ามีการรีเซต
3. ให้อีมูเลชันซีพียูอยู่ในสถานะ WAIT
4. เขียนข้อมูลที่เก็บค่ารีจิสเตอร์ของอีมูเลชันซีพียูให้เหมือนกับสถานะเริ่มต้นที่ซีพียูรีเซต เช่น PC = 0000

คำสั่งเกี่ยวกับการแสดงและแก้ไขค่ารีจิสเตอร์

Register R [register]

ชื่อของรีจีสเตอร์ สำหรับ Z-80 ได้แก่

PC ,AF ,BC ,DE ,HL ,SP ,IX ,

IY ,AF',BC',DE',HL',A ,F ,B ,C ,D ,E ,H ,L ,I

IFF = Interrupt enable Flip Flop (IFF2)

SF = Sign Flag

ZF = Zero Flag

HF = Haft-carry Flag

HC = Haft-Carry Flag

PF = Parity/overflow Flag

NF = subtract flag

CF = Carry Flag

สำหรับ 8085 ได้แก่

PC ,AF ,BC ,DE ,HL ,SP ,A ,F ,B ,C ,D ,E ,H ,L

IM = Interrupt Mask

IE = Interrupt Enable

SF = Sign Flag

ZF = Zero Flag

AC = Auxiliary Carry

PF = Parity Flag

CF = Carry Flag

หากคำสั่งนี้ไม่มีชื่อรีจีสเตอร์ต่อท้าย จะเป็นคำสั่งขอดูค่าในรีจีสเตอร์ทั้งหมดแต่ถ้า มีชื่อรีจีสเตอร์ต่อท้ายคำสั่งจะเป็นการแก้ไขค่าในรีจีสเตอร์ตัวนั้นโดยจะแสดงค่าเดิมให้ดูด้วย

การทำงาน

1. หาพารามิเตอร์

2. ถ้าไม่มีพารามิเตอร์ให้นำข้อมูลค่ารีจิสเตอร์ทั้งหมดมาแสดงบนจอภาพ

3. ถ้ามีพารามิเตอร์ให้แสดงค่ารีจิสเตอร์ตัวนั้นแล้วรอรับค่าใหม่จากผู้ใช้ไปเก็บไว้เป็นข้อมูลเพื่อใช้ก่อนเปลี่ยนโมดเป็นอีมูเลชัน

<u>คำสั่งเกี่ยวกับการทำงานในโปรแกรมของผู้ใช้ทีละคำสั่ง</u>

Trace T [value|I|M]

ทำงานในโปรแกรมของผู้ใช้ทีละคำสั่งถ้ากำหนดพารามิเตอร์เป็น I จะแสดงคำสั่งที่กำลังจะทำงานในรูปแอดเดรส สถานะและข้อมูลออพโค้ด รอให้ผู้ใช้กดปุ่มใดๆยกเว้น Enter และ Esc จะทำงานในคำสั่งนั้นแล้วแสดงคำสั่งนั้นเป็นภาษาแอสเซมบลี แล้วแสดงคำสั่งต่อไปเพื่อรอการทำงานถ้าต้องการเลิกทำงานให้กด Enter หรือ Esc

ทำงานในโปรแกรมของผู้ใช้ทีละแมชชีนไซเคิลถ้ากำหนดพารามิเตอร์เป็น M จะแสดงคำสั่งที่จะทำงานในรูปแอดเดรส สถานะและข้อมูล รอให้ผู้ใช้กดปุ่มใดๆก็จะทำงานในแมชชีนไซเคิลนั้นแล้วแสดงแมชชีนไซเคิลต่อไป สำหรับแมชชีนไซเคิลสุดท้ายของคำสั่งจะแสดงคำสั่งเป็นภาษาแอสเซมบลีด้วย การเลิกทำงาน เหมือนการใช้พารามิเตอร์ I

อักษรย่อที่ใช้แทนสถานะของซีพียู มีดังนี้

F = opcode Fetch

R = memory Read

W = memory Write

I = Input

O = Output

A = interrupt Acknowledge

H = Halt

ถ้ากำหนดพารามิเตอร์เป็น value จะทำงานทีละคำสั่งแล้วแสดงค่ารีจิสเตอร์ที่ได้จากคำสั่งนั้นแล้วทำงานคำสั่งต่อไปจนครบจำนวนคำสั่งที่กำหนดไว้ ถ้าไม่กำหนดพารามิเตอร์จะทำงานแบบนี้เพียงคำสั่งเดียว หลังจากเลิกทำงานจะแสดงคำสั่งต่อไปในรูปภาษาแอสเซมบลีอีกหนึ่งคำสั่ง

การทำงาน

1. หาพารามิเตอร์
2. คืนค่ารีจิสเตอร์ให้กับอีมูเลชันซีพียู
3. ให้อีมูเลชันซีพียูอยู่ในสถานะ WAIT และติดต่อกับหน่วยความจำ
4. ถ้าพารามิเตอร์เป็น M

4.1 อ่านสัญญาณแอดเดรส สถานะและข้อมูลของอีมูเลชันซีพียูมาแสดงบนจอ

4.2 ถ้าผู้ใช้กดปุ่มใดๆนอกจาก Enter หรือ Esc ให้ยกเลิกสัญญาณ WAIT ในแมชชีนไซเคิลนั้น อีมูเลชันซีพียูจะทำงานในแมชชีนไซเคิลนั้นแล้วและรอการทำงานในสถานะ WAIT ในแมชชีนไซเคิลต่อไป แล้วกลับไปทำงาน ข้อ 4.1 จนกว่าผู้ใช้กดปุ่ม Enter หรือ Esc สำหรับแมชชีนไซเคิลสุดท้ายของคำสั่งจะแสดงคำสั่งเป็นภาษาแอสเซมบลีด้วย

5. ถ้าพารามิเตอร์เป็น I

5.1 อ่านสัญญาณแอดเดรส สถานะและข้อมูลของอีมูเลชันซีพียูมาแสดงบนจอ

5.2 ถ้าผู้ใช้กดปุ่มใดๆนอกจาก Enter หรือ Esc ให้ยกเลิกสัญญาณ WAIT ในแมชชีนไซเคิลนั้น อีมูเลชันซีพียูจะทำงานในแมชชีนไซเคิลนั้นแล้วและรอการทำงานในสถานะ WAIT ในแมชชีนไซเคิลต่อไป ถ้าแมชชีนไซเคิลนั้นไม่ใช่แมชชีนไซเคิลแรกของคำสั่งภาษาเครื่องให้ยกเลิกสัญญาณ WAIT ไปยังแมชชีนไซเคิลต่อไป จนพบแมชชีนไซเคิลแรกของคำสั่งจึงแสดงคำสั่งที่ผ่านมาเป็นภาษาแอสเซมบลี แล้วกลับไปทำงาน ข้อ 5.1 จนกว่าผู้ใช้กดปุ่ม Enter หรือ Esc

6. ถ้าไม่มีพารามิเตอร์

6.1 อ่านข้อมูลในหน่วยความจำที่โปรแกรมกำลังจะทำงานมาแปลเป็นภาษาแอสเซมบลี แสดงบนจอภาพ

6.2 ยกเลิกสัญญาณ WAIT 1 แมชชีนไซเคิลแล้วเก็บค่ารีจิสเตอร์แสดงบนจอภาพ

7. ถ้ามีพารามิเตอร์ value

7.1 อ่านข้อมูลในหน่วยความจำที่โปรแกรมกำลังจะทำงานมาแปลเป็นภาษาแอสเซมบลี แสดงบนจอภาพ

7.2 ยกเลิกสัญญาณ WAIT 1 แมชชีนไซเคิล แล้วเก็บค่ารีจิสเตอร์แสดงบนจอภาพ

7.3 ลดค่า value ลง 1 ถ้า value ไม่เท่ากับ 0 กลับไปทำข้อ 7.1

การทำงานของโปรแกรมคืนค่ารีจิสเตอร์

1. ให้อีมูเลชันชีพียูอยู่ในสถานะ WAIT และติดต่อกับหน่วยความจำ
2. ให้อีมูเลชันชีพียูอ่านคำสั่ง

```
E1              POP  HL  ;RESTORE L' THEN H'
D1              POP  DE  ;RESTORE E' THEN D'
C1              POP  BC  ;RESTORE C' THEN B'
F1              POP  AF  ;RESTORE F' THEN A'
D9              EXX      ;BC' = BC,DE' = DE,HL' = HL
08              EX AF,AF';AF' = AF
FD E1           POP  IY  ;RESTORE LOW(IY) THEN HIGH
DD E1           POP  IX  ;RESTORE LOW(IX) THEN HIGH
                ;IF SAVE_IFF = 01B THEN
FB                 EI
                ;IF SAVE_IFF = 10B THEN
F3                 DI
F1              POP  AF  ;RESTORE NUL FOR IFF THEN I
ED 47           LD I,A   ;I = A
E1              POP  HL  ;RESTORE L THEN H
D1              POP  DE  ;RESTORE E THEN D
C1              POP  BC  ;RESTORE C THEN B
F1              POP  AF  ;RESTORE F THEN A
31 00 00        LD SP,00 ;RESTORE SP
```

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

```
C3 00 00            JP 0000  ;RESTORE PC
```

3. การทำงานในข้อ 2 จะพบว่าอีมูเลชันซีพียูมีหน้าที่อ่านคำสั่งเพียงอย่างเดียว คำสั่งและค่าคงที่ ที่วงจรควบคุมกำหนดเป็นค่ารีจิสเตอร์ได้มาจากข้อมูลที่เก็บค่ารีจิสเตอร์หลังการทำงานในโมดอีมูเลชันครั้งที่แล้วร่วมกับข้อมูลที่ผู้ใช้กำหนดด้วยคำสั่ง R

การทำงานของโปรแกรมเก็บค่ารีจิสเตอร์

1. ให้อีมูเลชันซีพียูอยู่ในสถานะ WAIT และติดต่อกับหน่วยความจำ
2. ถ้าแมชชีนไซเคิลนั้นไม่ใช่แมชชีนไซเคิลแรกของคำสั่งภาษาเครื่องให้ยกเลิกสัญญาณ WAIT ไปยังแมชชีนไซเคิลต่อไป จนพบแมชชีนไซเคิลแรกของคำสั่ง
3. ให้อีมูเลชันซีพียูอ่านคำสั่ง

```
                    ;SAVE PC FORM ADDRESS BUS
F5                  PUSH AF  ;SAVE SP FROM ADDRESS BUS
                             ;SAVE A THEN F
C5                  PUSH BC  ;SAVE B THEN C
D5                  PUSH DE  ;SAVE D THEN E
E5                  PUSH HL  ;SAVE H THEN L
ED 57               LD A,I   ;A = I ,BIT(F,2) = IFF
F5                  PUSH AF  ;SAVE I THEN IFF
DD E5               PUSH IX  ;SAVE HIGH(IX) THEN LOW
FD E5               PUSH IY  ;SAVE HIGH(IY) THEN LOW
08                  EX AF,AF';AF = AF'
D9                  EXX      ;BC = BC',DE = DE',HL = HL'
F5                  PUSH AF  ;SAVE A' THEN F'
C5                  PUSH BC  ;SAVE B' THEN C'
D5                  PUSH DE  ;SAVE D' THEN E'
E5                  PUSH HL  ;SAVE H' THEN L'
```

3. การทำงานในข้อ 2 จะพบว่าอีมูเลชันซีพียูมีหน้าที่อ่านคำสั่งและส่งข้อมูลออกมาที่บัสเป็นค่ารีจิสเตอร์ต่างๆวงจรควบคุมจะเก็บค่าเหล่านี้ไว้ใช้ในการทำงานครั้งต่อไป

<u>คำสั่งเกี่ยวกับการกำหนดจุดหยุด</u>

Breakpoint B [P[S address [value]|C]]|[H[S|C]]|[X[H|L|C]]

การกำหนดจุดหยุดในคำสั่งนี้มี 3 แบบ คือ

การกำหนดจุดหยุดเป็นแอดเดรสที่ซีพียูอ่านคำสั่ง 1 แอดเดรส มีการกำหนด value ใช้เป็นตัวนับจำนวนครั้งในการผ่านแอดเดรสจุดหยุด ก่อนให้หยุดทำงานจริงๆ (ถ้าไม่กำหนด value คือ 1 ครั้ง 0 หมายถึง 65536 ครั้ง)

กำหนดให้หยุดเมื่อซีพียู HALT

กำหนดให้หยุดเมื่อสัญญาณภายนอกเป็นสถานะตรงที่กำหนด

การทำงานสำหรับคำสั่งต่างๆมีดังนี้

BPS address [value]

นำค่าพารามิเตอร์มาเก็บเป็นข้อมูลเพื่อกำหนดให้ฮาร์ดแวร์ก่อนการทำงานในเวลาจริง ค่าที่เก็บคือแอดเดรสที่กำหนดเป็นจุดหยุด จำนวนครั้งที่นับจุดหยุดก่อนจะหยุดทำงานจริงและสถานะที่แสดงว่ามีการกำหนดจุดหยุดหรือไม่

BP

เป็นการเรียกดูค่าแอดเดรสที่กำหนดเป็นจุดหยุด

BPC

เป็นการกำหนดให้ไม่ใช้จุดหยุดเมื่อทำงานในเวลาจริง

BHS

เป็นการกำหนดให้หยุดทำงานเมื่ออีมูเลชันซีพียูอยู่ในสถานะ HALT

BH

เป็นการเรียกดูสถานะว่ากำหนดให้หยุดที่สถานะ HALT หรือไม่

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

BHC

เป็นการกำหนดให้ไม่ใช้การหยุดที่สถานะ HALT

BXH

เป็นการกำหนดให้หยุดทำงานเมื่อสัญญาณภายนอกมีสถานะ 1

BXL

เป็นการกำหนดให้หยุดทำงานเมื่อสัญญาณภายนอกมีสถานะ 0

BXC

เป็นการกำหนดไม่ให้ใช้การหยุดจากสัญญาณภายนอก

B

เรียกดูสถานะการหยุดทุกๆแบบที่กำหนดไว้

<u>คำสั่งเกี่ยวกับการให้ซีพียูทำงานในเวลาจริงจนกว่าจะพบจุดหยุด</u>

Go G [address]

คำสั่งให้ซีพียูทำงานตามเวลาจริงในโปรแกรม โดยเริ่มที่แอดเดรสที่ผู้ใช้กำหนดต่อท้ายคำสั่ง แต่ถ้าไม่มี จะเริ่มที่แอดเดรสซึ่งค่า PC ชี้อยู่ เมื่อหยุดจะแสดงสถานะการหยุด คำสั่งที่กำลังจะทำงานในรูปแอดเดรส สถานะและข้อมูลออฟโค้ด

การทำงาน

1. คืนค่ารีจิสเตอร์ จะได้แอดเดรสขณะนั้นเท่ากับค่า PC

2. ขณะที่อีมูเลชันซีพียูอยู่ในสถานะ WAIT ให้ติดต่อกับหน่วยความจำ

3. โปรแกรมฮาร์ดแวร์ต่างๆที่ควบคุมการหยุดการทำงานตามที่ผู้ใช้กำหนด และยกเลิกสัญญาณ WAIT

4. คอนโทรลซีพียูรออ่านสัญญาณ BRK จากวงจรควบคุมการหยุดการทำงาน และรอการกดปุ่ม Esc จากผู้ใช้เมื่อพบสัญญาณเหล่านี้จะหยุดการทำงานโดยสร้างสัญญาณ WAIT ขึ้นอีกครั้งหนึ่ง ถ้าเป็นสัญญาณ BRK ที่เกิดจากการกำหนดจุดหยุดโปรแกรมจะอ่านว่าผ่านจุดหยุดครบจำนวนครั้งที่ต้องการหรือยังถ้ายังจะปล่อยให้ทำงานต่อไป

สถานะการหยุดมีดังนี้

USER BREAK

BREAK WHEN HALT

ACCESS NON-EXISTENT

FETCH OPCODE FROM DATA MEMORY

WRITE TO READ ONLY MEMORY

BREAK EXTERNAL

BREAK POINT

<u>คำสั่งเกี่ยวกับการติดตามการทำงานของซีพียูในเวลาจริง</u>

Trace Realtime TR B;C;F

คำสั่งเกี่ยวกับการติดตามการทำงานในเวลาจริง มี 3 คำสั่ง คือ

Trace Backward TRB

เก็บข้อมูลตั้งแต่เริ่มทำงานจนถึงหยุดทำงาน ถ้าข้อมูลเต็มจะเก็บทับข้อมูลในช่วงแรก ข้อมูลที่ได้ คือการติดตามการทำงานในช่วง 2048 แมชชีนไซเคิลสุดท้ายก่อนจะหยุดตามจุดหยุดที่ผู้ใช้ตั้งไว้

การทำงาน

1. ตั้งวงจรนับที่กำหนดแอดเดรสของบัฟเฟอร์ให้เป็น 0 กำหนดให้สัญญาณ ENWRBF เป็น 0 กำหนดจุดหยุดต่างๆเหมือนการทำงานในคำสั่ง G แล้วเริ่มทำงานโดยยกเลิก WAIT

2. รอตรวจสัญญาณ BRK เหมือนในคำสั่ง G เมื่อพบแล้วสร้างสัญญาณ WAIT และอ่านแอดเดรสของบัฟเฟอร์ในขณะนั้นเพื่อบอกให้ผู้ใช้ทราบจุดหยุดที่แท้จริง

3. ถ้าแมชชีนไซเคิลที่หยุดไม่ใช่ไซเคิลแรกของคำสั่งให้ยกเลิกสัญญาณ WAIT ไปยังแมชชีนไซเคิลต่อไปจนพบแมชชีนไซเคิลแรกของคำสั่ง

4. อ่านแอดเดรสของบัฟเฟอร์เพื่อหาตำแหน่งแรกที่จะแสดงผล ถ้าไม่มีสัญญาณ OVF ตำแหน่งแรกคือ 0 ถ้ามีสัญญาณ OVF แสดงว่ามีการเขียนข้อมูลใหม่ทับข้อมูลเก่าตำแหน่งแรกคือแอดเดรสของบัฟเฟอร์ที่อยู่ถัดจากตำแหน่งสุดท้ายที่อ่านได้

Trace Center TRC

เก็บข้อมูลตั้งแต่เริ่มทำงานจนพบจุดหยุดแล้วเก็บข้อมูลต่ออีก 1024 แมชชีนไซเคิล จึงหยุดการทำงาน ถ้าข้อมูลเต็มจะเก็บทับข้อมูลในช่วงแรก ข้อมูลที่ได้คือการติดตามการทำงานในช่วง 2048 แมชชีนไซเคิลสุดท้าย

การทำงาน

1. ตั้งวงจรนับที่ 0 ให้สัญญาณ ENWRBF เป็น 0 กำหนดจุดหยุดต่างๆเหมือนคำสั่ง G แล้วเริ่มทำงาน

2. รอตรวจสัญญาณ BRK เมื่อพบแล้วอ่านแอดเดรสของบัฟเฟอร์ขณะนั้นเพื่อบอกผู้ใช้ให้ทราบสาเหตุการหยุด

3. ยกเลิกจุดหยุดที่ผู้ใช้ตั้งไว้แล้วให้สัญญาณ ENBDLY เป็น 0 แล้วทำงานต่อ

4. รอสัญญาณ BRK ซึ่งจะเกิดจากวงจรนับที่ทำให้เกิดการหยุดที่ 1024 ไซเคิลหลังจากพบจุดหยุดของผู้ใช้

5. อ่านแอดเดรสของบัฟเฟอร์เพื่อหาตำแหน่งแรกที่จะแสดงผลเหมือนคำสั่ง TRB

Trace Forward TRF

ทำงานตามปรกติโดยไม่มีการบันทึกจนพบจุดหยุดจึงเริ่มบันทึกแล้วทำงานต่อจนบัฟเฟอร์เต็มแล้วหยุดการทำงาน ข้อมูลที่ได้ คือ 2048 ไซเคิล หลังพบจุดหยุด

การทำงาน

1. ถ้าผู้ใช้ไม่ตั้งจุดหยุดจะเริ่มเก็บข้อมูลตั้งแต่เริ่มทำงาน ถ้าผู้ใช้ตั้งจุดหยุดให้ทำงานเหมือนคำสั่ง G จนพบจุดหยุดจึงเริ่มเก็บข้อมูล ยกเลิกจุดหยุดของผู้ใช้

2. เริ่มเก็บข้อมูล ตั้งวงจรนับที่ 7 สำหรับ Z-80 หรือจำนวนแมชชีนไซเคิลสูงสุดของคำสั่งของซีพียู + 1

3. รอตรวจสัญญาณ BRK ซึ่งจะเกิดจากสัญญาณ BFUL บนแผงวงจรติดตามการทำงานในเวลาจริงเมื่อพบแล้วอ่านแอดเดรสของบัฟเฟอร์เป็นตำแหน่งสุดท้ายที่จะแสดงผลโดยตำแหน่งแรกคือ 7 สำหรับ Z-80

Trace History TH [range[range[F][R][W][I][O][A]]]

คำสั่งแสดงข้อมูลที่บันทึกในบัฟเฟอร์ เป็นการทำงานในโปรแกรมของผู้ใช้ทีละแมชชีนไซเคิล แสดงในรูปแอดเดรส สถานะและข้อมูล สำหรับแมชชีนไซเคิลสุดท้ายของคำสั่งจะแสดงคำสั่งเป็นภาษาแอสเซมบลีด้วย

ข้อมูลที่บันทึกในบัฟเฟอร์มีขนาด 2048 แมชชีนไซเคิล ไม่สะดวกในการหาจุดที่ต้องการ จึงมีการกำหนด range จะแสดงเฉพาะช่วงแอดเดรสของบัฟเฟอร์ที่กำหนดในช่วง 2 กิโลไบต์ เช่น 0 ถึง 7FF คือข้อมูลที่บันทึกในบัฟเฟอร์ทั้งหมด ถ้ากำหนดเป็น 0 ถึง 7 ก็จะเป็นช่วง 8 แมชชีนไซเคิลแรก

นอกจากนี้ยังมีการกำหนดแอดเดรสและสถานะของแมชชีนไซเคิลให้แสดงเฉพาะช่วงแอดเดรสและสถานะที่สนใจ ในช่วงแอดเดรสของบัฟเฟอร์ที่กำหนด

การทำงาน

1. หาพารามิเตอร์

2. ถ้าไม่มีพารามิเตอร์จะแสดงข้อมูลทั้งหมดในบัฟเฟอร์ในรูปแมชชีนไซเคิลและมีการแปลเป็นภาษาแอสเซมบลีในทุกคำสั่ง

3. ถ้ามีพารามิเตอร์เป็น range ครั้งเดียวจะแสดงเฉพาะตำแหน่งที่ระบุใน range เท่านั้น

4. ถ้ามีพารามิเตอร์เป็น range 2 ครั้ง จะแสดงเฉพาะตำแน่งที่ระบุใน range แรก และแอดเดรสที่แสดงต้องอยู่ในช่วง range ที่ 2 ด้วย ถ้าพารามิเตอร์กำหนดสถานะจะแสดงเฉพาะสถานะนั้น

5. การคำนวณตำแหน่งของแอดเดรสของบัฟเฟอร์จริงๆเทียบกับที่ผู้ใช้ระบุมาจะเป็นดังนี้

ตำแหน่งจริง = (ตำแหน่งที่ผู้ใช้ต้องการ + ตำแหน่งเริ่มต้น) AND 7FFH