คู่มือใช้งาน SIM68

สงวนลิขสิทธิ์ ฐิต ศิริบูรณ์

สารบัญ

บทนำ

โปรแกรม SIM68 เป็นโปรแกรมจำลองการทำงานของไมโครโปรเซสเซอร์ Mortorolla 6800 ในระดับคำสั่ง เพื่อประโยชน์ในการศึกษาและพัฒนาระบบที่ใช้ไมโครโปรเซสเซอร์ 6800 โดยใช้ทดสอบการทำงานของโปรแกรมแอสเซมปลีของ 6800

SIM68 โดยจะอ่าน Hex file ที่เป็น "ภาษาเครื่อง" ของ 6800 เข้ามา และทำงานได้ตามที่คำสั่งเหล่านั้นกำหนด แสดงให้เห็นการเปลี่ยนแปลงของ register ที่อยู่ภายในไมโครโปรเซสเซอร์ และการเปลี่ยนแปลงที่เกิดขึ้นกับหน่วยความจำขนาด 64K byte ที่ 6800 สามารถใช้ได้โดยตรง และผู้ใช้สามารถตรวจสอบการทำงานของ โปรแกรมแอสเซมปลีของ 6800 ได้โดยการตั้ง Breakpoint หรือ single step ที่ละคำสั่ง หรือ กำหนดค่าใน register (Watch)

นอกจากนี้ SIM68 ยังมีให้ผู้ใช้เรียกใช้ subroutine ที่จำลองการทำงานของส่วน I/O ได้เช่น อ่านอักขระจากแป้นพิมพ์ หรือ แสดงอักขระออกมาที่จอภาพ

การเรียกใช้ SIM68

เพื่อให้โปรแกรมจำลองการทำงานระดับคำสั่งของไมโครโปรเซสเซอร์ที่จะถูกพัฒนาขึ้นนี้สามารถใช้ได้อย่างแพร่หลาย จึงตั้งเป้าหมายให้โปรแกรมที่จะพัฒนาขึ้นสามารถวิ่งบนคอมพิวเตอร์ส่วนบุคคลแบบ IBM ที่ทำงานภายใต้โปรแกรมจัดการ (operating system) MS. DOS รุ่น 3.3 หรือ สูงกว่า มีหน่วยความจำไม่น้อยกว่า 640KB มี floppy disk drive ไม่น้อยกว่า 1 drive และ ทำงานใน text mode ซึ่งข้อกำหนดเหล่านี้เพื่อให้คอมพิวเตอร์ส่วนบุคคลที่อยู่ทั่วไปหรือ แม้แต่เครื่องรุ่นเก่าก็สามารถใช้ได้

ก่อนที่จะเรียกใช้ SIM68 จะต้องมี Hex File ของ 6800 ที่ต้องการเสียก่อน Hex file นี้สร้างโดยใช้ Assembler เปลี่ยน source code program ของโปรแกรมแอสเซมปลีของ 6800[1] และ Hex file จะต้องเป็นแบบ Mortorolla S-Format

การเรียกใช้ SIM68 ทำโดย ที่ DOS prompt พิมพ์

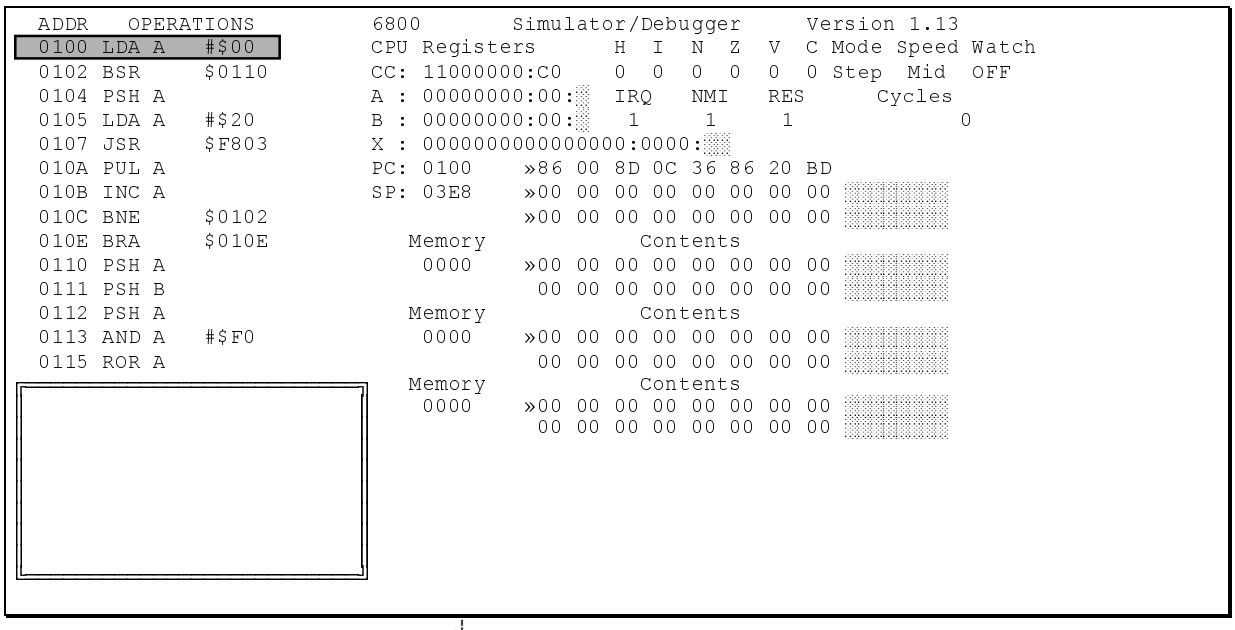
>SIM68 *ชื่อHex file* <enter>

---

[1] เพื่อให้ไม่เกิดความสับสนระหว่างโปรแกรมที่วิ่งบนไมโครโปรเซสเซอร์ กับ โปรแกรมจำลองการทำงานระดับคำสั่งของไมโครโปรเซสเซอร์ (SIM68) จะใช้คำว่า "ชุดคำสั่ง" เพื่อหมายถึง โปรแกรมที่วิ่งบนไมโครโปรเซสเซอร์ และ "โปรแกรม" เพื่อหมายถึง SIM68

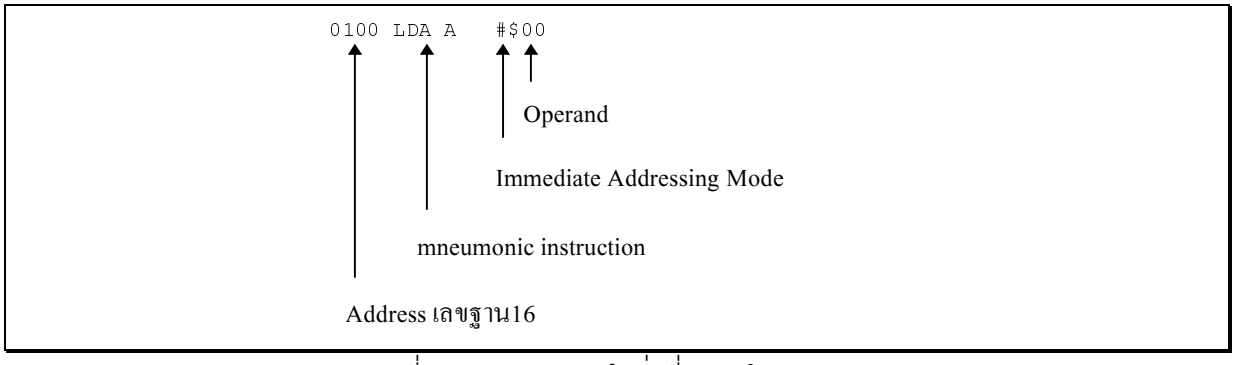
เมื่อโปรแกรมทำงานทางด้านซ้ายของจอภาพจะแสดงคำสั่งแอสแซมบลีซึ่งได้จากการ disassembler ชุดคำสั่งใน Hex file ส่วนทางด้านขวาของจอภาพจะแสดงค่าที่อยู่ใน register ต่างๆของ 6800 และ ค่าจาก memory ดังแสดงในรูปที่ 1.

จุดเริ่มต้นของชุดคำสั่งจะเป็น address แรกที่ปรากฏใน Hex file และจะมีแถบแสง (Highlight) ปรากฏอยู่เพื่อแสดงว่าเป็นคำสั่งที่จะทำงานเมื่อโปรแกรมจะ RUN ชุดคำสั่ง



รูปที่ 1. ตัวอย่างจอภาพของ SIM68

ทางด้านซ้ายของจอภาพจะแสดง Address ของแต่ละคำสั่งในรูปแบบเลขฐาน16 ตามด้วย Opcode ของคำสั่ง ในรูป mneumonic instruction และ Operand ของคำสั่งในรูปแบบเลขฐาน16 (แสดงด้วยเครื่องหมาย $) ถ้าคำสั่งเป็น immediate addressing mode จะมีเครื่องหมาย # นำหน้า Operand ดังตัวอย่างในรูปที่ 2.



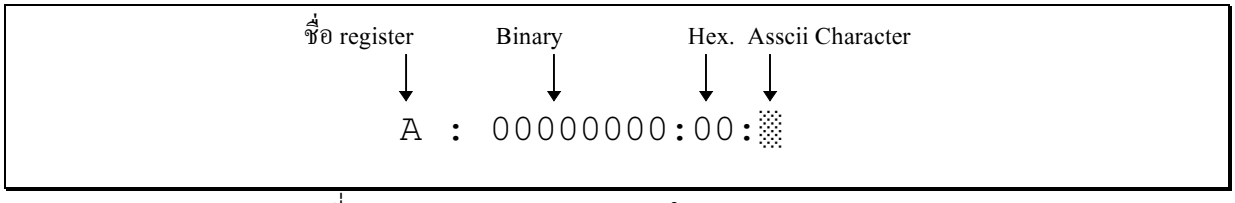
รูปที่ 2. ตัวอย่างของคำสั่งที่แสดงใน SIM68

สำหรับคำสั่งประเภท Branch ซึ่งใช้ Relative addressing mode ที่ตำแหน่งของ Operand จะแสดงค่า adddress ที่จะ branch ไปแทนที่จะเป็นค่า operand จริง ซึ่งเป็นค่า off set ทั้งนี้เพื่อจะได้สะดวกต่อ

การ “อ่าน” ชุดคำสั่ง ตัวอย่างเช่น ที่ address 0102 มีคำสั่ง BSR โดยมี operand คือ 0C ซึ่งตรงกับ address 0110 ใน SIM68 จะแสดงเป็น 0102 BSR $0110 ดังในรูปที่ 1.

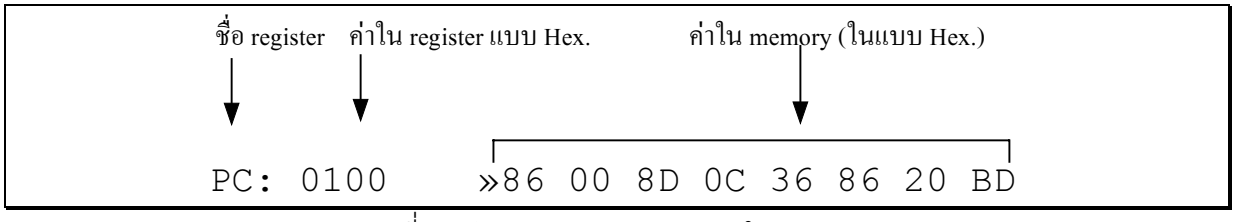
การแสดงค่าภายใน (content) ของ register

การแสดงค่าภายในของ Accumulator A (A), Accumulator B (B) และ Index register (X) จะแสดงรูป Binary number, Hexadecimal number (Hex.) และ Asscii character (ในกรณีที่เป็น non-printable character จะใช้ character ▒ แทน) ดังในรูปที่ 3.



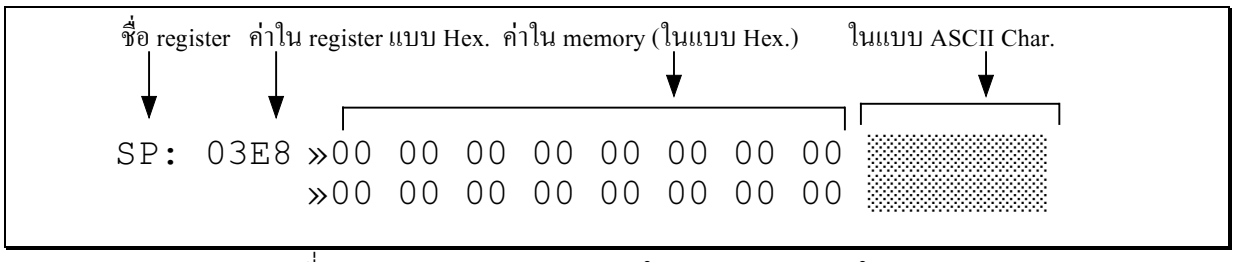
รูปที่ 3. ตัวอย่างการแสดงค่าภายในของ Accumulator A

การแสดงค่าภายในของ Program Counter register (PC) จะแสดงในรูปของ Hex. และ ค่าใน memory อีก 8 bytes (locations) ที่ถูก “ชี้” ด้วย PC ดังในรูปที่ 4.



รูปที่ 4. ตัวอย่างการแสดงค่าภายในของ PC

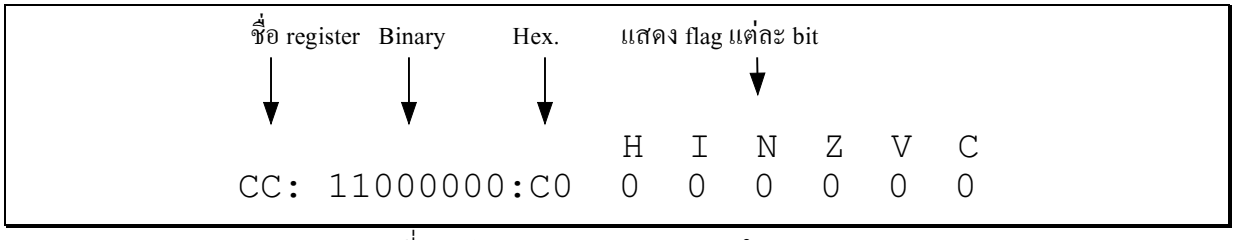
การแสดงค่าภายในของ Stack Pointer register (SP) จะแสดงในรูปของ Hex. และ ค่าใน Stack อีก 16 bytes (locations) ซึ่งคือค่าของ memory ที่ถูก “ชี้” ด้วย SP ค่าที่อยู่ภายในแต่ละ location ถูกแสดงในรูป Hex. และ Asscii character ดังในรูปที่ 5. โดย default[2] ค่า SP จะเป็น $03E8_{16}$



รูปที่ 5. ตัวอย่างการแสดงค่าภายในของ SP และ ค่าใน Stack

[2] ค่านี้ถูกเลือกมาเพื่อความสะดวกในการใช้งานของ SIM68

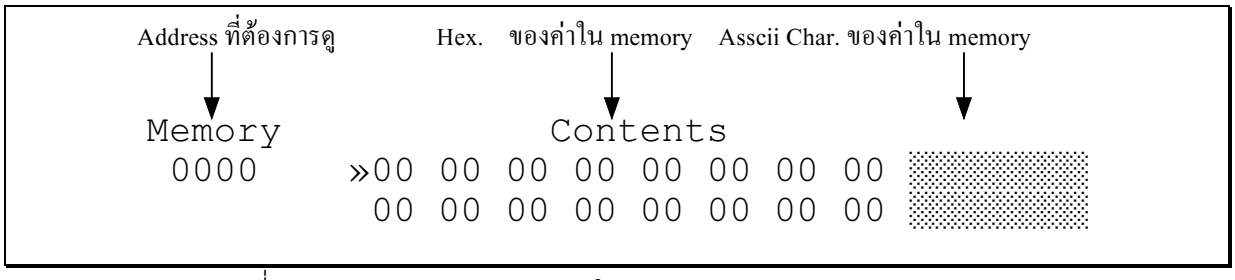
การแสดงค่าภายในของ Conditional Code register (CC), จะแสดงรูป Binary number, Hexadecimal number (Hex.) และ แยกแสดงแต่ละ bit ที่ตรงกับ flag แต่ละอัน ดังในรูปที่ 6.



รูปที่ 6. ตัวอย่างการแสดงค่าภายในของ CC

การแสดงค่าของ memory

ผู้ใช้ SIM68 สามารถจะดูค่าจาก memory ได้พร้อมกันถึง 3 บริเวณแต่ละบริเวณจะดูได้ 16 location ต่อเนื่องกันทั้งในแบบ Hexadecimal number (Hex.) และ แบบ Asscii character เมื่อผู้ใช้ต้องการจะดูค่าจากบริเวณใดทำได้โดยเปลี่ยนค่าของ address ของบริเวณนั้น ดังในรูปที่ 7.

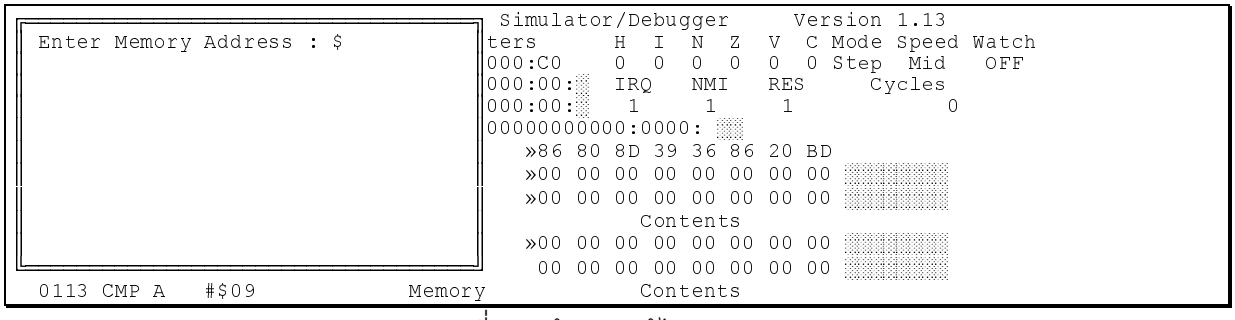


รูปที่ 7. ตัวอย่างการแสดงค่าภายในของ memory จาก address 0000

การเปลี่ยนแปลงค่าของ register และ memory

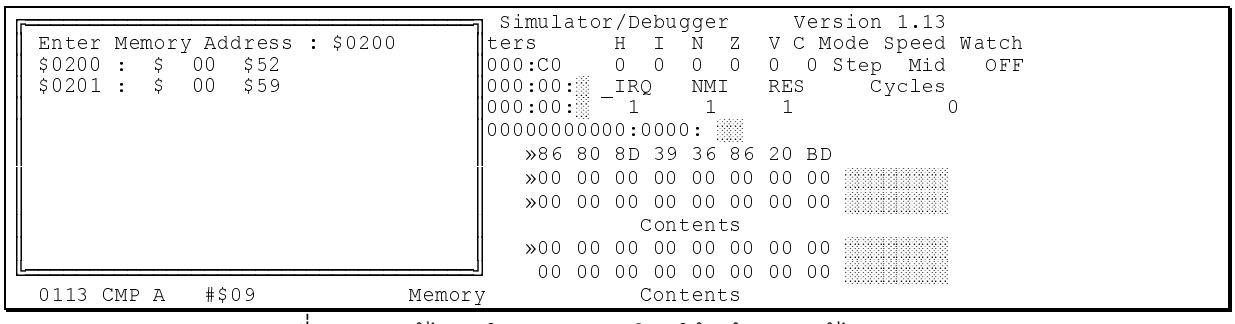
ผู้ใช้สามารถเปลี่ยนแปลงค่าของ register ต่างๆ และ memory ได้ด้วยการใช้ปุ่มเลื่อน cursor (ปุ่มลูกศร) เลื่อน cursor (เป็นกรอบสีแดงขนาด 1 character) ไปยังตำแหน่งที่ต้องการจะเปลี่ยนแปลง แล้วพิมพ์ค่าใหม่ที่ต้องการลงไป ซึ่งจะทำได้ทั้งใน ส่วน binary, hex. และ ASCII char.

นอกจากนี้ผู้ใช้ยังสามารถ ใช้หน้าต่างแก้ไข memory โดยกดแป้นพิมพ์ Alt+M เพื่อเรียกหน้าต่างแก้ไข memory ขึ้นมา ดังในรูปที่ 8.



รูปที่ 8. หน้าต่างแก้ไข memory

หลังจากที่หน้าต่างแก้ไข memory ปรากฏขึ้น ใส่ address ที่ต้องการจะแก้ไขในรูปแบบเลขฐาน 16 แล้วกด <enter> address และค่าที่อยู่ใน memory จะแสดงออกมา พิมพ์ค่าใหม่ที่ต้องการหลังเครื่องหมาย $ แล้วกด <enter> address จะเพิ่มขึ้นอีกหนึ่งและค่าที่อยู่ใน memory จะแสดงออกมา ในบรรทัดใหม่พิมพ์ค่าใหม่ที่ต้องการหลังเครื่องหมาย $ แล้วกด <enter> อย่างนี้ไปเรื่อยๆ ดังแสดงในรูป 9. เมื่อสิ้นสุดการแก้ไข กด <ESC>



รูปที่ 9. การแก้ไขค่าใน memory โดยใช้หน้าต่างแก้ไข memory

การจำลองสัญญาณ Hardware Interrupt

ไมโครโปรเซสเซอร์ 6800 มี Hardware Interrupt 3 อันคือ

Reset (RES) , Maskable Interrupt (IRQ) และ Non-maskable Interrupt (NMI) ผู้ใช้สามารถจำลองสัญญาณ เหล่านี้ได้ โดยการเลื่อน cursor ไปยัง bit ของแต่ละ Interrupt ดังในรูปที่ 10. แล้วเปลี่ยนค่า bit ของ Interrupt ที่ต้องการให้เป็น 0

รูปที่ 10. Bit จำลองสัญญาณ Hardware Interrupt ทั้ง 3 อัน

หรืออาจจะใช้แป้นพิมพ์ แทนการเลื่อน cursor แล้วเปลี่ยนค่า bit ของ Interrupt แป้นพิมพ์ที่ใช้แทนคือ

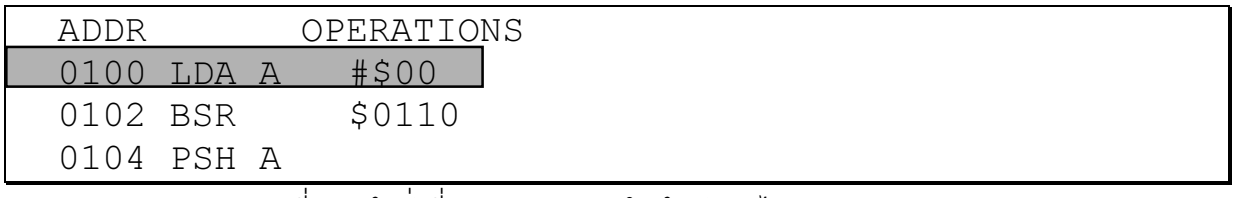
Shift F8 แทน IRQ

Shift F9 แทน NMI

Shift F10 แทน Reset

การ RUN ชุดคำสั่งของ 6800

ในส่วนที่แสดงชุดคำสั่ง จะมีคำสั่งหนึ่งที่ถูก highlight ดังรูปที่ 11. ซึ่งเป็นการแสดงถึงคำสั่งที่จะถูก execute ในลำดับต่อไป



รูปที่ 11. คำสั่งที่จะถูก execute ในลำดับต่อไปจะถูก Highlight

ค่าต่างๆ ของ register และ memory ที่ปรากฏบนหน้าจอของโปรแกรม SIM68 จะเป็นค่าที่เกิดขึ้น **ก่อน**ที่จะ execute คำสั่งที่ถูก highlight

ผู้ใช้สามารถจะ RUN ชุดคำสั่งได้ 2 วิธี

วิธีที่ 1. Execute ที่ละคำสั่ง (Single step execution) ผู้ใช้จะกดปุ่มบนแป้นพิมพ์ ทุกครั้งที่กดปุ่มคำสั่งที่ถูก highlight จะถูก execute และ ค่าต่างๆจะมีการเปลี่ยนแปลง พร้อมกับ highlight จะถูกเลื่อนไปยังคำสั่งในลำดับถัดไป ซึ่งทำได้โดยกดแป้นพิมพ์ F2

วิธีที่ 2. Execute อย่างต่อเนื่อง (RUN) เมื่อผู้ใช้สั่ง RUN โปรแกรม SIM68 จะทำชุดคำสั่งของ 6800 อย่างต่อเนื่องโดยผู้ใช้ไม่ต้องคอยกดปุ่ม single step และ ค่าต่างๆ จะถูก Update ตลอดเวลา ซึ่งทำได้โดยกดแป้นพิมพ์ F1
นอกจากนี้ผู้ใช้ยังสามารถเลือก “ความเร็ว” (Speed) ในการ RUN ได้ 3 ระดับคือ Fast (เร็วสุด) --> Med (ปานกลาง) --> Slow(ช้า) โดยการกดแป้นพิมพ์ F10 ซึ่งจะเปลี่ยนระดับความเร็วตามลำดับ

นอกจากนี้ผู้ใช้สามารถให้ SIM68 ทำงานถอยหลังที่ละคำสั่งได้ (Back Single Step) โดยใช้การกดแป้นพิมพ์ Shift F2 SIM68 จะแก้ไขค่าต่างๆทั้งใน register และ memory ที่ถูกเปลี่ยนแปลงให้กลับมาอยู่ในสภาพเดิมก่อน execute คำสั่งนั้น การย้อนหลังนี้ทำได้ 50 คำสั่ง

ระหว่างที่ SIM68 ทำงานใน mode RUN ผู้ใช้สามารถเปลี่ยนค่าต่างๆใน register หรือ ใน memory ได้ตลอดเวลาโดยไม่ต้องหยุดการ RUN ชุดคำสั่ง

SIM68 ทำงานใน mode RUN จนกว่าผู้ใช้จะเปลี่ยนไปอยู่ mode single step โดยการกดปุ่ม single step (F2)

การตั้ง Breakpoints

Breakpoint คือคำสั่งในชุดคำสั่งที่ถูกกำหนดให้เป็นจุด “หยุด” ก่อนที่จะถูก execute ในขณะที่ SIM68 กำลังอยู่ใน mode RUN สำหรับใน SIM68 มีการตั้ง breakpoint ได้ 2 แบบ คือ แบบครั้งเดียว (Non-Sticky) และแบบถาวร (Sticky)

Breakpoint แบบครั้งเดียว (Non-Sticky) คือ เมื่อ SIM68 หยุดที่คำสั่งที่มี breakpoint แบบครั้งเดียว SIM68 จะหยุดก่อนที่จะ execute คำสั่งนั้นแล้ว ลบ breakpoint นี้ทิ้งไป และถ้า SIM68 ทำงานมาถึงคำสั่งนี้อีกจะไม่หยุด

ส่วน breakpoint แบบถาวร (Sticky) เมื่อ SIM68 หยุดที่คำสั่งที่มี breakpoint แบบถาวร SIM68 จะหยุดก่อนที่จะ execute คำสั่งนั้น แต่ breakpoint ที่ตั้งไว้ยังคงอยู่ และถ้า SIM68 ทำงานมาถึงคำสั่งนี้อีกก็จะหยุดที่คำสั่งนี้อีก

การตั้ง breakpoint ทำได้ 2 วิธีคือ

วิธีที่ 1. การตั้ง breakpoint สำหรับคำสั่งที่แสดงอยู่ที่จอภาพ ใช้แป้นพิมพ์ F3 สำหรับเลื่อนเครื่องหมายกำหนดตำแหน่ง breakpoint (>) ขึ้น และใช้แป้นพิมพ์ F4 สำหรับเลื่อนเครื่องหมายกำหนดตำแหน่ง breakpoint (>) ลง ดังในรูปที่ 12.

```
 0102 BSR    $0110
>0104 PSH A
 0105 LDA A  #$20
 0107 JSR    $F803
```

รูปที่ 12. เครื่องหมายกำหนดตำแหน่ง breakpoint

เมื่อเครื่องหมายกำหนดตำแหน่ง breakpoint “ชี้” ที่คำสั่งที่ต้องการจะตั้ง breakpoint ผู้ใช้สามารถจะตั้ง breakpoint ที่คำสั่งนั้นได้โดยกดแป้นพิมพ์ Shift+F3 สำหรับ breakpoint แบบถาวร หรือ กดแป้นพิมพ์ Shift+F4 สำหรับ breakpoint แบบครั้งเดียว เมื่อ breakpoint ถูก “ตั้ง” ที่ คำสั่งใดจะมีเครื่องหมาย breakpoint ปรากฏที่หน้าคำสั่งนั้นตามชนิดของ breakpoint ดังในรูปที่ 13. และรูปที่ 14.

```
 0102 BSR    $0110
▶0104 PSH A
 0105 LDA A  #$20
 0107 JSR    $F803
```

รูปที่ 13. สัญญลักษณ์ breakpoint แบบถาวร (Shift+F3)

```
 0102 BSR    $0110
```

```
»0104 PSH A
 0105 LDA A   #$20
 0107 JSR     $F803
```

รูปที่ 14. สัญญลักษณ์ breakpoint แบบครั้งเดียว (Shift+F4)

วิธีที่ 2. ใช้หน้าต่างตั้ง/แก้ไข Breakpoint โดยกดแป้นพิมพ์ Alt+B จะได้ ดังรูปที่ 15.

```
          Break Points Set Up
 BreakPoint # 1 at $xxxx No Breakpoint;
 BreakPoint # 2 at $xxxx No Breakpoint;
 BreakPoint # 3 at $xxxx No Breakpoint;
 BreakPoint # 4 at $xxxx No Breakpoint;
 BreakPoint # 5 at $xxxx No Breakpoint;
 BreakPoint # 6 at $xxxx No Breakpoint;
 BreakPoint # 7 at $xxxx No Breakpoint;
 BreakPoint # 8 at $xxxx No Breakpoint;
 BreakPoint # 9 at $xxxx No Breakpoint;
 BreakPoint #10 at $xxxx No Breakpoint;
 BreakPoint #11 at $xxxx No Breakpoint;
                          [MORE]
 Use ïî to move the highlight.
 Hit <CR> to select.
 Hit ESC to close the window.
```

รูปที่ 15. หน้าต่างตั้ง/แก้ไข breakpoint

หน้าต่างตั้ง/แก้ไข breakpoint นี้จะสามารถใช้ตั้ง/แก้ไข breakpoint ที่คำสั่งใดๆได้ โดยคำสั่งเหล่านั้นไม่จำเป็นต้องถูกแสดงอยู่ที่จอภาพในขณะนั้น ผู้ใช้สามารถตั้ง breakpoint ได้ถึง 25 แห่ง โดยกดแป้นลูกศรชี้ลง เพื่อเลื่อนแถบแสงยัง breakpoint ที่ต้องการจะตั้งหรือแก้ไข แล้วกด <enter>[3] หน้าต่างจะเปลี่ยนแปลงเป็นในรูปที่ 16.

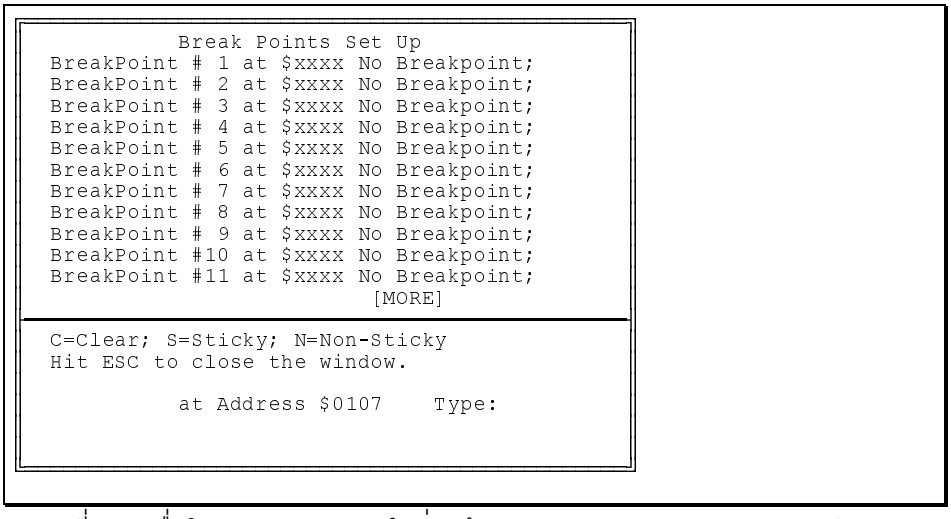
```
          Break Points Set Up
 BreakPoint # 1 at $xxxx No Breakpoint;
 BreakPoint # 2 at $xxxx No Breakpoint;
 BreakPoint # 3 at $xxxx No Breakpoint;
 BreakPoint # 4 at $xxxx No Breakpoint;
 BreakPoint # 5 at $xxxx No Breakpoint;
 BreakPoint # 6 at $xxxx No Breakpoint;
 BreakPoint # 7 at $xxxx No Breakpoint;
 BreakPoint # 8 at $xxxx No Breakpoint;
 BreakPoint # 9 at $xxxx No Breakpoint;
 BreakPoint #10 at $xxxx No Breakpoint;
 BreakPoint #11 at $xxxx No Breakpoint;
                          [MORE]
 Enter the Address for breakpoint # 1.
 Hit ESC to close the window.

          at Address $
```

รูปที่ 16. เมื่อกด <enter> จะรอให้ผู้ใช้ใส่ address ของคำสั่ง

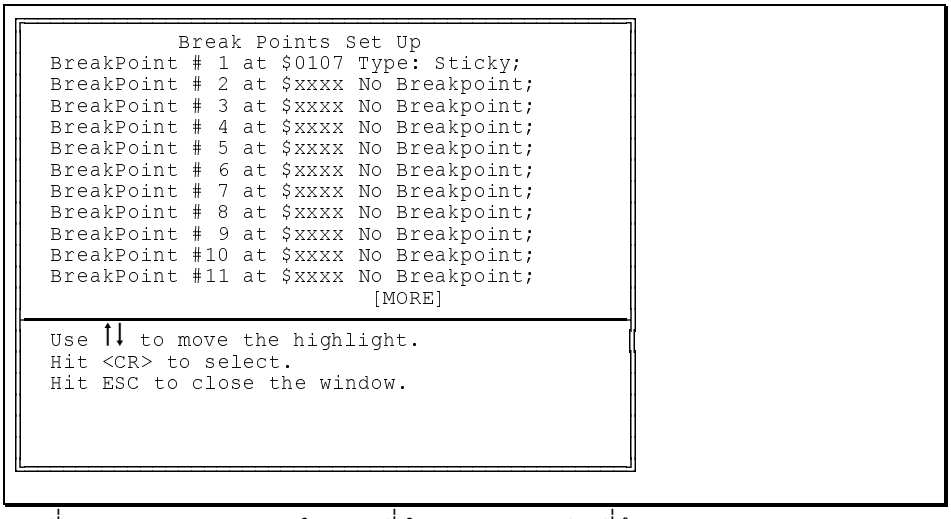
[3] ในหน้าต่างจะใช้ <CR> แทน <enter>

ผู้ใช้ใส่ address ของคำสั่งที่ตั้ง breakpoint เป็นเลขฐาน 16 แล้วกด <enter> เพื่อกำหนดชนิดของ brealpoint ดังรูปที่ 17.



รูปที่ 17. เมื่อใส่ address ของคำสั่งแล้วกด <enter> SIM68 จะถามชนิดของ breakpoint

ผู้ใช้เลือกชนิดของ breakpoint ได้ 3 ชนิด คือ Clear (ลบ breakpoint กด C) Sticky (ถาวร กด S) และ Non-Sticky (ครั้งเดียว กด N) breakpoint หมายเลขที่กำหนด จะแสดงให้เห็นตำแหน่งและชนิดของ breakpoint ดังในรูปที่ 18. เมื่อสิ้นสุดการตั้ง/แก้ไข กด <ESC> เพื่อปิดหน้าต่าง



รูปที่ 18. Breakpoint ถูกกำหนดที่ตำแหน่งและชนิดที่ต้องการ

การใช้ Watch

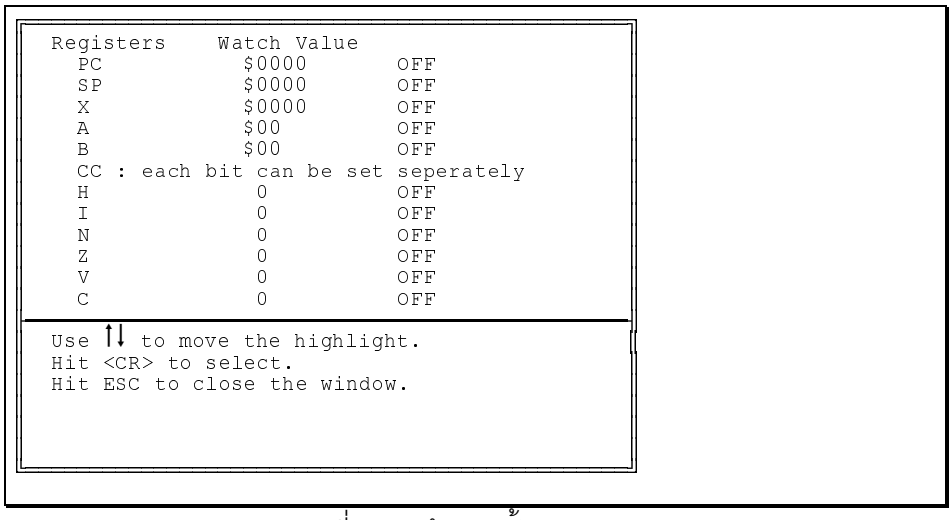
Watch คือ การที่ SIM68 คอยตรวจสอบว่า register มีค่าตรงกับค่าที่ต้องการเมื่อไหร่ และเมื่อ register มีค่าตรงกับค่าที่กำหนด SIM68 จะหยุดที่คำสั่งหลังคำสั่งที่สร้างค่าที่กำหนดให้ Watch

ตัวอย่างเช่น ถ้าผู้ใช้กำหนดค่า Watch ไว้ที่ Accumulator A ให้เป็น $3C_{16}$ ชุดคำสั่งจะหยุดทำงานเมื่อค่าใน Accumulator A เป็น $3C_{16}$

SIM68 สามารถกำหนด Watch ให้กับ X register PC[4] SP Accumulator A Accumulator B และ แต่ละ bit ของ CC

ขั้นตอนของการใช้ Watch

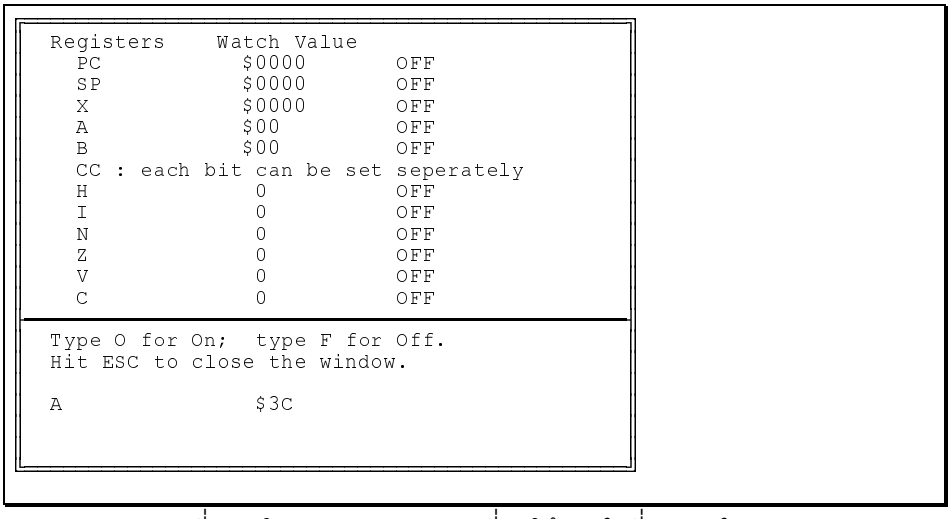
ขั้นที่ 1 กำหนดค่าและ register ที่ต้องการให้ตรวจสอบโดยกดแป้น Shift+F5 เพื่อเรียกหน้าต่างตั้งค่า Watch ออกมาดังในรูป 19.



รูปที่ 19. หน้าต่างตั้งค่า Watch

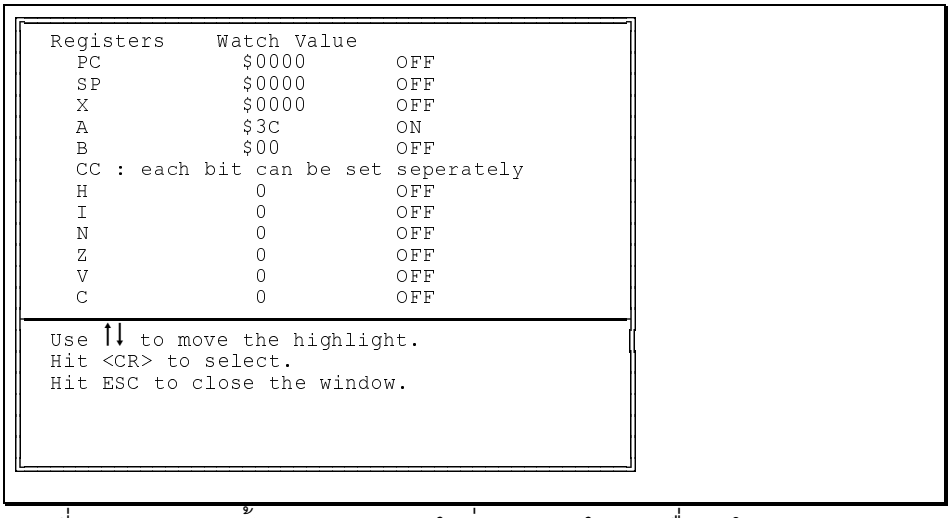
แล้วใช้ปุ่มลูกศรขึ้นลงเลื่อนแถบแสงไปที่ register ที่ต้องการกำหนดค่า Watch แล้วกด <enter> พิมพ์ค่าที่ต้องการ ในรูปแบบเลขฐาน 16 ลงไปแล้วกด <enter> ดังรูปที่ 20.

[4] การตั้ง Watch ให้กับ PC จะได้ผลคล้ายกับตั้ง Brealpoint



รูปที่ 20. ใส่ค่าของ register ที่จะให้ชุดคำสั่งหยุดทำงาน

จากนั้นเลือกกด แป้นพิมพ์ O (เพื่อใช้ค่า Watch) หรือ แป้นพิมพ์ F (เพื่อเลิกใช้ค่า Watch) ดังในรูปที่ เมื่อใช้หน้าต่างเสร็จกด <ESC> แล้วทำขั้นที่ 2



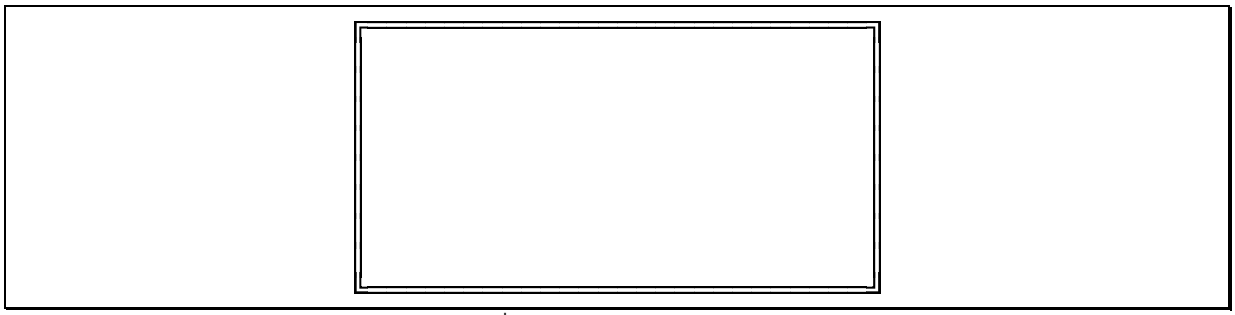
รูปที่ 21. แสดงการตั้งค่า Watch ชุดคำสั่งจะหยุดทำงานเมื่อค่าใน Accumulator $A=3C_{16}$

ขั้นที่ 2. หลังจากกำหนดค่าและ register ที่ต้องการแล้ว กดแป้นพิมพ์ F5 เพื่อให้ SIM68 คอยตรวจสอบ register (Watch) โดยที่ใต้คำว่า Wacth จะเป็น ON ถ้าไม่ต้องการให้ SIM68 คอยตรวจสอบ register กดแป้นพิมพ์ F5 อีกครั้ง ใต้คำว่า Wacth จะเป็น OFF เมื่อ Watch ON การทำงานของ SIM68 จะช้ากว่าตอน Watch OFF

การใช้ Subroutine จำลอง

เพื่อให้ผู้ใช้ได้ทดลองการทำงานที่เชื่อมต่อกับ Input/Output SIM68 มี “subroutine จำลอง” ที่ทำงานพิเศษต่างๆ ซึ่งผู้ใช้เพียงแต่เขียนคำสั่ง JSR หรือ BSR ไปที่ address ของ “subroutine จำลอง” เหล่านี้

InChar (F800) เมื่อ SIM68 ทำงานมาถึงคำสั่ง JSR F800 หรือ BSR F800 SIM68 จะหยุดการทำงานรอให้ผู้ใช้กดแป้นพิมพ์ และเมื่อผู้ใช้กดแป้นพิมพ์ค่าระหัส ASCII ของอักษรตัวนั้นจะถูกเก็บไว้ใน Acc. A และตัวอักษรนั้นจะถูกแสดงออกมาที่ Simulator Screen ของ SIM68 (รูปที่ 22.) ซึ่งอยู่ด้านล่างซ้ายมือของจอภาพ



รูปที่ 22. Simulator Screen

InCharNoEcho (F801) เมื่อ SIM68 ทำงานมาถึงคำสั่ง JSR F801 หรือ BSR F801 SIM68 จะหยุดการทำงานรอให้ผู้ใช้กดแป้นพิมพ์ และเมื่อผู้ใช้กดแป้นพิมพ์ค่าระหัส ASCII ของอักษรตัวนั้นจะถูกเก็บไว้ใน Acc. A แต่ตัวอักษรนั้นจะไม่ถูกแสดงออกมาที่ Simulator Screen ของ SIM68 (No echo)

OutCh (F803) เมื่อ SIM68 ทำงานมาถึงคำสั่ง JSR F803 หรือ BSR F803 SIM68 จะแสดงอักษรตามระหัส ASCII ที่อยู่ใน Acc. A ออกมาที่ Simulator Screen

RandomJsr (F80A) เมื่อ SIM68 ทำงานมาถึงคำสั่ง JSR F80A หรือ BSR F80A SIM68 จะสร้างเลขสุ่ม (random number) ระหว่าง 0-255 ขึ้น แล้วเก็บลงใน Acc. A

คำสั่งอื่นๆของ SIM68

ออกจาก SIM68 กดแป้นพิมพ์ Alt+Q ถ้าต้องการออกจาก SIM68 กดแป้นพิมพ์ Y (yes) จะกลับมาที่ DOS

Reset การทำงานของ SIM68 กดแป้นพิมพ์ Alt+R SIM68 จะ clear memory และ register ต่างๆ แล้ว reload ชุดคำสั่งเดิม

Load ชุดคำสั่งใหม่ กดแป้นพิมพ์ Alt+L SIM68 จะ clear memory และ register ต่างๆ แล้ว จะถามชื่อ File (Hex format) ของชุดคำสั่งใหม่ที่ต้องการจะ load

ตารางสรุปคำสั่งของ SIM68

	ไม่กด Shift	กด Shift
F1	RUN (ทำงานต่อเนื่อง)	*ไม่ได้ใช้*
F2	ทำงานที่ละคำสั่ง (Single Step)	ทำงานถอยหลังที่ละคำสั่ง (Back Single Step)
F3	เลื่อนตัวชี้ Breakpoint ขึ้น	ตั้ง Breakpoint แบบถาวร (Sticky)
F4	เลื่อนตัวชี้ Breakpoint ลง	ตั้ง Breakpoint แบบครั้งเดียว (Non-Sticky)
F5	ใช้/ยกเลิก การตรวจสอบค่า (Watch)	เปิดหน้าต่างกำหนดค่าและ register ที่จะตรวจสอบ
F6	*ไม่ได้ใช้*	*ไม่ได้ใช้*
F7	*ไม่ได้ใช้*	*ไม่ได้ใช้*
F8	*ไม่ได้ใช้*	ส่งสัญญาณ IRQ
F9	*ไม่ได้ใช้*	ส่งสัญญาณ NMI
F10	เปลี่ยนแปลงความเร็วในการ RUN	ส่งสัญญาณ Reset

แป้นพิมพ์	การทำงาน
Alt+B	เปิดหน้าต่างสำหรับกำหนดค่า Breakpoint
Alt+L	Load ชุดคำสั่งใหม่
Alt+M	เปิดหน้าต่างสำหรับแก้ไขค่าใน Memory
Alt+Q	สิ้นสุดการทำงานของ SIM68
Alt+R	Reset การทำงานของ SIM68
Alt+F10	แสดงผลในแบบ Monochorm