บทที่ 4

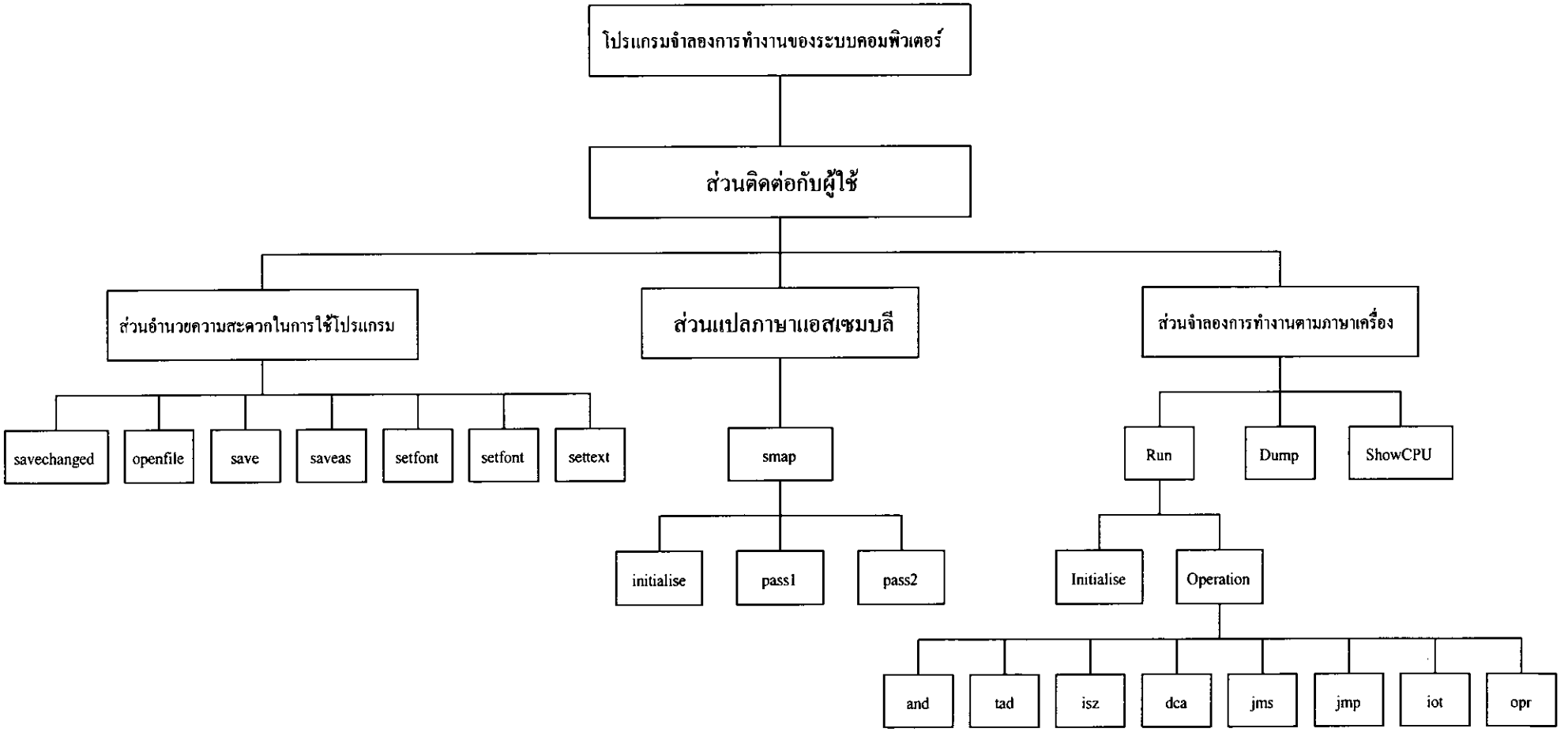
การพัฒนาระบบ

โปรแกรมที่พัฒนาขึ้นเป็นโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์ โดยโปรแกรมประกอบด้วยส่วนหลัก ๆ คือ ส่วนแปลโปรแกรมภาษาแอสเซมบลี ส่วนจำลองการทำงาน และส่วนติดต่อกับผู้ใช้

โปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์จะต้องมีส่วนสำหรับการติดต่อใช้งานโปรแกรมจากผู้ใช้ในรูปแบบของเมนู คือ ส่วนของการป้อนโปรแกรมภาษาแอสเซมบลี หรือการเปิดจากแฟ้มข้อมูลที่ได้สร้างมาก่อนแล้ว ส่วนการดำเนินงานแปลจากภาษาแอสเซมบลีเป็นภาษาเครื่อง และส่วนจำลองการทำงานในการประมวลผลข้อมูล ดังนั้นการดำเนินงานของระบบส่วนใหญ่จะเป็นการติดต่อกับผู้ใช้เพื่อดำเนินงานกับคำสั่งที่รับจากผู้ใช้ในรูปแบบของกราฟิก โปรแกรมทั้งหมดถูกพัฒนาโดยภาษาจาวา

4.1 โครงสร้างการดำเนินงาน

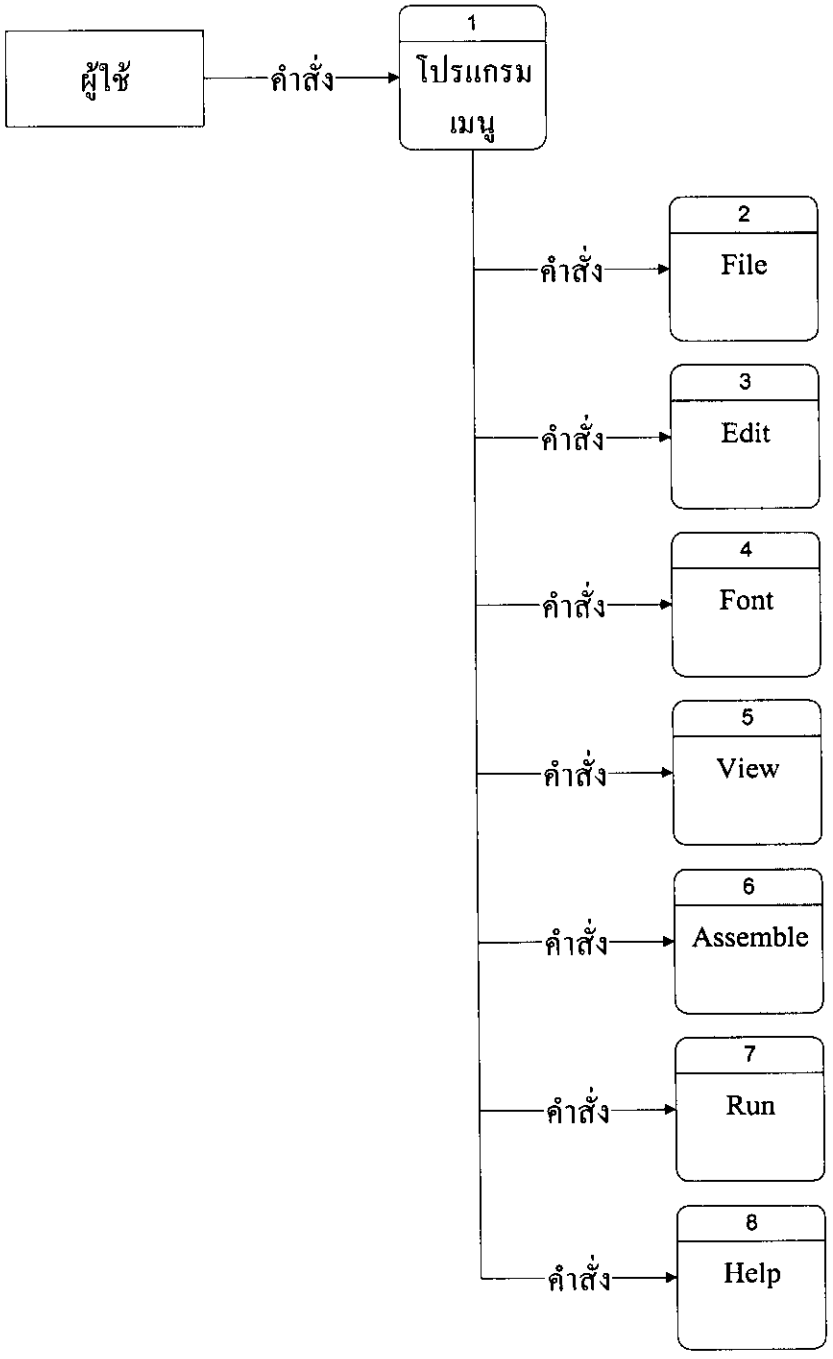
จากแนวคิดในการออกแบบโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์ในบทที่ 3 ได้แบ่งส่วนต่าง ๆ ของโปรแกรมออกเป็นส่วน ๆ โดยแต่ละส่วนแบ่งการทำงานย่อย ๆ ออกเป็นมอดูล (module) แต่ละมอดูลจะทำงานเฉพาะอย่าง และนำมอดูลทั้งหมดมาเชื่อมโยงให้ทำงานร่วมกัน ดังนั้นการทำงานของโปรแกรมทั้งหมดจะประกอบด้วยมอดูลเป็นจำนวนมาก ซึ่งสามารถแสดงกลุ่มของมอดูลในแต่ละส่วนการทำงานในโปรแกรมตามหน้าที่การทำงานด้วย Program structured chart ดังแสดงในภาพประกอบ 4.1



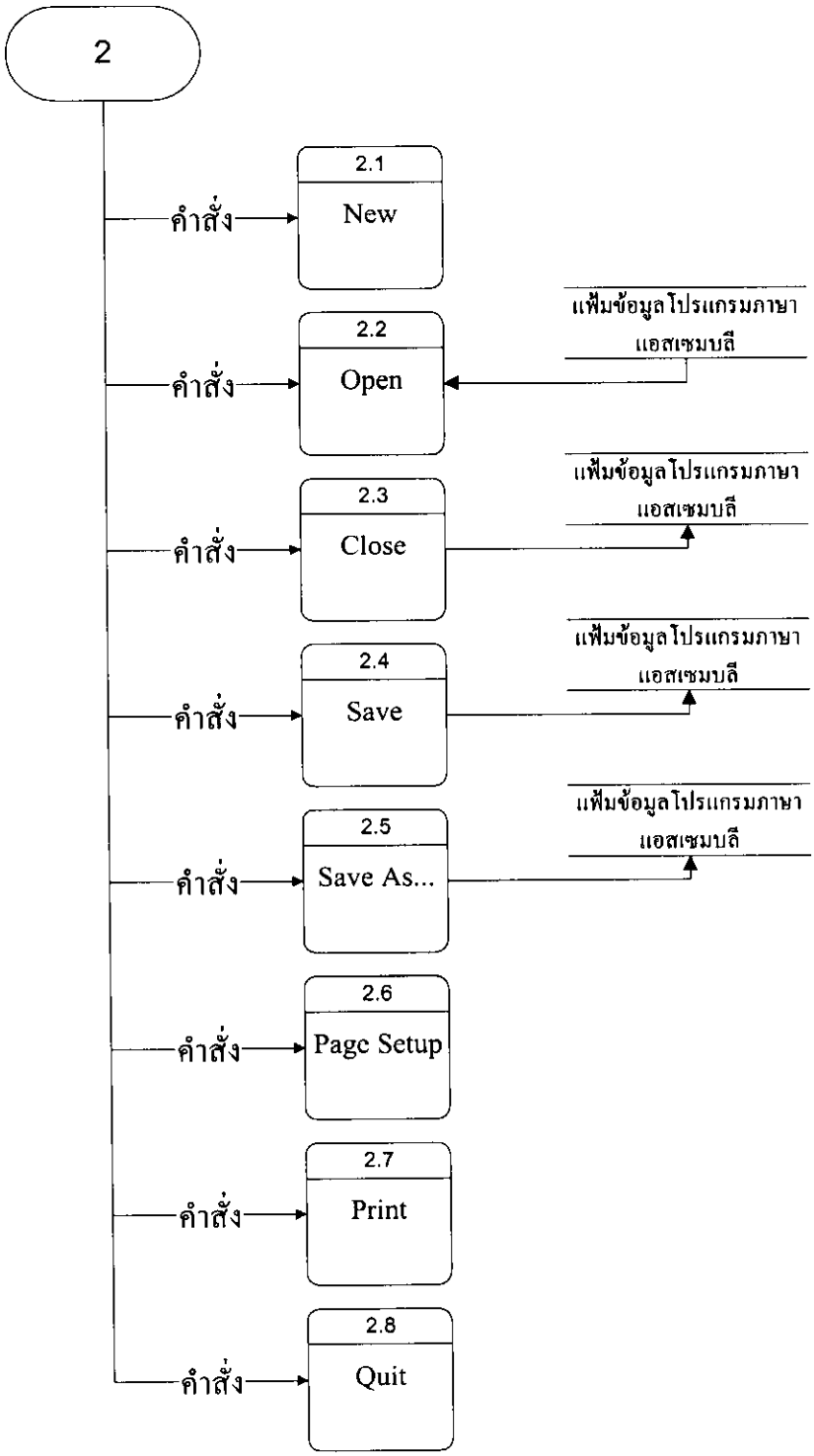
ภาพประกอบ 4.1 Program structured chart

4.2 กระบวนการดำเนินงาน

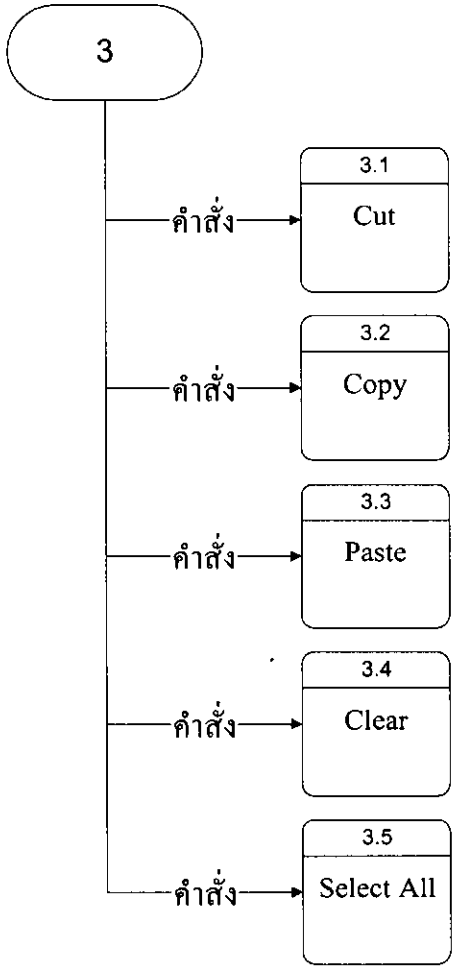
ขั้นตอนการดำเนินงานของโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์สามารถอธิบายโดยการเขียนให้อยู่ในรูปแผนภาพกระแสข้อมูล (Data Flow Diagram) ซึ่งช่วยให้เข้าใจกระบวนการดำเนินงานต่าง ๆ และการไหลเวียนของข้อมูลที่เกิดขึ้นในโปรแกรม สำหรับแผนภาพกระแสข้อมูลของกระบวนการที่สำคัญของโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์สามารถแสดงได้ดังภาพประกอบ 4.2 ถึงภาพประกอบ 4.9 ส่วนคำอธิบายรายละเอียดการทำงานของกระบวนการต่าง ๆ จากแผนภาพกระแสข้อมูลแสดงได้ดังตาราง 4.1 ถึงตาราง 4.8 ตามลำดับ



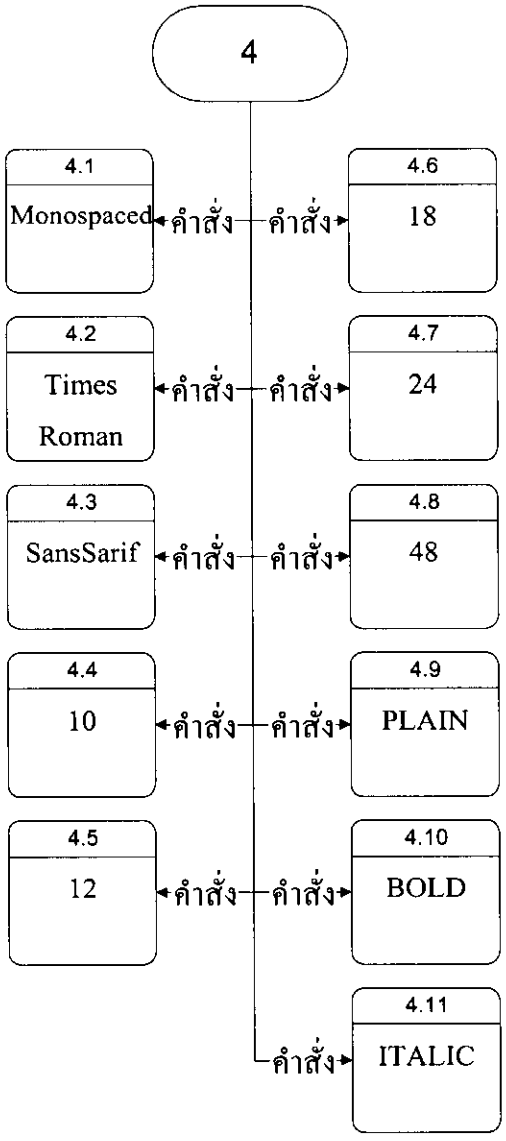
ภาพประกอบ 4.2 แผนภาพกระแสข้อมูลของโปรแกรมจำลองฯ



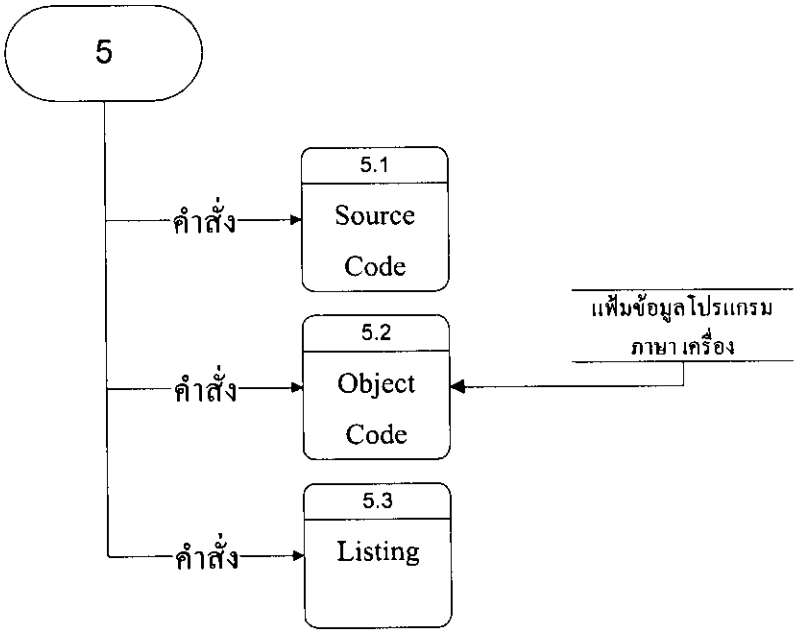
ภาพประกอบ 4.3 แผนภาพกระแสข้อมูลกระบวนการ เมนู File



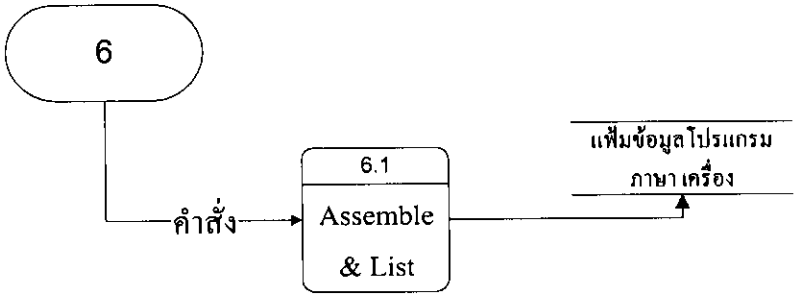
ภาพประกอบ 4.4 แผนภาพกระแสข้อมูลกระบวนการ เมนู Edit



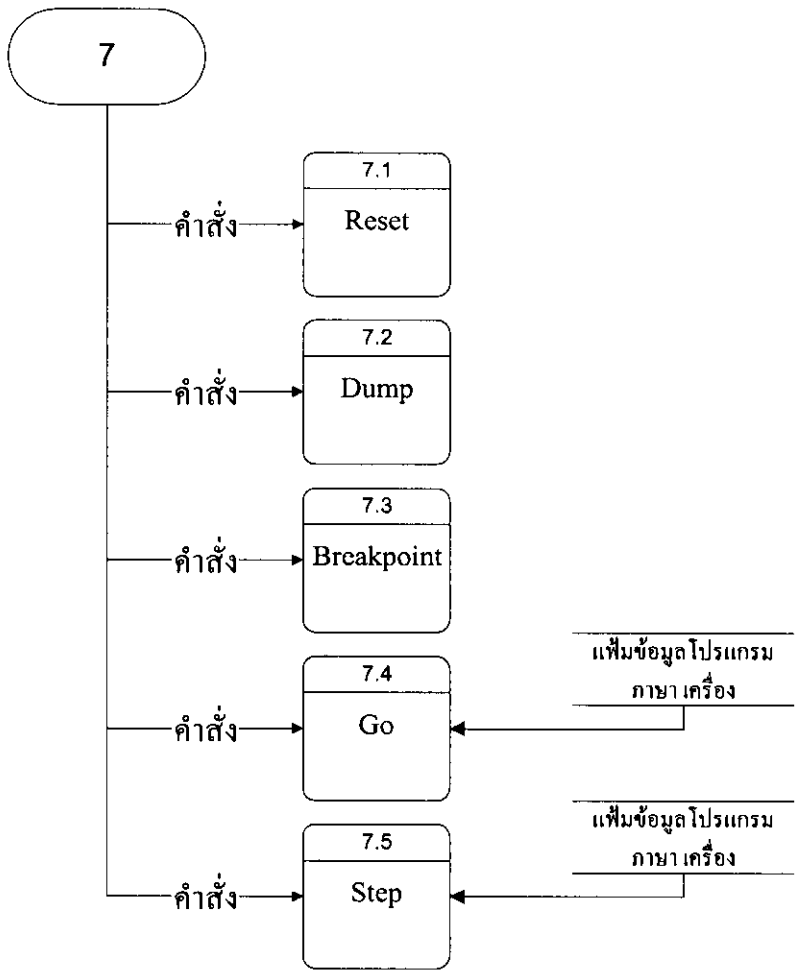
ภาพประกอบ 4.5 แผนภาพกระแสข้อมูลกระบวนการ เมนู Font



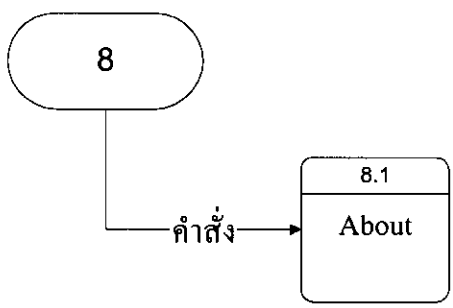
ภาพประกอบ 4.6 แผนภาพกระแสข้อมูลกระบวนการ เมนู View



ภาพประกอบ 4.7 แผนภาพกระแสข้อมูลกระบวนการ เมนู Assemble



ภาพประกอบ 4.8 แผนภาพกระแสข้อมูลกระบวนการ เมนู Run



ภาพประกอบ 4.9 แผนภาพกระแสข้อมูลกระบวนการ เมนู Help

ตาราง 4.1 รายละเอียดการดำเนินงานของแต่ละกระบวนการในภาพประกอบ 4.2

กระบวนการที่	การทำงาน
1	ควบคุมการเลือกรายการหลักของโปรแกรม เป็นการแสดงผลแบบ Pull Down Menu
2	ควบคุมการเลือกรายการย่อยของกระบวนการ File
3	ควบคุมการเลือกรายการย่อยของกระบวนการ Edit
4	ควบคุมการเลือกรายการย่อยของกระบวนการ Font
5	ควบคุมการเลือกรายการย่อยของกระบวนการ View
6	ควบคุมการเลือกรายการย่อยของกระบวนการ Assemble
7	ควบคุมการเลือกรายการย่อยของกระบวนการ Run
8	ควบคุมการเลือกรายการย่อยของกระบวนการ Help

ตาราง 4.2 รายละเอียดการดำเนินงานของแต่ละกระบวนการในภาพประกอบ 4.3

กระบวนการที่	การทำงาน
2.1	เขียนโปรแกรมภาษาแอสเซมบลีใหม่
2.2	เปิดโปรแกรมภาษาแอสเซมบลีจากแฟ้มข้อมูล
2.3	ปิดแฟ้มข้อมูลโปรแกรมภาษาแอสเซมบลี
2.4	บันทึกโปรแกรมภาษาแอสเซมบลีเก็บไว้ในแฟ้มข้อมูล
2.5	บันทึกโปรแกรมภาษาแอสเซมบลีเก็บไว้ในแฟ้มข้อมูลโดยใช้ชื่อใหม่
2.6	กำหนดรายละเอียดต่าง ๆ ของกระดาษ
2.7	พิมพ์ลงบนเครื่องพิมพ์
2.8	ออกจากโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์

ตาราง 4.3 รายละเอียดการดำเนินงานของแต่ละกระบวนการในภาพประกอบ 4.4

กระบวนการที่	การทำงาน
3.1	ลบข้อความที่ถูกเลือกและคัดลอกข้อความเหล่านั้นไปเก็บไว้ในหน่วยความจำหลัก
3.2	คัดลอกข้อความที่ถูกเลือกไปเก็บไว้ในหน่วยความจำหลัก
3.3	กำหนดข้อความ โดยทำสำเนาจากข้อความที่ถูกคัดลอกไว้ในหน่วยความจำหลัก
3.4	ลบข้อความทั้งหมดออกจากพื้นที่ที่ใช้เขียนโปรแกรมภาษาแอสเซมบลี
3.5	เลือกข้อความทั้งหมด

ตาราง 4.4 รายละเอียดการดำเนินงานของแต่ละกระบวนการในภาพประกอบ 4.5

กระบวนการที่	การทำงาน
4.1	กำหนดรูปแบบตัวอักขระเป็นแบบ Monospaced
4.2	กำหนดรูปแบบตัวอักขระเป็นแบบ Time Roman
4.3	กำหนดรูปแบบตัวอักขระเป็นแบบ San Sarif
4.4	กำหนดขนาดอักขระให้มีขนาด 10
4.5	กำหนดขนาดอักขระให้มีขนาด 12
4.6	กำหนดขนาดอักขระให้มีขนาด 18
4.7	กำหนดขนาดอักขระให้มีขนาด 24
4.8	กำหนดขนาดอักขระให้มีขนาด 48
4.9	กำหนดรูปแบบตัวอักขระเป็นแบบตัวปกติ
4.10	กำหนดรูปแบบตัวอักขระเป็นแบบตัวหนา
4.11	กำหนดรูปแบบตัวอักขระเป็นแบบตัวเอียง

ตาราง 4.5 รายละเอียดการดำเนินงานของแต่ละกระบวนการในภาพประกอบ 4.6

กระบวนการที่	การทำงาน
5.1	แสดงโปรแกรมภาษาแอสเซมบลี
5.2	แสดงโปรแกรมภาษาเครื่องของเครื่อง PDP-8 ที่จำลอง
5.3	แสดงรายละเอียดต่าง ๆ จากการแปลโปรแกรมภาษาแอสเซมบลี

ตาราง 4.6 รายละเอียดการดำเนินงานของแต่ละกระบวนการในภาพประกอบ 4.7

กระบวนการที่	การทำงาน
6.1	แปลโปรแกรมภาษาแอสเซมบลีให้เป็นภาษาเครื่องของเครื่อง PDP-8 ที่จำลอง และ แสดงรายละเอียดต่าง ๆ จากการแปลโปรแกรมภาษาแอสเซมบลี โดยแสดงผลเช่นเดียวกับการดำเนินงานของกระบวนการที่ 5.3

ตาราง 4.7 รายละเอียดการดำเนินงานของแต่ละกระบวนการในภาพประกอบ 4.8

กระบวนการที่	การทำงาน
7.1	ปรับสภาพแวดล้อมเพื่อเตรียมความพร้อมในการดำเนินงานใหม่กับโปรแกรม
7.2	แสดงรายละเอียดตำแหน่งความจำหลักที่เก็บค่าต่าง ๆ จากการดำเนินงานของโปรแกรม
7.3	กำหนดจุดหยุดการทำงานของคำสั่งภาษาเครื่องที่ต้องการพิจารณาผลการดำเนินงานหลังเสร็จสิ้นการดำเนินงานของคำสั่งนั้น
7.4	ดำเนินงานตามโปรแกรมภาษาเครื่อง PDP-8
7.5	ดำเนินงานตามคำสั่งภาษาเครื่อง PDP-8 ทีละคำสั่งโดยผู้ใช้กำหนด

ตาราง 4.8 รายละเอียดการดำเนินงานของแต่ละกระบวนการในภาพประกอบ 4.9

กระบวนการที่	การทำงาน
8.1	แสดงข้อมูลทั่วไปเกี่ยวกับโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์

4.3 ขั้นตอนวิธี

โปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์มีขั้นตอนวิธีการดำเนินงานของกระบวนการต่าง ๆ ซึ่งสามารถอธิบายโดยเขียนให้อยู่ในลักษณะของ Pseudo-code สำหรับขั้นตอนวิธีที่สำคัญมีดังนี้

การทำงานหลักของโปรแกรม

โปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์ถูกพัฒนาให้ทำงานในลักษณะให้ทำงานตามเหตุการณ์ที่เกิดขึ้นจากการเลือกเมนูการทำงาน โดยขั้นตอนวิธีสามารถแสดงได้ดังภาพประกอบ 4.10

การทำงานในส่วนของการแปลโปรแกรมภาษาแอสเซมบลี

ในการโปรแกรมภาษาแอสเซมบลีจะใช้การอ่านโปรแกรมภาษาแอสเซมบลีที่ต้องการแปลจำนวนสองรอบจึงสามารถทำการแปลเป็นภาษาเครื่องได้ โดยส่วนการทำงานหลักในการแปลโปรแกรมภาษาแอสเซมบลีมีขั้นตอนวิธีที่แสดงได้ดังภาพประกอบ 4.11 ส่วนรายละเอียดในการแปลภาษาแอสเซมบลีในรอบที่ 1 และรอบที่ 2 สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.12 และ 4.13 ตามลำดับ

การทำงานในส่วนของการจำลองการทำงานตามภาษาเครื่อง

ในส่วนของการนำภาษาเครื่องมาจำลองการทำงานนั้นมีขั้นตอนวิธีหลักที่เป็นส่วนสำคัญในการจำลองการทำงานตามภาษาเครื่องที่แปลจากส่วนแปลภาษาแอสเซมบลีดังนี้

- ส่วนที่ทำหน้าที่ตรวจสอบตำแหน่งของหน่วยความจำหลักที่ถูกอ้างในคำสั่ง สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.14
- ส่วนที่ทำหน้าที่แยกคำสั่งภาษาเครื่องจำนวน 8 คำสั่งเพื่อส่งการทำงานให้กับฟังก์ชันของคำสั่งต่าง ๆ สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.15
- ส่วนที่จำลองการทำงานตามภาษาเครื่องของคำสั่ง and สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.16
- ส่วนที่จำลองการทำงานตามภาษาเครื่องของคำสั่ง tad สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.17
- ส่วนที่จำลองการทำงานตามภาษาเครื่องของคำสั่ง isz สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.18

- ส่วนที่จำลองการทำงานตามภาษาเครื่องของคำสั่ง dca สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.19
- ส่วนที่จำลองการทำงานตามภาษาเครื่องของคำสั่ง jms สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.20
- ส่วนที่จำลองการทำงานตามภาษาเครื่องของคำสั่ง jmp สามารถแสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.21
- ส่วนจัดการคำสั่งในกลุ่ม opreate ซึ่งประกอบด้วยคำสั่งสองกลุ่ม เมื่อตรวจสอบว่าเป็นคำสั่งภาษาเครื่องในกลุ่ม operate แล้ว จะทำการแยกว่าเป็นคำสั่งในกลุ่มใดก่อนจึงนำไปแยกเป็นคำสั่งภาษาเครื่องและดำเนินการทำงานตามคำสั่งต่าง ๆ ต่อไป สามารถแสดงขั้นตอนวิธีในการแยกคำสั่งในกลุ่ม operate ได้ดังภาพประกอบ 4.22 ส่วนคำสั่ง ratete ได้แสดงขั้นตอนวิธีได้ดังภาพประกอบ 4.23 และ 4.24 ตามลำดับ

START *"การทำงานหลักของโปรแกรม"*
CREATE สร้างวินโดว์หลักของโปรแกรม
DO
    WAIT รอจนกว่าจะมีการเรียกเมนูคำสั่ง
    CALL เรียกฟังก์ชันการทำงานที่เกี่ยวข้องกันเมนูที่เลือก
WHILE เมนูที่เลือกไม่ใช่เมนูจบการทำงานของโปรแกรม
ERASE ทำลายวินโดว์ของโปรแกรม
STOP

ภาพประกอบ 4.10 ขั้นตอนวิธี การทำงานหลักของโปรแกรม

START *"การทำงานหลักในการแปลโปรแกรมภาษาแอสเซมบลี"*
SET กำหนดค่าเริ่มต้น
CALL *"การแปลภาษาแอสเซมบลีรอบที่ 1"*
IF มีความผิดพลาดจากการแปลภาษาแอสเซมบลี
    PRINT รายงานความผิดพลาดจากการแปลในรอบที่ 1
END IF
CALL *"การแปลภาษาแอสเซมบลีรอบที่2"*

ภาพประกอบ 4.11 ขั้นตอนวิธี การทำงานหลักในการแปลโปรแกรมภาษาแอสเซมบลี

IF ไม่มีความผิดพลาดจากการแปลภาษาแอสเซมบลี

PRINT รายงานความสำเร็จจากการแปลภาษาแอสเซมบลี

END IF

SET เก็บผลจากการแปลภาษาแอสเซมบลี

STOP

ภาพประกอบ 4.11 ขั้นตอนวิธี การทำงานหลักในการแปลโปรแกรมภาษาแอสเซมบลี (ต่อ)

START *"การแปลภาษาแอสเซมบลีรอบที่ 1"*

SET กำหนดค่าเริ่มต้น

WHILE ยังไม่สิ้นสุดโปรแกรมภาษาแอสเซมบลี DO

SWITCH CALL *"อ่านอักขระชุดถัดไป"*

CASE เป็น COMMENT

CALL *"ให้ข้ามไปบรรทัดถัดไป"*

CASE เป็น ORIGIN

SET กำหนดค่าตำแหน่งหน่วยความจำเริ่มต้น

CASE เป็น LABELS

CALL *"เพิ่ม SYMBOL ลงในตาราง"*

CASE เป็น ENDING

SET สิ้นสุดโปรแกรม = TRUE

CASE เป็น OCTAL หรือ IDENTIFIER

SET เพิ่มค่าตำแหน่งหน่วยความจำ

CALL *"ให้ข้ามไปบรรทัดถัดไป"*

END WHILE

STOP

ภาพประกอบ 4.12 ขั้นตอนวิธี การแปลภาษาแอสเซมบลีรอบที่ 1

START *"การแปลภาษาแอสเซมบลีรอบที่ 2"*

SET กำหนดค่าเริ่มต้น

WHILE ยังไม่สิ้นสุดโปรแกรมภาษาแอสเซมบลีหรือไม่พบข้อผิดพลาด DO

SWITCH CALL *"อ่านอักขระชุดถัดไป"*

CASE เป็น COMMENT

CALL *"ให้ข้ามไปบรรทัดถัดไป"*

CASE เป็น ORIGIN

SET กำหนดค่าตำแหน่งหน่วยความจำเริ่มต้น

IF CALL *"อ่านอักขระชุดถัดไป"* ไม่เท่ากับ NULL

CALL *"ตรวจสอบความผิดพลาด"*

END IF

CASE เป็น OCTAL หรือ VARIABLE

CALL *"หาภาษาเครื่อง"*

CASE เป็น MRIOPCODE

CALL *"MEMREFS"*

CASE เป็น IOTOPCODE

CALL *"IOTS"*

CASE เป็น OPROPCODE

CALL *"OPERATE"*

CASE เป็น ENDING

SET สิ้นสุดโปรแกรม = TRUE

CASE เป็น NULL หรือ ERRORS หรือ LAB8

END WHILE

STOP

ภาพประกอบ 4.13 ขั้นตอนวิธี การแปลภาษาแอสเซมบลีรอบที่ 2

```
START "การตรวจสอบตำแหน่งของความจำหลักที่ถูกอ้างในคำสั่ง"
IF opcode บิต 4 == 1
        SET cpage = TRUE
END IF
IF opcode บิต 3 == 1
        SET indirect = TRUE
END IF
SET opcode = opcode AND 127
IF indirect == FALSE AND cpage == FALSE
        SET addr = opcode
ELSE    IF indirect == FALSE AND cpage == TRUE
                SET addr = PC AND 3968 + opcode
        ELSE    IF indirect == TRUE AND cpage == FALSE
                        SET addr = memory[opcode]
                ELSE
                        temp = PC AND 3968 + opcode
                        addr = memory[remp]
                END IF
        END IF
END IF
addr = addr AND 4095
STOP
```

ภาพประกอบ 4.14 ขั้นตอนวิธี การตรวจสอบตำแหน่งของความจำหลักที่ถูกอ้างในคำสั่ง

```
START "การแยกคำสั่งภาษาเครื่อง"
IF ตรวจสอบบิต 0, 1 และ 2 == 000
        CALL "การทำงานคำสั่ง and"
ELSE    IF ตรวจสอบบิต 0, 1 และ 2 == 001
                CALL "การทำงานคำสั่ง tad"
```

ภาพประกอบ 4.15 ขั้นตอนวิธี การแยกคำสั่งภาษาเครื่อง

```
ELSE  IF ตรวจสอบบิต 0, 1 และ 2 == 010
          CALL "การทำงานคำสั่ง isz"
      ELSE  IF ตรวจสอบบิต 0, 1 และ 2 == 011
                CALL "การทำงานคำสั่ง dca"
            ELSE  IF ตรวจสอบบิต 0, 1 และ 2 == 100
                      CALL "การทำงานคำสั่ง jms"
                  ELSE  IF ตรวจสอบบิต 0, 1 และ 2 == 101
                            CALL "การทำงานคำสั่ง jmp"
                        ELSE  IF ตรวจสอบบิต 0, 1 และ 2 == 110
                                  CALL "การทำงานคำสั่ง iot"
                              ELSE  IF ตรวจสอบบิต 0, 1 และ 2 == 111
                                        CALL "การทำงานคำสั่ง opr"
                                    END IF
                              END IF
                        END IF
                  END IF
            END IF
      END IF
END IF
STOP
```

ภาพประกอบ 4.15 ขั้นตอนวิธี การแยกคำสั่งภาษาเครื่อง (ต่อ)

```
START "การทำงานตามภาษาเครื่องของคำสั่ง and"
SET address = CALL "การตรวจสอบตำแหน่งของหน่วยความจำหลักที่ถูกอ้างในคำสั่ง"
SET ACC = ACC AND memory[address]
STOP
```

ภาพประกอบ 4.16 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่ง and

START *"การทำงานตามภาษาเครื่องของคำสั่ง tad"*

SET address = CALL *"การตรวจสอบตำแหน่งของหน่วยความจำหลักที่ถูกอ้างในคำสั่ง"*

SET ACC = ACC + memory[address]

IF ตรวจสอบ ACC ว่ามีจำนวน 13 บิต

    LINK = NOT LINK

END IF

SET ACC = ACC AND 4095

STOP

ภาพประกอบ 4.17 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่ง tad

START *"การทำงานตามภาษาเครื่องของคำสั่ง isz"*

SET address = CALL *"การตรวจสอบตำแหน่งของหน่วยความจำหลักที่ถูกอ้างในคำสั่ง"*

SET memory[address] = memory[address] + 1

IF ตรวจสอบ memory[address] == 0

    SET PC = PC + 1

END IF

SET PC = PC AND 4095

STOP

ภาพประกอบ 4.18 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่ง isz

START *"การทำงานตามภาษาเครื่องของคำสั่ง dca"*

SET address = CALL *"การตรวจสอบตำแหน่งของหน่วยความจำหลักที่ถูกอ้างในคำสั่ง"*

SET memory[address] = ACC

SET ACC = 0

STOP

ภาพประกอบ 4.19 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่ง dca

START *"การทำงานตามภาษาเครื่องของคำสั่ง jms"*

SET address = CALL *"การตรวจสอบตำแหน่งของหน่วยความจำหลักที่ถูกอ้างในคำสั่ง"*

SET memory[address] = PC

SET PC = address + 1

STOP

ภาพประกอบ 4.20 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่ง jms

START *"การทำงานตามภาษาเครื่องของคำสั่ง jmp"*

SET address = CALL *"การตรวจสอบตำแหน่งของหน่วยความจำหลักที่ถูกอ้างในคำสั่ง"*

SET PC = address

STOP

ภาพประกอบ 4.21 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่ง jmp

START *"การจัดการคำสั่งในกลุ่ม operate"*

IF คำสั่งภาษาเครื่อง บิต 3 == 0

IF คำสั่งภาษาเครื่อง บิต 4 ==1

SET ACC = 0

END IF

IF คำสั่งภาษาเครื่อง บิต 5 == 1

SET LINK = FALSE

END IF

IF คำสั่งภาษาเครื่อง บิต 6 == 1

SET ACC = NOT ACC

END IF

IF คำสั่งภาษาเครื่อง บิต 7 == 1

SET LINK = NOT LINK

END IF

ภาพประกอบ 4.22 ขั้นตอนวิธี การจัดการคำสั่งในกลุ่ม operate

```
IF คำสั่งภาษาเครื่อง บิต 11 == 1
        SET ACC = ACC + 1
END IF
IF คำสั่งภาษาเครื่อง บิต 8 == 1
        CALL "เลื่อนข้อมูลทางขวา"
        IF คำสั่งภาษาเครื่อง บิต 10 == 1
                CALL "เลื่อนข้อมูลทางขวา"
        END IF
END IF
IF คำสั่งภาษาเครื่อง บิต 9 == 1
        CALL "เลื่อนข้อมูลทางซ้าย"
        IF คำสั่งภาษาเครื่อง บิต 10 == 1
                CALL "เลื่อนข้อมูลทางซ้าย"
        END IF
END IF
ELSE
        IF คำสั่งภาษาเครื่อง บิต 8 == 0
                IF คำสั่งภาษาเครื่อง บิต 5 == 1
                        IF ตรวจสอบ ACC บิต 0 == 1
                                SET PC = PC +1
                        END IF
                END IF
                IF คำสั่งภาษาเครื่อง บิต 6 == 1 AND คำสั่งภาษาเครื่อง บิต 5 == 0
                        IF ACC == 0
                                SET PC = PC +1
                        END IF
                END IF
                IF คำสั่งภาษาเครื่อง บิต 7 == 1
```

ภาพประกอบ 4.22 ขั้นตอนวิธี การจัดการคำสั่งในกลุ่ม operate (ต่อ)

```
            IF LINK == TRUE
                SET PC = PC +1
            END IF
        END IF
ELSE
        IF คำสั่งภาษาเครื่อง บิต 5, 6, 7 == 1
            SET PC = PC +1
        END IF
        IF คำสั่งภาษาเครื่อง บิต 7 = 1
            IF LINK == FALSE
                SET PC = PC +1
            END IF
        END IF
        IF คำสั่งภาษาเครื่อง บิต 5, 6 == 1
            IF ACC บิต 0 == 0 AND ACC > 0
                SET PC = PC +1
            END IF
        END IF
        IF คำสั่งภาษาเครื่อง บิต 7 == 1
            IF ACC <> 0
                SET PC = PC +1
            END IF
        END IF
        IF คำสั่งภาษาเครื่อง บิต 5 == 1
            IF ACC บิต 0 == 0 AND ACC >= 0
                SET PC = PC +1
            END IF
        END IF
END IF
```

ภาพประกอบ 4.22 ขั้นตอนวิธี การจัดการคำสั่งในกลุ่ม operate (ต่อ)

```
        IF คำสั่งภาษาเครื่อง บิต 4  == 1
                SET ACC = 0
        END IF
        IF คำสั่งภาษาเครื่อง บิต 10  == 1
                SET running = FALSE
        END IF
END IF
SET PC = address
STOP
```

ภาพประกอบ 4.22 ขั้นตอนวิธี การจัดการคำสั่งในกลุ่ม operate (ต่อ)

```
START  "การทำงานตามภาษาเครื่องของคำสั่งเลื่อนข้อมูลทางซ้าย"
SET  ACC = เลื่อนข้อมูลใน ACC ไปทางซ้าย จำนวน 1 บิต
IF LINK == TRUE
        SET ACC = ACC OR 1
END IF
IF ACC AND 4096 == 4096
        SET LINK = TRUE
ELSE    LINK = FALSE
END IF
SET ACC = ACC AND 4095
STOP
```

ภาพประกอบ 4.23 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่งเลื่อนข้อมูลทางซ้าย

```
START  "การทำงานตามภาษาเครื่องของคำสั่งเลื่อนข้อมูลทางขวา"
IF ACC AND 1 == 1
        SET link_tmp = TRUE
END IF
```

ภาพประกอบ 4.24 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่งเลื่อนข้อมูลทางขวา

SET ACC = เลื่อนข้อมูลใน ACC ไปทางขวา จำนวน 1 บิต

IF LINK == TRUE

    SET ACC = ACC OR 2048

END IF

SET LINK = link_tmp

SET ACC = ACC AND 4095

STOP

ภาพประกอบ 4.24 ขั้นตอนวิธี การทำงานตามภาษาเครื่องของคำสั่งเลื่อนข้อมูลทางขวา (ต่อ)

4.4 การดำเนินงาน

จากโครงสร้างข้อมูลในบทที่ 3 สามารถนำมาพัฒนาโปรแกรมตามรูปแบบของภาษาจาวา โดยมีโครงสร้างข้อมูลในลักษณะของคลาสตามรายละเอียดดังนี้

คลาส AsmTextInfo

คลาส AsmTextInfo ใช้ในการเก็บผลจากการแปลภาษาแอสเซมบลี มีรายละเอียดของตัวแปรสมาชิกดังแสดงในตาราง 4.9

คลาส Assemble8

คลาส Assemble8 สืบทอดมาจากคลาส Jframe เป็นคลาสที่ใช้ในการจัดการแปลภาษาแอสเซมบลี มีรายละเอียดของตัวแปรสมาชิกและฟังก์ชันสมาชิกซึ่งแก้ไขเพิ่มเติมจากคลาส Jframe ดังแสดงในตาราง 4.10 และ ตาราง 4.11 ตามลำดับ

คลาส Exec8

คลาส Exec8 สืบทอดมาจากคลาส Jframe เป็นคลาสที่ใช้ในการดำเนินการตามคำสั่งจากภาษาเครื่องที่แปลได้ภาษาแอสเซมบลีโดยมีการจำลองการทำงานจากความจำหลักจำลอง มีรายละเอียดของตัวแปรสมาชิกและฟังก์ชันสมาชิกซึ่งแก้ไขเพิ่มเติมจากคลาส Jframe ดังแสดงในตาราง 4.12 และ ตาราง 4.13 ตามลำดับ

คลาส simulateCPUDialog

คลาส simulateCPUDialog สืบทอดมาจากคลาส Jframe เป็นคลาสที่เป็นวินโดว์ที่ใช้ในการแสดงค่าต่าง ๆ ภายในหน่วยประมวลผลกลางจำลอง

คลาส teletypeDialog

คลาส teletypeDialog สืบทอดมาจากคลาส Jframe เป็นคลาสที่เป็นวินโดว์ที่ใช้ในการแสดงค่าต่าง ๆ ภายในอุปกรณ์เอาต์พุตจำลองที่เป็น teletype

คลาส MyFilter

คลาส MyFilter สืบทอดมาจากคลาส FileFilter เป็นคลาสที่ทำหน้าที่จัดการกับแฟ้มข้อมูลที่เก็บโปรแกรมภาษาแอสเซมบลี มีรายละเอียดของตัวแปรสมาชิกและฟังก์ชันสมาชิกซึ่งแก้ไขเพิ่มเติมจากคลาส FileFilter ดังแสดงในตาราง 4.14 และ ตาราง 4.15 ตามลำดับ

คลาส PSUSim8

คลาส PSUSim8 สืบทอดมาจากคลาส JFrame เป็นคลาสหลักของโปรแกรมที่ทำหน้าที่ควบคุมการทำงานของวินโดว์ของโปรแกรมจำลองฯ มีรายละเอียดของตัวแปรสมาชิกและฟังก์ชันสมาชิกซึ่งแก้ไขเพิ่มเติมจากคลาส Jframe ดังแสดงในตาราง 4.16 และ ตาราง 4.17 ตามลำดับ

ตาราง 4.9 รายละเอียดของตัวแปรสมาชิกของคลาส AsmTextInfo

ชื่อตัวแปรสมาชิก	**หน้าที่**
AsmObject	เก็บอักขระที่เป็นภาษาเครื่องที่ได้จากการแปลภาษาแอสเซมบลี
AsmListing	เก็บอักขระที่แสดงรายละเอียดต่าง ๆ ซึ่งเป็นผลลัพธ์ทั้งหมดจากการแปลภาษาแอสเซมบลี

ตาราง 4.10 รายละเอียดของตัวแปรสมาชิกของคลาส Assemble8

ชื่อตัวแปรสมาชิก	หน้าที่
DD, LN, LO, XP, SX, UNDEF	เป็นตัวแปรค่าคงที่ใช้ในการแสดงความหมายความผิดพลาดจากการแปลภาษาแอสเซมบลี
LABEL8, MRI, OPR, IOT, INDIR	เป็นตัวแปรค่าคงที่ใช้แสดงความหมายของกลุ่มอักขระว่าเป็นกลุ่มอักขระแทนคำสั่งชนิดใด
LABELS, COMMENT, OCTAL, ORIGIN, ENDING, IDENTIFIER, NIL, ERRORS	เป็นตัวแปรค่าคงที่ใช้ในการแสดงความหมายของกลุ่มอักขระที่เป็นโทเคนว่าเป็นโทเคนชนิดใด
XLAB8, XCOMMENT, XOCTAL, XORIGIN, XENDING, XNIL, XERRORS, XVARIABLE, XMRIOPCODE, XIOTOPCODE, XOPROPCODE, XINDIRECTBIT	เป็นตัวแปรค่าคงที่ใช้ในการแสดงความหมายของกลุ่มอักขระที่เป็นโทเคนว่าเป็นโทเคนชนิดใด ซึ่งจะใช้ค่าคงที่เหล่านี้ในการแปลภาษาแอสเซมบลีรอบที่ 2
START, IDENT, INOCT, DONE	เป็นตัวแปรค่าคงที่ใช้ในการแสดงสถานะในการอ่านชุดอักขระจากโปรแกรมภาษาแอสเซมบลี
NAME	เป็นตัวแปรค่าคงที่ที่มีโครงสร้างข้อมูลแบบแถวลำดับใช้ในการเก็บชุดอักขระที่เป็นคำสั่งภาษาแอสเซมบลีแต่ละคำสั่ง
STYPE	เป็นตัวแปรค่าคงที่ที่มีโครงสร้างข้อมูลแบบแถวลำดับใช้ในการเก็บค่าที่ใช้แทนชนิดของคำสั่งภาษาแอสเซมบลีในแต่ละคำสั่ง
VALUE	เป็นตัวแปรค่าคงที่ที่มีโครงสร้างข้อมูลแบบแถวลำดับใช้ในการเก็บค่าที่ใช้เป็นภาษาเครื่องซึ่งเป็นค่าเริ่มต้นของแต่ละคำสั่ง
error	แสดงว่ามีความผิดพลาดจากการแปลภาษาแอสเซมบลีหรือไม่
ended	แสดงว่าชุดอักขระภาษาแอสเซมบลีที่จะทำการแปลสิ้นสุดหรือไม่

ตาราง 4.10 รายละเอียดของตัวแปรสมาชิกของคลาส Assemble8 (ต่อ)

ชื่อตัวแปรสมาชิก	หน้าที่
line	เก็บชุดอักขระในบรรทัดที่ต้องการแปล
numchars	จำนวนอักขระในบรรทัดที่กำลังแปล
pos_source	ตำแหน่งบรรทัดของชุดอักขระ
posn	ตำแหน่งของอักขระในบรรทัด
symtab	เป็นโครงสร้างข้อมูลที่เป็นแถวลำดับจำนวน 512 แถว โดยในแต่ละแถวลำดับ เก็บค่าของคำสั่งภาษาแอสเซมบลี ชนิดคำสั่งและภาษาเครื่อง
numsym	จำนวนสัญลักษณ์ที่กำหนดขึ้น
where	ตำแหน่งของหน่วยความจำ
TextSource	ชุดอักขระที่เป็นโปรแกรมภาษาแอสเซมบลี
Object	ชุดอักขระที่เป็นโปรแกรมภาษาเครื่อง
ListingText	ชุดอักขระที่เก็บค่ารายละเอียดต่าง ๆ จากการแปลภาษาแอสเซมบลี
passno	รอบในการแปล
printval	ค่าที่เป็นภาษาเครื่อง
listing	แสดงว่าต้องการเก็บรายละเอียดจากการแปลหรือไม่
errpass	แสดงรอบที่เกิดความผิดพลาดจากการแปล

ตาราง 4.11 รายละเอียดของฟังก์ชันสมาชิกของคลาส Assemble8

ชื่อฟังก์ชันสมาชิก	หน้าที่
errs	จัดการและแสดงผลความผิดพลาดจากการแปลภาษาแอสเซมบลี
andsymbol	นำค่าจากตารางที่เก็บคำสั่งที่จะแสดงร่วมกับภาษาเครื่องมาเก็บในตัวแปลชุดอักขระเพื่อเตรียมนำไปแสดงผล
printsymtab	นำค่าจากตารางที่เก็บคำสั่งที่จะแสดงร่วมกับภาษาเครื่องและรายละเอียดต่าง ๆ จากการแปลภาษาแอสเซมบลีมาเก็บในตัวแปลชุดอักขระเพื่อเตรียมนำไปแสดงผล
tab	พิมพ์ช่องว่างตามระยะที่กำหนด

ตาราง 4.11 รายละเอียดของฟังก์ชันสมาชิกของคลาส Assemble8 (ต่อ)

ชื่อฟังก์ชันสมาชิก	หน้าที่
dumpline	ขึ้นบรรทัดใหม่
skipline	ข้ามบรรทัด
getline	นำโปรแกรมภาษาแอสเซมบลี 1 บรรทัดมาเพื่อเตรียมทำการแปล
nextsym	ค้นหาชุดอักขระชุดต่อไปจากโปรแกรมภาษาแอสเซมบลีและทำการกำหนดว่าชุดอักขระที่ได้เป็นสถานะใด
findposn	ค้นหาตำแหน่งของชุดอักขระในตารางเก็บคำสั่ง
xnextsym	ค้นหาชุดอักขระชุดต่อไปจากโปรแกรมภาษาแอสเซมบลีและทำการกำหนดว่าชุดอักขระที่ได้เป็นสถานะใดโดยในในการแปลของรอบที่สอง
insertsymbol	เพิ่มชุดอักขระที่ได้ลงในตาราง
pass1	แปลภาษาแอสเซมบลีในรอบที่หนึ่ง
writecode	เก็บค่าของภาษาเครื่องที่ได้ไว้ในตัวแปลเพื่อเตรียมนำไปใช้งาน
iots	หาค่าของภาษาเครื่องของคำสั่งในกลุ่ม IOT
operate	หาค่าของภาษาเครื่องของคำสั่งในกลุ่ม OPERAATE
memrefs	หาค่าของภาษาเครื่องของคำสั่งในกลุ่ม Memory Reference
pass2	แปลภาษาแอสเซมบลีในรอบที่สอง
initialise	กำหนดค่าเริ่มต้นก่อนเริ่มทำการแปลภาษาแอสเซมบลี
zeroline	กำหนดบรรทัดว่างหนึ่งบรรทัด
smap	ดำเนินการแปลภาษาแอสเซมบลี

ตาราง 4.12 รายละเอียดของตัวแปรสมาชิกของคลาส Exec8

ชื่อตัวแปรสมาชิก	หน้าที่
memory	เป็นตัวแปรค่าคงที่ที่มีโครงสร้างข้อมูลแบบแถวลำดับใช้ในการจำลองหน่วยความจำหลัก
PC	ใช้ในการจำลองรีจิสเตอร์ program counter
ACC	ใช้ในการจำลองรีจิสเตอร์ accumulator
LINK	ใช้ในการจำลองรีจิสเตอร์ link

ตาราง 4.12 รายละเอียดของตัวแปรสมาชิกของคลาส Exec8 (ต่อ)

ชื่อตัวแปรสมาชิก	หน้าที่
INT_EN	ใช้ในการจำลองรีจิสเตอร์ interrupt-enabled
TTYPE_FLAG	ใช้ในการจำลองรีจิสเตอร์ teletype flag
TTYPE_REG	ใช้ในการจำลองรีจิสเตอร์ teletype
ttypeText	อักขระที่จะแสดงที่ teletype
KRB_FLAG	ใช้ในการจำลองรีจิสเตอร์ keyboard flag
KRB_REG	ใช้ในการจำลองรีจิสเตอร์ keyboard
CLK_FLAG	ใช้ในการจำลองรีจิสเตอร์ clock flag
Location	ตำแหน่งของหน่วยความจำหลัก
pos_object	ตำแหน่งของภาษาเครื่อง
line	บรรทัดอักขระ
temp	ตัวแปรเก็บอักขระชั่วคราว
Object	เก็บอักขระที่แสดงภาษาเครื่อง
Running	แสดงสถานะว่ากำลังอยู่ในระหว่างการทำงานตามคำสั่งภาษาเครื่อง
Address	ตำแหน่งของหน่วยความจำที่ถูกอ้างจากคำสั่งภาษาเครื่อง
DumpText	เก็บชุดอักขระที่แทนค่าในหน่วยความจำที่จะนำไปแสดง
RunStep	กำหนดให้จำลองการทำงานแบบครั้งละคำสั่ง
keyboardInput	เก็บชุดอักขระที่ได้รับจากการป้อนเข้าทางแป้นพิมพ์
DISK_TRANS_FLAG	ใช้ในการจำลองรีจิสเตอร์ disk transmit flag
disk_block_register	ใช้ในการจำลองรีจิสเตอร์ disk controller
diskBuffer	เก็บชุดอักขระที่จะนำไปติดต่อกับจานบันทึก
seekIndex	ตำแหน่งในการหาข้อมูลในจานบันทึก
DISK_SEEK_FLAG	ใช้ในการจำลองรีจิสเตอร์ disk seek flag
DiskMemoryReg	ใช้ในการจำลองรีจิสเตอร์ disk memory
seekTemp	เก็บค่าที่จะติดต่อกับจานบันทึกชั่วคราว
SimClock	จำลองวงจรนาฬิกาของระบบ
ClockTick	จังหวะของสัญญาณนาฬิกา
ClockUse	แสดงการใช้วงจรนาฬิกาหรือไม่

ตาราง 4.13 รายละเอียดของฟังก์ชันสมาชิกของคลาส Exec8

ชื่อฟังก์ชันสมาชิก	หน้าที่
getline	นำภาษาเครื่องมาหนึ่งบรรทัด
initialise	กำหนดค่าเริ่มต้นก่อนการทำการจำลองการทำงานตามคำสั่งภาษาเครื่อง
addressing	หาค่าของตำแหน่งของหน่วยความจำที่ถูกอ้างในคำสั่งภาษาเครื่อง
and	ดำเนินการตามคำสั่ง and
tad	ดำเนินการตามคำสั่ง tad
isz	ดำเนินการตามคำสั่ง isz
dca	ดำเนินการตามคำสั่ง dca
jms	ดำเนินการตามคำสั่ง jms
jmp	ดำเนินการตามคำสั่ง jmp
rotate_left	ดำเนินการตามคำสั่ง rotate left
rotate_right	ดำเนินการตามคำสั่ง rotate right
opr	ดำเนินการตามคำสั่งต่าง ๆ ที่อยู่ในกลุ่มคำสั่ง operate
intcode	ดำเนินการตามคำสั่ง interrupt
keybcode	ดำเนินการตามคำสั่งในกลุ่มที่ติดต่อกับแป้นพิมพ์
ttypecode	ดำเนินการตามคำสั่งในกลุ่มที่ติดต่อกับ teletype
diskseekcode	ดำเนินการตามคำสั่งในกลุ่ม disk seek
disktranscode	ดำเนินการตามคำสั่งในกลุ่ม disk transfer
clockcode	ดำเนินการตามคำสั่งในกลุ่มของนาฬิกา
iot	ดำเนินการแยกกลุ่มคำสั่งต่าง ๆ ในการติดต่อกับอุปกรณ์ภายนอก
Operation	ดำเนินการแยกกลุ่มคำสั่งต่าง ๆ ในกลุ่ม operate
Run	ดำเนินการจำลองการทำงานตามคำสั่งภาษาเครื่อง
IntToStr4	กำหนดให้แสดงค่าตามเลขฐานแปดให้ครบจำนวนสี่หลัก
ShowCPU	แสดงค่าต่าง ๆ ภายในหน่วยประมวลผลกลางจำลอง
Go	ดำเนินการทำงานตามเมนูคำสั่ง go
Dump	ดำเนินการทำงานตามเมนูคำสั่ง dump
Step	ดำเนินการทำงานตามเมนูคำสั่ง step

ตาราง 4.14 รายละเอียดของตัวแปรสมาชิกของคลาส MyFilter

ชื่อตัวแปรสมาชิก	หน้าที่
txt	เป็นค่าคงที่ชนิดข้อมูลอักขระที่เก็บอักขระที่เป็นส่วนขยายของแฟ้มที่เป็นค่าปริยาย คือ txt

ตาราง 4.15 รายละเอียดของฟังก์ชันสมาชิกของคลาส MyFilter

ชื่อฟังก์ชันสมาชิก	หน้าที่
getExtension	หาส่วนขยายของแฟ้มข้อมูล
getDescription	กำหนดอักขระข้อความปริยายของแฟ้มข้อมูลทุกแฟ้มที่มีส่วนขยายเป็น txt
accept	ตรวจสอบส่วนขยายของแฟ้มข้อมูล

ตาราง 4.16 รายละเอียดของตัวแปรสมาชิกของคลาส PSUSim8

ชื่อตัวแปรสมาชิก	หน้าที่
bar	กำหนดให้มีเมนูบาร์
fileMenu	กำหนดเมนูหลัก File
newItem	กำหนดเมนูย่อย New
openItem	กำหนดเมนูย่อย Open
closeItem	กำหนดเมนูย่อย Close
saveItem	กำหนดเมนูย่อย Save
saveasItem	กำหนดเมนูย่อย Save As...
pageItem	กำหนดเมนูย่อย Page Setup
printItem	กำหนดเมนูย่อย Print
quitItem	กำหนดเมนูย่อย Quit
editMenu	กำหนดเมนูหลัก Edit
cutItem	กำหนดเมนูย่อย Cut
copyItem	กำหนดเมนูย่อย Copy
pasteItem	กำหนดเมนูย่อย Paste

ตาราง 4.16 รายละเอียดของตัวแปรสมาชิกของคลาส PSUSim8 (ต่อ)

ชื่อตัวแปรสมาชิก	หน้าที่
clearItem	กำหนดเมนูย่อย Clear
selectItem	กำหนดเมนูย่อย Select All
fontMenu	กำหนดเมนูหลัก Font
monospacedItem	กำหนดเมนูย่อย Monospaced
timesromanItem	กำหนดเมนูย่อย Times Roman
sansserifItem	กำหนดเมนูย่อย SansSerif
s10Item	กำหนดเมนูย่อย 10
s12Item	กำหนดเมนูย่อย 12
s18Item	กำหนดเมนูย่อย 18
s24Item	กำหนดเมนูย่อย 24
s48Item	กำหนดเมนูย่อย 48
plainItem	กำหนดเมนูย่อย PLAIN
boldItem	กำหนดเมนูย่อย BOLD
italicItem	กำหนดเมนูย่อย ITALIC
viewMenu	กำหนดเมนูหลัก View
sourcecodeItem	กำหนดเมนูย่อย Source Code
objectcodeItem	กำหนดเมนูย่อย Object Code
listingItem	กำหนดเมนูย่อย Listing
assembleMenu	กำหนดเมนูหลัก Assemble
assembleItem	กำหนดเมนูย่อย Assemble
assemblelistItem	กำหนดเมนูย่อย Assemble & List
runMenu	กำหนดเมนูหลัก Run
resetItem	กำหนดเมนูย่อย Reset
dumpItem	กำหนดเมนูย่อย Dump
breakItem	กำหนดเมนูย่อย Breakpoints
goItem	กำหนดเมนูย่อย Go

ตาราง 4.16 รายละเอียดของตัวแปรสมาชิกของคลาส PSUSim8 (ต่อ)

ชื่อตัวแปรสมาชิก	หน้าที่
stepItem	กำหนดเมนูย่อย Step
stepstepItem	กำหนดเมนูย่อย Step-Step
helpMenu	กำหนดเมนูหลัก Help
aboutItem	กำหนดเมนูย่อย About
t	เป็นพื้นที่แสดงชุดอักขระที่จะให้แสดงให้กับผู้ใช้โปรแกรม
file_name	เป็นตัวแปรชุดอักขระที่เก็บชื่อแฟ้มข้อมูล
smap_text	เป็นตัวแปรชุดอักขระที่เก็บผลลัพธ์การการแปลภาษาแอสเซมบลี
SourceText	เป็นตัวแปรชุดอักขระที่เก็บโปรแกรมภาษาแอสเซมบลีก่อนทำการแปลเป็นภาษาเครื่อง
ObjectText	เป็นตัวแปรชุดอักขระที่เก็บโปรแกรมภาษาเครื่องที่ได้จากการแปลภาษาแอสเซมบลี
ListingText	เป็นตัวแปรชุดอักขระที่เก็บรายละเอียดต่าง ๆ ที่ได้จากการแปลภาษาแอสเซมบลี
DumpText	เป็นตัวแปรชุดอักขระที่เก็บค่าภายในหน่วยความจำหลักจำลองที่ได้หลักจากดำเนินงานทำการจำลองการทำงานตามภาษาเครื่องที่ได้จากการแปล

ตาราง 4.17 รายละเอียดของฟังก์ชันสมาชิกของคลาส PSUSim8

ชื่อฟังก์ชันสมาชิก	หน้าที่
openfile	เปิดแฟ้มข้อมูล
savechanged	บันทึกแฟ้มข้อมูล กรณีที่ปิดการทำงานของโปรแกรมแต่ข้อมูลที่อยู่ภายใน text editor มีการเปลี่ยนแปลง
save	บันทึกแฟ้มข้อมูล
saveas	บันทึกแฟ้มข้อมูลโดยสอบถาชื่อแฟ้มข้อมูลที่จะทำการบันทึก
setfont	กำหนดรูปแบบอักขระที่จะแสดง
exitPSUSim8	ออกจากโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์

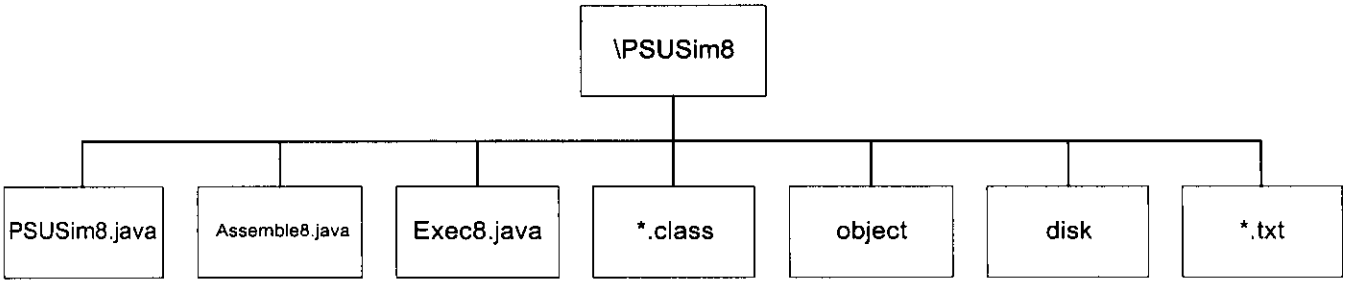
ตาราง 4.17 รายละเอียดของฟังก์ชันสมาชิกของคลาส PSUSim8 (ต่อ)

ชื่อฟังก์ชันสมาชิก	หน้าที่
PSUSim8	ฟังก์ชันการทำงานหลักของโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์
ActionPerformed	ตรวจสอบการเลือกเมนูการทำงานและเรียกฟังก์ชันการทำงานตามเมนูที่เลือก

4.5 สารบบแฟ้มข้อมูล

สารบบแฟ้มข้อมูล (File Directory) ที่ใช้ในการจัดเก็บแฟ้มข้อมูลทั้งหมดที่ได้พัฒนาขึ้นเป็นแบบแผนผังต้นไม้ (Tree Diagram) แสดงได้ดังภาพประกอบ 4.25 โดยมีรายละเอียดดังนี้

- **สารบบ** ***\PSUSim8*** เป็นที่ซึ่งใช้ในการจัดเก็บโปรแกรมทั้งหมดที่ได้ทำการพัฒนาขึ้น
- **แฟ้มข้อมูล** ***PSUSim8.java*** เป็นแฟ้มข้อมูลที่ใช้เก็บโปรแกรมต้นฉบับที่เป็นฟังก์ชันการทำงานในส่วนที่ใช้ติดต่อกับผู้ใช้และการแสดงผล
- **แฟ้มข้อมูล** ***Assemble8.java*** เป็นแฟ้มข้อมูลที่ใช้เก็บโปรแกรมต้นฉบับที่เป็นฟังก์ชันการทำงานในส่วนที่ใช้ในการแปลภาษาแอสเซมบลี PDP-8 ให้เป็นภาษาเครื่อง
- **แฟ้มข้อมูล** ***Exec8.java*** เป็นแฟ้มข้อมูลที่ใช้เก็บโปรแกรมต้นฉบับที่เป็นฟังก์ชันการทำงานในส่วนที่ใช้ในจำลองการทำงานภาษาเครื่องบนหน่วยความจำหลักที่ได้จำลองขึ้น
- **แฟ้มข้อมูล** ****.class*** เป็นแฟ้มข้อมูลที่ได้จากการคอมไพล์โปรแกรมต้นฉบับให้เป็น Byte Code เพื่อใช้ในการทำงานบนระบบปฏิบัติการต่าง ๆ
- **แฟ้มข้อมูล object** เป็นแฟ้มข้อมูลที่ใช้เก็บโปรแกรมภาษาเครื่อง PDP-8 จากการแปลโปรแกรมภาษาแอสเซมบลี
- **แฟ้มข้อมูล disk** เป็นแฟ้มข้อมูลที่ใช้จำลองข้อมูลที่เก็บใน disk ที่เป็นหน่วยรับเข้า/ส่งออกของ PDP-8
- **แฟ้มข้อมูล** ****.txt*** เป็นแฟ้มข้อมูลที่ใช้เก็บโปรแกรมภาษาแอสเซมบลีของ PDP-8 เพื่อใช้ในการทดสอบการทำงานของโปรแกรมจำลองการทำงานของระบบคอมพิวเตอร์



ภาพประกอบ 4.25 สารบบแฟ้มข้อมูลของโปรแกรมจำลองฯ