บทที่ 8

การประมวลผลจอภาพและคีย์บอร์ด

INTRODUCTION TO SCREEN AND KEYBOARD PROCESSING

วัตถุประสงค์

หลังจากที่ท่านศึกษาบทนี้ท่านจะมีความเข้าใจดังต่อไปนี้

- ความหมายของการอินเตอร์รัพท์
- การอินเตอร์รัพท์ของดอส
- การอินเตอร์รัพท์ของไบออส
- การเขียนโปรแกรมภาษาแอสแซมบลีร่วมกับอินเตอร์รัพท์

CT 215 185

บทที่ 8
การประมวลผลจอภาพและคีย์บอร์ด
INTRODUCTION TO SCREEN AND KEYBOARD PROCESSING

บทนำ

ในบทนี้โปรแกรมจะต้องมีการกำหนดรายการข้อมูลในส่วนพื้นที่ของข้อมูล หรือข้อมูลที่มากับคำสั่ง การทำงานส่วนมากโปรแกรมจะต้องการรับข้อมูลจากคีย์บอร์ด ดิสก์หรือรอมเดม และผลลัพธ์แสดงที่เครื่องพิมพ์หรือดิสก์ บทนี้จะอธิบายพื้นฐานการแสดงผลข่าวสารบนจอภาพ และรับข้อมูลจากคีย์บอร์ด

การรับและการแสดงผลจะต้องบอกระบบว่า การประมวลผลของ INPUT หรือ OUTPUT และอุปกรณ์อื่นๆ คำสั่ง INT(interrupt) จะเป็นตัวกำหนด INPUT/OUTPUT คำสั่ง INT 2 ชนิด คือ BIOS ใช้คำสั่งINT 10H สำหรับแสดงผลจอภาพ และการทำงานของ DOS คือคำสั่ง INT 21H สำหรับการแสดงผล output และรับข้อมูล การทำงานเหล่านี้เป็นการบริการ (serviceหรือ function) การเรียกการทำงานเพียงแต่เซ็ทค่าเริ่มแรกให้กับรีจิสเตอร์ AH ซึ่งเป็นตัวกำหนดชนิดการทำงาน

ท่านสามารถใช้คำสั่ง INT 10H กับการแสดงผลที่จอภาพ และคำสั่ง INT 16H ในการรับข้อมูลจากคีย์บอร์ด ซึ่งการทำงานทั้งสองเป็นการเคลื่อนย้ายโดยตรงไปที่ BIOS อย่างไรก็ตาม การทำงานของมันก็มีความยุ่งยากพอสมควร คำสั่ง INT 21H เป็นการเข้าถึงในระดับที่สูงกว่าของคำสั่งอินเตอร์รัพท์ ซึ่งเป็นการเคลื่อนย้ายการควบคุมไปยัง DOS สำหรับตัวอย่างการรับข้อมูลจากคีย์บอร์ด อาจต้องการนับจำนวนตัวอักขระที่ป้อนเข้ามา ตรวจสอบค่าสูงสุดของตัวอักขระ และตรวจสอบตัวอักขระที่ป้อนเข้ามา ในส่วนของ DOS INT 21H การทำงานจะรวมถึงการประมวลผลและการเคลื่อนย้ายโดยอัตโนมัติไปที่ BIOS สำหรับการทำงานระดับต่ำกว่า

เพื่อความสะดวกหนังสือเล่มนี้จะอ้างถึง 0DH ในการป้อนสำหรับคีย์บอร์ด และ Carriage Return สำหรับจอภาพและเครื่องพิมพ์

อินเตอร์รัพท์ INTERRUPTS

การอินเตอร์รัพท์จะเกิดขึ้นเมื่อมีการเอ็กซีคิวส์โปรแกรม การอินเตอร์รัพท์เป็นการขัดจังหวะโปรแกรมเพื่อที่จะทำงานตามเหตุผลต่างๆที่ต้องการ ปกติจะใช้ในการทำงานของอุปกรณ์รอบข้างเช่น คีย์บอร์ด DISK DRIVE หรือเครื่องพิมพ์ ตัว CPU ของ INTEL จะมีการอินเตอร์รัพท์อยู่ 2 ชนิดคือ

ฮาร์ดแวร์อินเตอร์รัพท์ Hardware Interrupt

คือจะมีการจ่ายสัญญานเมื่ออุปกรณ์รอบข้างต้องการข้อมูลจาก CPU ส่วน Software Interrupt คือการเรียกโปรแกรมย่อยของระบบปฏิบัติการ ปกติใช้กับงานอินพุตเอ้าพุต

KEYBOARD เป็นอุปกรณ์อินพุตชนิดหนึ่ง ของตัวอย่างในการทำงาน Hardware Interrupt เมื่อมีการกดคีย์ Hardware Interrupt ก็จะทำงานโดยมีชิพ 8259 Interrupt Controller ทำหน้าที่ควบคุมการทำงานของการอินเตอร์รัพท์ ตัว CPU จะให้โปรแกรมหยุดทำงานชั่วคราวและเอ็กซีคิวส์โปรแกรมย่อยทำหน้าที่เก็บตัวอักขระในคีย์บอร์ด เมื่อสิ้นสุดการทำงานโปรแกรมย่อย มันก็จะย้อนกลับมาทำงานที่โปรแกรมเดิมอีกครั้ง

SOFTWARE INTERRUPT การทำงานของอินเตอร์รัพท์ชนิดนี้ไม่สามารถทำการขัดจังหวะการทำงานทั้งหมดได้เพราะซอพท์แวร์อินเตอร์รัพท์เป็นเครื่องมือช่วยในการติดต่อ อินพุต เอ้าพุต ของโปรแกรมประยุกต์โดยทำงานร่วมกับการบริการของระบบปฏิบัติการ การบริการเหล่านี้จะเป็นโปรแกรมเล็กๆ อยู่ใน ROM BIOS และอยู่ในส่วนของ Resident portion of DOS

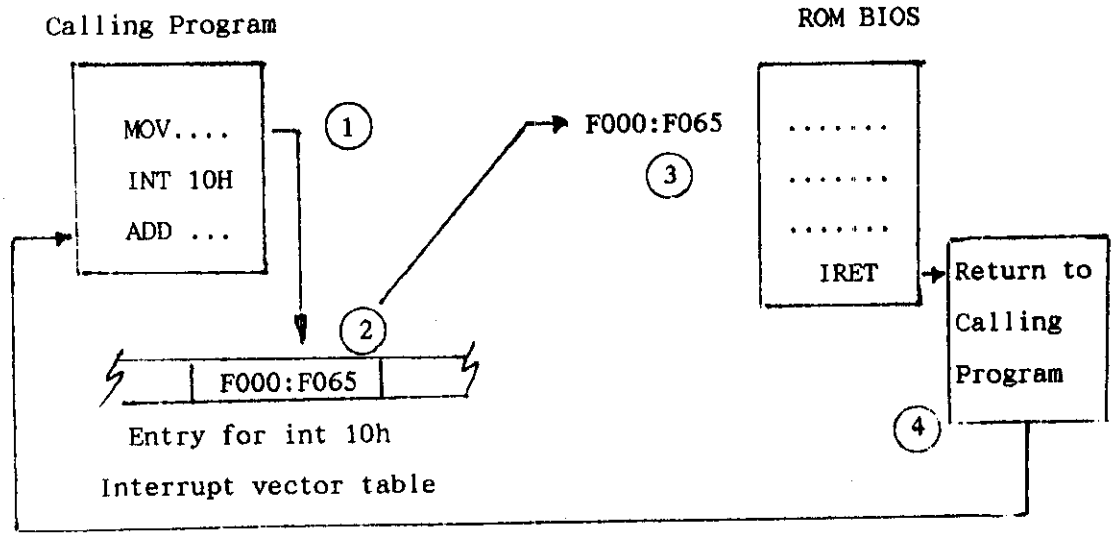
คำสั่ง INT

คำสั่ง INT ใช้สำหรับเรียกงานย่อยของระบบปฏิบัติการ โดยกำหนดฟังชั่นตัวเลขในช่วง 00-0FFH ก่อนที่จะใช้คำสั่ง INT ทำการเอ็กซีคิวส์ รีจีสเตอร์ AH ใช้เก็บค่าฟังชั่นตัวเลขก่อนที่จะใช้คำสั่ง INT ทำการเอ็กซีคิวส์ เพื่อเรียกโปรแกรมย่อยของระบบปฏิบัติการ มีรูปแบบดังนี้

```
INT   number
```

เมื่อมีการใช้คำสั่ง INT สำหรับการติดต่อ I/O , File และการจัดการ Video งานบริการเหล่านี้เป็นส่วนหนึ่งของ BIOS และ DOS

INTERRUPT VECTOR TABLE: ตัว CPU จะประมวลผลคำสั่งอินเตอร์รัพท์เพื่อการเรียกใช้ตารางอินเตอร์รัพท์เวคเตอร์ จากตารางแอดเดรสที่ต่ำสุด 1024 ไบท์ของหน่วยความจำ แต่ละจุดในตารางมีขนาด 32 บิต ของเซคเม้นต์ออฟเซ้ตแอดเดรส เพื่อกำหนดการทำงานในระบบปฏิบัติการ



จากรูปเป็นการแสดงขั้นตอนการทำงานในการเอ็กซีคิวส์คำสั่ง INT

1) ฟังชั่นหมายเลขการทำงานของคำสั่งเอ็กซีคิวส์ จะบอก CPU ถึงจุดเข้าใน Interrupt vector Table ดังที่แสดงในรูปการใช้ INT 10H (VIDEO SERVICE)
2) CPU จะโดดไปที่แอดเดรสที่เก็บใน Interrupt vector table (F000:F065)
3) จะใช้งาน VIDEO SERVICE จาก ROM BIOS ที่ตำแหน่ง F000:F065 เริ่มต้นเอ็กซีคิวส์คำสั่ง และสิ้นสุดการทำงานที่คำสั่ง IRET
4) คำสั่ง IRET (INTERRUPT RETURN) จะกระโดดไปโปรแกรมหลักเพื่อการเอ็กซีคิวส์คำสั่งต่อไป หลังจากคำสั่งเดิมที่กระโดดกลับมา

ซอฟแวร์อินเตอร์รัพท์ COMMON SOFTWARE INTERRUPTS

ซอฟแวร์อินเตอร์รัพท์ที่ใช้เรียกโปรแกรมย่อยใน ROM BIOS หรือหน่วยความจำที่เป็น Resident ของ DOS ที่ใช้บ่อยๆดังต่อไปนี้

-INT 10H : VIDEO SERVICES เป็นการแสดงผลทางจอภาพ ควบคุมตำแหน่ง Cursor เคลียจอภาพ แสดงกราฟฟิค Scroll the screen
-INT 16H : KEYBOARD SERVICES ใช้ในการรับข้อมูลจากคีย์บอร์ดและตรวจสอบสถานะ
-INT 17H : PRINTER SERVICES เซ็ตค่าการพิมพ์ และตรวจสอบสถานะเครื่องพิมพ์

-INT 21H : DOS SERVICES การบริการของ DOS สำหรับอินพุตเอ้าพุต File การจัดการหน่วยความจำ

DOS FUNCTION CALLS

INT 21H คือการบริการของ DOS FUNCTION CALLS มีฟังชั่นที่แตกต่างกัน 87 ฟังชั่น โดยการใช้ INT 21H นี้ ฟังชั่นที่แตกต่างกันทั้งหมดจะเป็นตัวกำหนดฟังชั่นการทำงานที่จะอยู่ใน AH ดังตัวอย่างฟังชั่น 00 - 0CH การทำงานของฟังชั่น DOS ส่วนมากจะมีการตรวจสอบคีย์ CTRL - BREAK ถ้ามีการกดคีย์นี้ โปรแกรมจะหยุดทำงานของโปรแกรม

01H : CONSOLE INPUT WITH ECHO

ดอสฟังชั่น 1 เป็นการอรับตัวอักขระที่ส่งมาจากคีย์บอร์ดและข้อมูลนี้จะเก็บในรีจีสเตอร์ AL การกดคีย์ Ctrl-Break มีผลต่อการทำงาน ตัวอักขระที่รับมานั้นจะแสดงผลที่จอภาพ จากตัวอย่างคำสั่งที่ใช้ในการรับข้อมูล 1 ตัวอักษร และเก็บข้อมูลไว้ในตัวแปร CHAR:

```
MOV   AH,1        ;CONSOLE INPUT FUNCTION
INT   21H         ;CALL DOS ,KEYBOARD IN AL
MOV   CHAR,AL     ;SAVE THE CHARACTER
```

02H : CHARACTER OUTPUT

ดอสฟังชั่น 2 เป็นการส่งตัวอักขระไปที่ Console การกดคีย์ Ctrl-Break มีผลต่อการทำงาน ตัวอักขระที่จะส่งเก็บไว้ใน DL ดังตัวอย่าง

```
MOV       AH,2        ;SELECT DOS FUNCTION
MOV       DL,'*'      ;CHARACTER TO BE DISPLAY
INT       21H
```

ถ้า AL อาจมีการเปลี่ยนแปลงได้ถ้าการทำงานของ DOS ระหว่างการเรียกใช้ INT 21H ดังนั้นเพื่อความแน่ใจ ในการเก็บค่าของ AL ให้คงเดิม เขียนคำสั่งดังนี้

```
        MOV     BYTEVAL,AL      ;SAVE CONTAINS OF AL
        MOV     AH,2
        MOV     DL,'*'
        INT     21H
        MOV     AL,BYTEVAL
```

ตัวอย่าง 8.1 การเขียนโปรแกรมอ่านตัวอักขระและแสดงผลในบรรทัดต่อไป

```
        TITLE   PROGRAM ECHO
        .MODEL SMALL
        .STACK 100H
        .CODE
        MAIN    PROC
;DISPLAY PROMPT
        MOV  AH,2       ;DISPLAY CHARACTER FUNCTION
        MOV  DL,'?'     ;CHARACTER IS '?'
        INT  21H
;INPUT CHARACTER
        MOV  AH,1       ;READ CHARACTER FUNCTION
        INT  21H        ;CHARACTER IN AL
        MOV  BL,AL      ;SAVE IT IN BL
;GO TO NEW LINE
        MOV  AH,2
        MOV  DL,0DH     ; CARRIAGE RETURN
        INT  21H
        MOV  DL,0AH     ;LINE FEED
        INT  21H
;DISPLAY CHARACTER
        MOV  DL,BL      LRETRIVED CHARACTER
        INT  21H
;RETURN TO DOS
        MOV  AH,4CH     ;DOS EXIT FUNCTION
MAIN    ENDP
        END  MAIN
```

05H : PRINTER OUTPUT

การพิมพ์ตัวอักขระที่อยู่ในรีจีสเตอร์ DL โดยการเรียกการบริการ 5 ดอสจะทำงานด้วยการคอยจนกระทั่งเครื่องพิมพ์พร้อมและรับตัวอักขระ ถ้าเครื่องพิมพ์ไม่พร้อมท่านสามารถใช้ CTRL-BREAK การทำงานนี้จะกำหนดที่เอ้าพุตเครื่องพิมพ์เป็น 1 (LTP 1) โดยการส่งรหัส CR (0DH) เพื่อกำหนดการพิมพ์ เครื่องพิมพ์จำนวนมากจะเก็บตัวอักขระในบัฟเฟอร์จนกระทั่งบัฟเฟอร์เต็ม หรือ CR เพื่อพิมพ์ ดังตัวอย่าง

```
MOV  AH,5       ;SELECT PRINTER OUTPUT
MOV  DL,'$'     ;CHARACTER TO BE PRINTED
INT  21H
MOV  DL,0DH     ;CR
INT  21H
```

06H : DIRECT CONSOLE INPUT OUTPUT

ดอสฟังชั่น 6 เป็นการอ่านหรือเขียนข้อมูลจาก CONSOLE การทำงานของ CTRL - BREAK ไม่มีผลการทำงานในการควบคุมตัวอักขระอาจจะถูกอ่านหรือเขียนปราศจากการทำงานเช่น CR , TAB และอื่นๆ ถ้าต้องการเป็นอินพุต DL = 0FFH ดอสจะไม่รอคอยตัวอักขระที่กด แต่จะนำตัวอักขระไปไว้ใน AL ถ้ามีการคอยข้อมูลในบัฟเฟอร์ของคีย์บอร์ดและไม่มีตัวอักขระ ZF = 1 ถ้าต้องการเป็นอินพุต DL = ตัวอักขระที่แสดง ดังตัวอย่าง

CHARACTER INPUT

```
MOV  AH,6       ;REQUEST DOS FUNCTION 6
MOV  DL,0FFH    ;LOOK FOR INPUT DON,T WAIT
INT  21H        ;IF KEY PRESSED AL = CHARACTER
                ;OTHERWISE ZF = 1
```

CHARACTER OUTPUT

```
MOV  AH,6       ;REQUEST DOS FUNCTION 6
MOV  DL,'$'     ;CHARACTER TO BE OUTPUT
INT  21H
```

07H: DIRECT CONSOLE INPUT

ดอสฟังชั่น 7 เป็นการรับตัวอักขระที่มีการรอคอยที่มาจากคอนโซล ตัวอักขระจะไม่ ECHO และ CTRL-BREAK ไม่มีผล ฟังชั่นการรับอินพุตชนิดนี้ เหมาะที่จะใช้กับคีย์พิเศษเช่น ARROW KEY

```
MOV  AH,7        ;INPUT FUNCTION
INT  21H
MOV  CHAR,AL     ;SAVE CHARACTER
```

08H : CONSOLE INPUT WITHOUT ECHO

การทำงานของฟังชั่น 8 เป็นการรับข้อมูลตัวอักขระที่มาจาก CONSOLE แต่ไม่แสดงผล การกดคีย์ CTRL-BREAK จะมีผลต่อการทำงาน ข้อมูลที่รับเข้ามาจะเก็บไว้ใน AL

```
        MOV  AH,8        ;INPUT FUNCTION
        INT  21H
        MOV  CHAR,AL     ;SAVE CHARACTER
        ....

AGAIN:  MOV  AH,8        ;CONSOLE INPUT WITHOUT ECHO
        INT  21H
        CMP  AL,'Y'
        JE   EXIT
        CMP  AL,'y'
        JE   EXIT
        CMP  AL,'N'
        JE   EXIT
        CMP  AL,'n'
        JNE  AGAIN
EXIT:   AND  AL,0DFH
        MOV  AH,2        ;CONSOLE CHARACTER OUTPUT
        MOV  DL,AL
        INT  21H
```

09H : STRING OUTPUT

ดอสฟังชั่น 9 แสดงตัวอักขระชุดสตริงบนคอนโซล ค่าออฟเซ็ตแอดเดรสของชุดสตริงจะเก็บไว้อยู่ในรีจีสเตอร์ DX และการสิ้นสุดของชุดสตริงด้วยตัวอักขระ $ (Dollar sign) การควบคุมของตัวอักขระเช่น TAB และ CR จะมีผลต่อดอส ดังตัวอย่าง

```
        MOV  AH,9        ;STRING OUTPUT FUNCTION
        MOV  DX,OFFSET STRING ; OFFSET ADDRESS OF THE STRING
        INT  21H
        ---
STRING  DB   'THIS IS A BYTE STRING.'0AH,0DH,'$'
```

ตัวอย่าง 8.2 โปรแกรมการพิมพ์สตริงโดยใช้ฟังชั่น 9

```
                TITLE  PRINT STRING PROGRAM
                .MODEL SMALL
                .STACK 100H
                .DATA
MSG             DB   'HELLO!$'
                .CODE
                MAIN       PROC
  ;INITIALIZE DS
                MOV  AX,@DATA
                MOV  DS,AX
  ;DISPLAY MESSAGE
                LEA  DX,MSG
                MOV  AH,9
                INT  21H
    ;RETURN TO DOS
                MOV  AH,4CH
                INT  21H
             MAIN ENDP
                END  MAIN
```

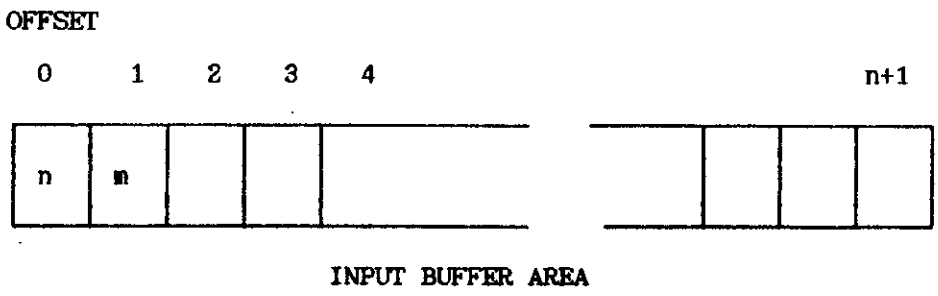
ตัวอย่าง 8.3 โปรแกรมในการเปลี่ยนตัวอักษรตัวเล็กเป็นตัวใหญ่

```
                TITLE  CASE CONVERSION PROGRAM
                .MODEL SMALL
                .STACK 100H
                .DATA
CR              EQU     0DH
LF              EQU     0AH
MSG1            DB      'ENTER A LOWER CASE LETTER:$'
MSG2            DB      '0DH,0AH,'IN UPPER CASE IT IS: '
CHAR            DB      ?,'$'
                .CODE
                MAIN    PROC
                MOV  AX,@DATA
                MOV  DS,AX
 ;PRINT USER PROMPT
                LEA  DX,MSG1    ;GET FIRST MESSAGE
                MOV  AH,9       ;DISPLAY STRING FUNCTION
                INT  21H
;INPUT CHARACTER AND CONVERT TO UPPER CASE
                MOV  AH,1       ;READ CHARACTER FUNCTION
                INT  21H        ;CHARACTER IN AL
                SUB  AL,20H     ;CONVERT TO UPPER CASE
                MOV  CHAR,AL      ;SAVE IT
;DISPLAY ON THE NEXT LINE
                LEA  DX,MSG2    ;GET SECOND MESSAGE
                MOV  AH,9       ;DISPLAY STRING FUNCTION
                INT  21H        ;DISPLAY MESSAGE AND UPPER CASE
;RETURN TO DOS
                MOV  AH,4CH     ;DOS EXIT FUNCTION
MAIN            ENDP
                END  MAIN
```

0AH : BUFFERED CONSOLE INPUT

การทำงานของฟังชั่น 10 (0AH) สามารถอ่านตัวอักขระสตริงได้สูงถึง 256 ตัว จากคอนโซล และนำไปเก็บไว้ในบัฟเฟอร์ คีย์ BS อาจจะใช้ในการลบตัวอักขระและเลื่อนเคอร์เซอร์ ท่านสามารถสิ้นสุดสตริงด้วยการกด ENTER คอสจะทำการตรวจสอบ EXTENDED KEYS ออกเช่น ARROW KEYS ,PAGEUP PAGEDOWN และตัวอักขระทั้งหมดที่แสดงผลบนคอนโซล

ก่อนที่ฟังชั่นจะถูกเรียก DX จะต้องมีค่าออฟเซ็ตของคีย์บอร์ดมีรูปแบบดังต่อไปนี้



INPUT BUFFER AREA

MAXIMUM NUMBER OF CHARACTERS

NUMBER OF CHARACTER ACTUALLY INPUT

ตัวอย่าง 8.4 ต่อไปนี้จะกำหนด MAX_KEYS = 32 , CHARS_INPUT ใส่โดยการใช้ดอสและค่าของ BUFFER เก็บค่า INPUT CHARACTER

```
                MOV   AH,0AH              ;CONSOLE INPUT
                MOV   DX,OFFSET MAXKEYS   ;DX POINTS TO KEYBOARD AREA
                INT   21H
                ---
MAX_KEYS        DB    32                  ;MAX CHARACTER
CHARS_INPUT     DB    ?                   ;CHARACTER ACTUALLY INPUT
BUFFER_INPUT    DB    32 DUP(0)           ;HOLDS THE INPUT
```

สมมุติเราป้อนข้อมูล 21 ตัวจากคอนโซลด้วยค่า My name is kip irvine ถ้าเรา DUMP บัฟเฟอร์หลังจากที่เรารับข้อมูลไปแล้ว แสดงไบท์ออฟเซ็ต = 0001 จำนวนของตัวอักขระที่รับ (15H) ตามด้วย INPUT CHARACTER ดังนี้

```
MAX_KEYS      CHARS_INPUT
    20          15
     4D  79  20  6E  61  6D  65  20  69  73  20  My name is
     4B  69  70  20  49  72  76  69  6E  65  0D  kip Irvine
                                                 |
                                                 CR
```

จอภาพ (THE SCREEN)

การทำงานของจอภาพ เราสามารถกำหนดตำแหน่งทุกๆจุดบนจอภาพโดยใช้ cursor ขนิดของจอภาพ เช่น มอนิเตอร์ชนิด 25 บรรทัด(0-24) และ 80 คอลัมน์ (0-79) ดังตัวอย่างของตำแหน่งcursorต่อไปนี้

LOCATION	DECIMAL FORMAT		HEX FORMAT	
	ROW	COLUMN	ROW	COLUMN
UPPER LEFT CORNER	00	00	00	00
UPPER RIGHT CORNER	00	79	00	4F
CENTER OF SCREEN	12	39/40	0C	27/28
LOWER LEFT CORNER	24	00	18	00
LOWER RIGHT CORNER	24	79	18	4F

ระบบจะเป็นตัวจัดหาพื้นที่ว่างในหน่วยความจำสำหรับบัฟเฟอร์ของจอภาพ จอภาพชนิดโมโนโครม(monochrome)จะมีแอดเดรสหน่วยความจำเริ่มต้นที่ B0000H ของ BIOS และมีหน่วยความ

จำสนับสนุน 4K จำนวน 2K สำหรับตัวอักขระและอีก 2K สำหรับ attribute ของอักขระแต่ละตัวที่กำหนดการทำงาน reverse video,blinking,high intensity และ underlining

จอภาพชนิดcolor/graphic จะมีหน่วยความจำสนับสนุน 16K-byte แอดเดรสเริ่มต้นที่ B8000H ท่านสามารถประมวลผลใน text mode โดยแสดงผลที่จอภาพเป็น page ชนิด 80 คอลัมน์และ 40 คอลัมน์ โดยค่าที่กำหนด 0 ถึง 3 เป็นชนิด 80 คอลัมน์ และ 0-7 สำหรับชนิด 40 คอลัมน์

B8000 :	PAGE 0	PAGE 1	PAGE 2	PAFE 3

การกำหนดจำนวนเพจเป็น 0 ซึ่งเป็นรูปแบบที่จะแสดงผลทุกเพจในหน่วยความจำ แต่ละเพจจะมีขนาด 4K ซึ่ง 2K แรกจะเป็นข้อมูลของ 25 บรรทัด 80 คอลัมน์ของตัวอักขระ และอีก 2K สำหรับ attribute ของตัวอักขระบนจอภาพ

การอินเตอร์รัพท์การแสดงผลของจอภาพ เป็นการเคลื่อนย้ายข้อมูลโดยตรงไปยังหน่วยความจำขึ้นอยู่กับชนิดของจอภาพ เช่น CGA,EGA,VGA ถึงแม้ว่าเทคนิคในการเขียนโปรแกรม อาจจะเคลื่อนย้ายข้อมูลโดยตรงไปยังพื้นที่หน่วยความจำของจอภาพ อย่าลืมว่าการทำงานเหล่านี้จะต้องใช้ร่วมกับคำสั่ง INT เช่น INT 10H,INT 21H ท่านสามารถใช้ INT 10H แสดงผลและเซ็ทตำแหน่งของ cursorและการเคลียร์จอภาพ และคำสั่ง INT 21H ใช้แสดงผล

การกำหนดตำแหน่งเคอร์เซอร์ (SETTING THE CURSOR)

การกำหนดตำแหน่งของ cursor ส่วนมากจะมีการกระทำใน text mode การกำหนดตำแหน่งนั้นเพื่อแสดงผลอักขระตัวต่อไป คำสั่ง INT 10H คือการทำงานของ BIOS สำหรับจอภาพ และใช้การบริการ 02 ในAH เพื่อบอกการกำหนดตำแหน่งของ cursor เพจ(หรือจอภาพ)ปกติจะกำหนด 0 ใน BH และ row,column จะต้องอยู่ใน DX ข้อมูลของรีจิสเตอร์อื่นๆ ไม่มีส่วนสำคัญ ดังตัวอย่างในการเซ็ท cursorไปบรรทัดที่ 5 ใน คอลัมน์ 12

```
MOV     AH,2     ; REQUEST SET CURSOR
MOV     BH,00    ; PAGE NUMBER 0
MOV     DH,05    ; ROW 05
MOV     DL,12    ; COLUMN 12
INT     10H      ; INTERRUPT-EXIT TO BIOS
```

การเซ็ตบรรทัดและคอลัมน์ ท่านอาจใช้คำสั่ง MOV ดังนี้

```
MOV     DX,050CH  ; ROW 5,COLUMN 12
```

การเคลียจอภาพ (CLEARING THE SCREEN)

prompts และคำสั่งที่อยู่บนจอภาพจะรอคอยจนกระทั่ง มีการเขียนทับหรือ SCROLLED OFF เมื่อโปรแกรมของท่านเริ่มต้น execute ท่านอาจจะต้องการเคลียร์จอภาพ ท่านสามารถเคลียร์ตำแหน่งเริ่มต้นและตำแหน่งสุดท้าย คำสั่ง INT ของ BIOS จะใช้ INT 10H ในการเคลียร์จอภาพหรือ SCROLLING โดยการใช้การบริการที่ 06H ใน AH และ 00H ใน AL ในการเคลียร์จอภาพทั้งหมด ค่าของ attribute อยู่ใน BH แอดเดรสเริ่มต้นของ row:column ใน DX

attribute ในตัวอย่างต่อไปนี้ เป็นการเซ็ทตัวหนังสือสีขาวให้กับตัวอักขระบนพื้นสีดำ ถ้าจอภาพของท่านเป็นจอสี ท่านก็สามารถใช้ 71H สำหรับ สีน้ำเงิน(1)บนพื้น สีขาว(7)

```
MOV     AX,0060H      ; AH=6,AL=00
MOV     BH,07         ; ATTRIBUTE :WHITE ON BLACK
MOV     CX,0000       ; UPPER LEFT ROW,COLUMN
MOV     DX,184FH      ; LOWER RIGHT ROW,COLUMN
INT     10H           ; INTERRUPT TRANSFER TO BIOS
```

การทำงานจอภาพและคีย์บอร์ด SCREEN AND KEYBOARD OPERTIONS : ORIGINAL DOS

ORIGINAL DOS FOR SCREEN DISPLAY

การใช้บริการของ DOS ในการแสดงผลตามที่กำหนดของชุดสตริงในพื้นที่ของข้อมูล ชุดสตริงจะต้องปิดด้วยตัวอักขระ $(dollar sign) ดังตัวอย่าง

```
NAMPRMP     DB   'CUSTOMER NAME?','$'   ; DISPLAY STRING
```

เครื่องหมาย $ ที่ใช้ตามหลังข้อมูลนั้น เป็นจุดสิ้นสุดของข้อมูล ผลลัพธ์แสดงจะไม่มี $ หรือ $ อาจเขียนติดกับชุดสตริงก็ได้ 'CUSTOMER NAME$' หรือสามารถกำหนด $ในบรรทัดต่อไปก็ได้ เช่น DB '$'

การทำงานในการแสดงผลตัวอักขระนั้น ใช้การบริการ 09H ใน AH และในการโหลดแอดเดรสของชุดสตริงที่จะแสดงผลไว้ใน DX และใช้คำสั่ง INT 21H ดังตัวอย่าง

```
MOV     AH,09           ; REQUEST DISPLAY
LEA     DX,NAMPRMP      ; LOAD ADDRESS OF PROMPT
INT     21H             ; DOS INTERRUPT
```

คำสั่ง LEA(load effective address) เป็นการโหลดแอดเดรสของ NAMPRMP ไว้ใน DX เพื่อบอกตำแหน่งการทำงานในการแสดงผลชุดสตริง สำหรับแอดเดรสจริงคำสั่ง LEA เป็นการโหลดค่า offset address ของ NAMPRMP และ DOS จะใช้แอดเดรสนี้ไปที่ DS บวกกับค่า DX (DS:DX)

การใช้ DOS แสดงชุดอักขระ ASCII

ตัวอักขระ ASCII 256 ตัว จะมีสัญลักษณ์ในการแทนบนจอภาพ ยกเว้นค่า 00 และ FF สัญลักษณ์ที่ไม่มีการแสดงผลของ ASCII เช่น blank ใช้ 20H

จากตัวอย่างโปรแกรม 8-1 เป็นการแสดงผลของ ASCII ในโปรแกรมจะมีการ CALL 3 procedure คือ B10CLR , C10SET และ D10DISP โปรแกรมย่อย B10CLR เป็นการเคลียร์จอภาพ และ C10SET เป็นการเซ็ท cursor ที่ 00,00 ส่วน D10DISP แสดงข้อมูลตัวอักขระ ASCII ซึ่งเริ่มต้นที่ 00H จนถึง 0FFH

```
                page  60,132
TITLE           DOSASC  (COM) Use DOS to display ASCII
                                characters 00-FF
                .MODEL  SMALL
                .CODE
                ORG     100H
BEGIN:          JMP     SHORT MAIN
CHAR            DB      00,'$'
;---------------- Main procedure ----------------
```

```
MAIN    PROC    NEAR
        CALL    B10CLR    ; Clear screen
        CALL    C10SET    ; Set cursor
        CALL    D10DISP   ; Display characters
        MOV     AH,4CH    ; Exit
        INT     21H
MAIN    ENDP
;---------------- Clear screen ---------------------
B10CLR  PROC    NEAR
        MOV     AX,0600H  ; Scroll full screen
        MOV     BH,07     ; Attribute: white on black
        MOV     CX,0000   ; Upper left location
        MOV     DX,184FH  ; Lower right location
        INT     10H
        RET
B10CLR  ENDP
;---------------- Set cursor to 00,00 -------------
C10SET  PROC    NEAR
        MOV     AH,02     ; Request set cursor
        MOV     BH,00     ; Page number 0
        MOV     DX,0000   ; Row 0,column 0
        INT     10H
        RET
C10SET  ENDP
;---------------- Display ASCII characters ----------
D10DISP PROC
        MOV     CX,256    ; Initialize 256 iterations
        LEA     DX,CHAR   ; Initialize address of chars
D20:
        MOV     AH,09     ; Display ASCII char
        INT     21H
        INC     CHAR      ; Increment for next  character
```

```
            LOOP    D20         ; Decrement CX,loop nonzero
            RET
D10DISP     ENDP
            END     BEGIN
```

รูปที่ 8-1 DOS service to display the ASCII character set

ปัญหาของการใช้ DOS การบริการ 9 มันจะไม่แสดงผล $ ตัวอักขระพิเศษที่ใช้ในการควบคุม cursor ในการเคลื่อนย้าย ถ้าเป็น 07H เป็นกรณีของ beep 08H เท่ากับการเคลื่อนกลับ 1 ตำแหน่ง (backspace) 0AH เท่ากับ linefeed และ 0DH เท่ากับ carriage return

ตัวอย่าง โปรแกรมนี้เราสามารถ assemble link และเปลี่ยนโปรแกรมเป็น .COM สำหรับการ execute เมื่อใส่ชื่อแฟ้มข้อมูล เช่น B:ASCII.COM

การแสดงผลบรรทัดแรกเริ่มต้นที่ blank(00H) รูปใบหน้ายิ้ม 2 รูป "HAPPYFACE"(01H และ 02H) และรูปหัวใจ (heart) ต่อมา DIAMOND และ CLUB(01H,04H,05H) 07H เท่ากับเสียง 06H,08H และ 0DH เป็นสัญญาณควบคุม อันที่จริง 0DHเท่ากับ carriage return (enter) ในการเริ่มต้นบรรทัดต่อไป 0EH คือโน้ตดนตรี

ท่านสามารถเปลี่ยนโปรแกรมโดยไม่ต้องแสดงอักขระควบคุม โดยใช้คำสั่งข้ามตัวอักขระทั้งหมดระหว่าง 08H และ 0DH ดังนี้

```
            CMP     CHAR,08H    ; LOWER THAN 08H
            JB      D30         ; YES ACCEPT
            CMP     CHAR,0DH    ; LOWER//EQUAL 0DH
            JBE     D40         ; YES,BYPASS
D30:
            MOV     AH,40H       ; DISOLAY<08H
            ---                  ; AND >0DH
            INT     21H
D40:        INC     CHAR
```

การใช้ DOS รับข้อมูลทางคีย์บอร์ด

การบริการของDOSในการรับข้อมูลจากคีย์บอร์ด พื้นที่ในการรับข้อมูลจะต้องการ parameter list ที่ใช้กำหนดฟิลด์ที่จะให้คำสั่ง INT ประมวลผล ขั้นแรก คำสั่งอินเตอร์รัพท์ต้องการทราบความยาวสูงสุดของข้อมูลเข้า วัตถุประสงค์ไม่ต้องการให้ผู้ใช้ป้อนข้อมูลมากเกินไป การทำงานของเสียงที่ลำโพงดังขึ้น มันจะไม่ยอมรับข้อมูลเพิ่ม ขั้นสอง การทำงานจะตรวจสอบจำนวนไบท์ที่ป้อนเข้ามา การกำหนด parameter list สำหรับข้อมูลเข้า ลาเบล(label)เป็นคำสั่งเทียม ซึ่งเป็นตัวกำหนด attribute ข้อมูลเป็นไบท์ ซึ่งข้อมูลจะอยู่ในรูปของไบท์ ไบท์แรกจะเป็นตัวกำหนดความยาวสูงสุดของจำนวนตัวอักขระ ค่าต่ำเป็น 0 ตัว ค่าสูงสุดฟิลด์ขนาด 1 ไบท์ จะเก็บค่าสูงสุดคือ FFH หรือ 255 ไบท์ที่ 2 สำหรับการทำงานในการเก็บข้อมูล ของตัวอักขระที่ป้อนเข้ามา ไบท์ที่ 3 จะเป็นฟิลด์เริ่มต้นในการเก็บอักขระที่พิมพ์เข้าไป

```
NAMEPAR   LABEL  BYTE          ; Start of parameter list
MAXLEN    DB     20            ; Maximum number of input
                               ; characters
ACTLEN    DB     ?             ; Actual number of input
                               ; characters
NAMEFLD   DB     20DUP(' ')    ; Characters entered from
                               ; keyboard
```

ในส่วนของ parameter list คำสั่งเทียม LABEL จะเป็นตัวบอกแอสเซมเบลอร์กำหนดขอบเขตของข้อมูลเป็นชนิดไบท์ โดยใช้ชื่อของตำแหน่ง NAMEPAR ซึ่ง NAMEPAR , MAXLEN จะอ้างถึงตำแหน่งของหน่วยความจำเดียวกัน

การบริการ 0AH ของDOS เป็นการขอร้องการรับข้อมูล ซึ่งค่า 0AH จะอยู่ใน AH และโหลดค่าแอดเดรสของ parameter list NAMEPAR ลงในDXและใช้ INT 21H

```
MOV     AH,0AH          ; Request input service
LEA     DX,NAMEPAR      ; Load address of parameter list
INT     21H             ; DOS interrupt
```

การทำงานของ INT จะรอผู้ใช้ในการป้อนข้อมูลและตรวจสอบตัวอักขระ หรือข้อมูลเกินค่าสูงสุดใน parameter list หรือไม่ การทำงานของข้อมูลจะแสดงให้เห็นที่จอภาพ และcursor

จะเลื่อนไปล่วงหน้า ผู้ใช้ก็สามารถกดคีย์ ENTER เป็นการสิ้นสุดในการป้อนข้อมูล การทำงานของการเคลื่อนย้ายอักขระเหล่านี้ (0DH) ไปยังฟิลด์ข้อมูลเข้า (NAMEFLD) แต่ไม่รวมถึงความยาวจริง ถ้าท่านคีย์ชื่อเช่น BROWN (ENTER) parameter list จะเป็นดังนี้

ASCII :	20	5	B	R	O	W	N	#				
HEX :	14	05	42	52	4F	57	4E	0D	20	20	20	20

การทำงานของตัวกำหนดความยาวข้อมูลเข้า 05H ในไบท์ที่ 2 ของ parameterlist ชื่อ ACTLEN ที่อยู่ในตัวอย่าง ตัวอักขระ ENTER คือ NAMEFLD+5 ส่วนเครื่องหมาย # จะเป็นตัวชี้ตัวอักขระเหล่านี้ เพราะค่า 0DH จะไม่มีการพิมพ์ แต่ความยาวสูงสุดกำหนดไว้ 20 รวมทั้งค่า 0DH ฉะนั้นความยาวของชื่อจริง จะมีเพียง 19 ตัวเท่านั้น

การทำงานจะไม่ยอมรับมากกว่าจำนวนค่าสูงสุดของตัวอักขระ ถ้าผู้ใช้คีย์มากกว่า 20 ตัว โดยปราศจากการกด enter จะมีเสียงออกที่ลำโพง มันจะรับเพียงคีย์ของ ENTER เท่านั้น

การรับและการแสดงชื่อ ACCEPTING AND DISPLAYING NAMES

โปรแกรม EXE ในรูป8-2เป็นการต้องการป้อนชื่อ และแสดงผลที่จุดกึ่งกลางจอภาพ และให้เสียงออกที่ลำโพง โปรแกรมจะยอมรับข้อมูลและแสดงผล จนกระทั่งผู้ใช้กด ENTER เพื่อหลุดจาก prompt ถ้าผู้ใช้ใส่ชื่อ TED SMITH โปรแกรมจะทำงานดังนี้

1. หารความยาว 09 ด้วย 2 : 9/2 = 4 เศษตัดทิ้ง
2. ลบออกจากค่า 40 : 40-4 = 36

ในโปรแกรม F10CENT คำสั่ง SHR จะเลื่อนความยาว 09 ไป 1 บิตไปทางขวาบิต 0000 1001 จะกลายเป็น 0000 0100

ผลของการหารความยาวด้วย 2 คำสั่ง NEG จะเปลี่ยนเครื่องหมายจาก +4 เป็น -4และบวกเข้ากับค่า 40 ซึ่งเป็นตำแหน่งแรกของคอลัมน์ 36 ในรีจิสเตอรี DL ซึ่งเป็นการเซ็ท cursor ที่บรรทัด 12 คอลัมน์ 36 และปรากฏชื่อบนจอภาพดังนี้

```
          page    60,132
TITLE     CTRNAME(EXE)  Accept names & center on screen
; ----------------------------------------------------
          .MODEL  SMALL
          .STACK  64
; ----------------------------------------------------
          .DATA
NAMEPAR   LABEL   BYTE              ; Name parameter list
MAXNLEN   DB      20                ; maximum length of name
NAMELEN   DB      ?                 ; no. of characters entered
NAMEFLD   DB      20 DUP(' '),'$'   ; name and delimiter
                                    ; for displaying
PROMPT    DB      'Name? ','$'
; ----------------------------------------------------
          .CODE
BEGIN     PROC    FAR
          MOV     AX,@data          ; Initialize segment
          MOV     DS,AX             ;    registers
          MOV     ES,AX
          CALL    Q10CLR            ; Clear screen
A20LOOP:
          MOV     DX,0000           ; Set cursor to 00,00
          CALL    Q20CURS
          CALL    B10PRMP           ; Display prompt
          CALL    D10INPT           ; Provide for input of
                                    ; name
          CALL    Q10CLR            ; Clear screen
          CMP     NAMELEN,00        ; Name entered?
          JE      A30               ;    no-exit
          CALL    E10CODE           ; Set bell & '$'
          CALL    F10CENT           ; Center & display name
          JMP     A20LOOP
```

```
A30:
            MOV     AH,4CH          ; Return to DOS
            INT     21H
BEGIN       ENDP
;---------------- Display prompt ----------------------------
B10PRMP     PROC    NEAR
            MOV     AH,09           ; Request display
            LEA     DX,PROMPT
            INT     21H
            RET
B10PRMP     ENDP
;---------------- Accept input of name ----------------------
D10INPT     PROC    NEAR
            MOV     AH,0AH          ; Request input
            LEA     DX,NAMEPAR
            INT     21H
            RET
D10INPT     ENDP
;---------------- Set bell and '$' delimiter ----------------
E10CODE     PROC    NEAR
            MOV     BH,00           ; Replace enter char (0D)
            MOV     BL,NAMELEN      ; with bell (07)
            MOV     NAMEFLD[BX],07
            MOV     NAMEFLD[BX+1],'$' ; Set display delimiter
            RET
E10CODE     ENDP
;---------------- Center and display name -------------------
F10CENT     PROC    NEAR
            MOV     DL,NAMELEN      ; Locate enter column:
            SHR     DL,1            ;  divide length by 2
            NEG     DL              ;  reverse sign,
            ADD     DL,40           ;  add 40
```

```
                MOV     DH,12           ; Center row
                CALL    Q20CURS         ; Set cursor
                MOV     AH,09
                LEA     DX,NAMEFLD      ; Display name
                INT     21H
                RET
F10CENT         ENDP
;---------- Clear screen ------------------------------
Q10CLR          PROC    NEAR
                MOV     AH,0600H        ; Request scroll screen
                MOV     BH,30           ; Color (07 for BW)
                MOV     CX,0000         ; From 00,00
                MOV     DX,184FH        ; To 24,79
                INT     10H             ; Call BIOS
                RET
Q10CLR          ENDP
;------------------- Set cursor row/column ----------------------
Q20CURS         PROC    NEAR            ; DX set on entry
                MOV     AH,02           ; Request set cursor
                MOV     BH,00           ; Page 0
                INT     10H             ; Call BIOS
                RET
Q20CURS         ENDP
                END     BEGIN
```

รูปที่ 8-2 Accepting and displaying names

```
ROW 12 :   Ted Smith
           |   |
COLUMN :   36  40
```

สังเกตจากคำสั่งในโปรแกรม E10CODE นั้น จะมีรหัส 07H(bell) ในพื้นที่ของการป้อนข้อมูลชื่อเข้าดังนี้

```
MOV     BH,00          ; REPLACE ENTER CHARACTER(0DH)
MOV     BL,NAMELEN     ; WITH BELL(07H)
MOV     NAMEFLD[BX],07
```

คำสั่ง MOV 2 คำสั่งแรก เป็นการเซ็ทความยาวของ BX คำสั่ง MOV คำสั่งที่ 3 จะอ้างถึงการทำงานของ index ในวงเล็บสี่เหลี่ยม ซึ่งหมายความว่า BX คือ index รีจิสเตอร์ ซึ่งเป็นตัวกำหนดแอดเดรส คำสั่ง MOV จะมีความยาวใน BX เท่ากับค่าแอดเดรส NAMEFLD และเคลื่อนย้าย 07H ไปยังแอดเดรสนี้ เช่นความยาวของ 05H คำสั่งจะใส่ 07H ที่ NAMEFLD+05 ต่อท้ายชื่อ คำสั่งสุดท้ายในส่วนของ E10CODE จะป้อน $ ตามหลัง 07H ดังนั้นเมื่อ F10CENT แสดงชื่อ จะมีเสียงจากลำโพง

REPLAYING WITH ONLY THE ENTER CHARACTER

ถ้าท่านคีย์ ENTER การทำงานรับข้อมูล และความยาวของข้อมูลจะมีค่าเป็น 0 จะได้ PARAMETER LIST ดังนี้

PARAMETER LIST (HEX) : |14|00|0D|...

จุดสุดท้ายของการสิ้นสุดการรับข้อมูลใน PROMPT สำหรับใส่ชื่อ ผู้ใช้สามารถกด ENTER ถ้ามีความยาวเป็น 0 เป็นการสิ้นสุดโปรแกรม

CLEARING THE ENTER CHARACTER

ท่านสามารถป้อนข้อมูลเข้าสำหรับวัตถุประสงค์ต่างๆ เช่นการพิมพ์รายงาน การเก็บข้อมูลในตาราง หรือ การเขียนข้อมูลลงดิสก์ สำหรับวัตถุประสงค์เหล่านี้ ท่านสามารถแทนด้วยตัวอักขระ ENTER(0DH) บางครั้งในส่วนของ NAMEFLD จะมี blank(20H) จะเป็นส่วนหนึ่งของความยาวในฟิลด์ NAMELEN จะมีข้อมูล 05 ตำแหน่งนี้คือ NAMEFLD+5 ท่านสามารถเคลื่อนความยาวนี้ไปเก็บใน BX สำหรับกำหนด index ของแอดเดรส NAMEFLD ดังนี้

```
        MOV     BH,00               ; SET BX
        MOV     BL,NAMELEN          ; TO 00 05
        MOV     NAMEFLD[BX],20H     ; CLEAR ENTER CHARACTER
```

คำสั่ง MOV 2 คำสั่งแรก จะเซ็ทค่า BX=5 เป็นความยาว คำสั่ง MOV คำสั่งที่ 3 จะเคลื่อน blank (20H) ไปยังแอดเดรสที่กำหนดในรีเปอรานด์แรก แอดเดรสของ NAMEFLD บวกกับค่าใน BX จะได้ค่า NAMEFLD+5

CLEARING THE INPUT AREA

INPUT	NAMEPAR(hex)												
1. BROWN	14	05	42	52	4F	57	4E	0D	20	20	20	...	20
2. HAMILTON	14	08	48	41	4D	49	4C	54	4F	4E	0D	...	20
3. ADAMS	14	05	41	44	41	4D	53	0D	45	5A	0D	...	20

ตัวอักขระ ENTER จะใช้แทนการสิ้นสุดข้อมูล พิจารณาจากข้อมูลเข้าต่อไปนี้ ในชื่อ HAMILTON ใช้แทนชื่อที่สั้นกว่าในชื่อ BROWN แต่ในส่วนชื่อของ ADAMS สั้นกว่าชื่อ HAMILTON ฉะนั้นชื่อของ ADAMS จะใช้แทนเฉพาะ HAMIL และป้อนอักขระ ENTER ใช้แทนตัวอักษร T ฉะนั้นจะเหลือตัวอักษร ON ตามหลังชื่อ ADAMS ท่านจะต้องเคลียร์ NAMEFLD ก่อนรับชื่อใหม่ดังนี้

```
        MOV     CX,20               ; Initialize for 20 loops
        MOV     SI,0000             ; Start position for name
B30 :
        MOV     NAMEFLD[SI],20H     ; One blank to name
        INC     SI                  ; Increment for next
                                    ; character
        LOOP    B30                 ; 20 times
```

แทนที่รีจิสเตอร์ SI ท่านควรจะใช้DIหรือBX เป็นวิธีการที่ดีกว่าในการเคลื่อนย้ายเวิร์ดของ blank 2 ตัวที่ใช้ใน LOOP 10 รอบ อย่างไรก็ตาม NAMEFLD จะกำหนดด้วย DB ท่านจะเขียนความยาวเวิร์ดทับที่เดิมโดยใช้ WORD และ PTR(pointer) ดังต่อไปนี้

```
                MOV     CX,10               ; Initialize for ten loops
                LEA     SI,NAMEFLD          ; Initialize start of name
B30 :
                MOV     WORD PTR[SI],2020H  ; Two blanks to name
                INC     SI                  ; Increment two position
                INC     SI                  ;  in name
                LOOP    B30                 ; Loop ten times
```

การทำงานการแปลคำสั่ง MOV ที่ B30 เป็นการเคลื่อนย้ายเวิร์ด blank ตามแอดเดรส SI กำหนด ตัวอย่างนี้ใช้คำสั่ง LEA แทนได้ และใช้วิธีการที่แตกต่างของคำสั่ง MOV ที่ B30 เพราะท่านไม่สามารถใช้รหัสคำสั่งได้

```
                MOV     WORD PTR[NAMEFLD],2020H     ; INVALID
```

การทำงานของจอภาพและคีบอร์ด SCREEN AND KEYBOARD OPERATIONS : EXTENDED DOS

เราตรวจสอบวิธีการทำงานของจอภาพและคีย์บอร์ด โดยใช้ DOS 2.0 ซึ่งใช้มากใน UNIX และ OS/2 วิธีการที่เพิ่มขึ้นรวมถึง file handle ซึ่งท่านสามารถโหลดลงใน BX เมื่อต้องการทำงานเกี่ยวกับ I/O ตามกฎเกณฑ์มาตรฐานของการใช้ ในแฟ้มข้อมูลที่กำหนดไว้

```
00      INPUT,NORMALLY KEYBOARD (CON)
01      OUTPUT,NORMALLY DISPLAY (CON)
02      ERROR OUTPUT,DISPLAY (CON)
03      AUXILIARY DEVICE (AUX)
04      PRINTER (LPT1 or PRN)
```

การใช้อินเตอร์รัพท์ของ DOS INT 21H และโหลดการบริการในรีจิสเตอร์ AH ด้วยค่า 3FH เป็นของ input และค่า 40H สำหรับค่า output เซ็ทค่า CX ด้วยจำนวนของไบท์ที่ต้องการอ่านหรือแสดงผล และโหลดค่า DX ด้วยแอดเดรสของ input หรือ output area

การทำงานจะต้องมีการเคลียร์แฟลกตัวทด และใส่ตัวอักขระที่ AX ที่จะป้อนเข้า หรือแสดงผล ซึ่งจะสำเร็จผล ถ้าไม่ประสบผลสำเร็จ จะเป็นการเซ็ทแฟลกตัวทด และ ใส่รหัสผิดพลาด

(error code) ใน AX ข้อมูลใน AX ขึ้นอยู่กับความยาวหรือรหัสผิดพลาดเงื่อนไขของรหัสผิดพลาด โดยการตรวจสอบแฟลกตัวทด

EXTENDED DOS FOR SCREEN DISPLAY

การบริการ DOS 40H เป็นการแสดงผล โหลดค่า 01 ใน BX จำนวนตัวอักขระที่จะแสดงผลใน CX และแอดเดรสของพื้นที่แสดงผลใน DX การทำงานจะตอบสนองเหมือนกับการบริการ 09 ถึง 07H(beep),08H(backspace),0AH(line feed) และ 0DH (carriage return) ดังตัวอย่างคำสั่งต่อไปนี้

```
DISAREA   DB   16 DUP('PC Users Society')  ; Display area
          MOV      AH,40H          ; Request display
          MOV      BX,01           ; File handle for output
          MOV      CX,16           ; Maximum 16 characters
          LEA      DX,DISAREA      ; Display area
          INT      21H             ; Call DOS
```

คำสั่ง LEA เป็นการโหลดแอดเดรส DISAREA ใน DX เพื่อเป็นการอนุญาตให้ DOS หาตำแหน่งของข่าวสารที่นำมาแสดง ถ้าทำงานสำเร็จจะเคลียร์แฟลกตัวทด และเซ็ทค่า AX เป็นจำนวนตัวอักขระที่แสดงแล้ว ถ้าการทำงานไม่สำเร็จก็จะเซ็ทค่าแฟลก และใส่รหัสผิดพลาดใน AX และค่า AX เป็นความยาวของข้อมูลหรือรหัสที่ผิดพลาด ถ้าตรวจการตรวจสอบข้อผิดพลาดโดยการตรวจสอบแฟลกดังนี้

```
          JC       ERROR-ROUTINE    ; Test for display error
```

การแสดงผลจอภาพ : DISPLAY ON THE SCREEN

เราสามารถใช้โปรแกรม DEBUG ตรวจสอบผลที่เกิดขึ้นภายในของคำสั่งอินเตอร์รัพท์การโหลด DEBUG จะมี prompt เกิดขึ้น ให้พิมพ์ค่า A 100 เป็นจุดเริ่มต้นของคำสั่งที่ แอดเดรส 100 อย่าลืมว่าโปรแกรม DEBUG สมมุติค่าจำนวนทั้งหมดเป็นเลขฐานสิบหก

```
100     MOV     AH,40
102     MOV     BX,01
105     MOV     CX,XX (Insert length of your name)
108     MOV     DX,10E
10B     INT     21
10D     RET
10E     DB      'Your name'
```

โปรแกรมจะเซ็ทค่า AH เพื่อต้องการแสดงผล และเซ็ทค่า 10EH ในรีจิสเตอร์ DX ซึ่งเป็นตำแหน่งของDBที่มีข้อมูล 'your name' ที่จุดสุดท้ายของโปรแกรม เมื่อท่านพิมพ์คำสั่งต่างๆเรียบร้อยแล้วให้กด ENTER อีกครั้งแล้วพิมพ์คำสั่ง U(U 100,10D) เพื่อทำการ unassemble โปรแกรม และใช้คำสั่ง R ตามด้วยคำสั่ง T เพื่อทำการ execute ทีละคำสั่ง เมื่อ debug execute มาถึงคำสั่ง INT 21H จะ execute ผ่าน BIOS ดังนั้นเมื่ออ่านถึง INT จะใช้คำสั่ง P จะexecute โดยตรงผ่าน BIOS ไปยังคำสั่งต่อไป ชื่อของท่านก็จะแสดงผลที่จอภาพ ใช้คำสั่ง Q กลับสู่ DOS

EXTENDED DOS FOR KEYBOARD INPUT

DOS การบริการ 3FH เป็นการทำงานของการรับข้อมูลของคีย์บอร์ด ค่าของ BX จะเป็นค่า file handle 00 , CX เก็บค่าสูงสุดของตัวอักขระที่รับได้ และ DX เป็นแอดเดรสของพื้นที่ที่รับข้อมูลเข้า ตัวอย่างคำสั่งต่อไปนี้ เป็นการใช้ DOS การบริการ 3FH

```
INAREA  DB      20 DUP(' ')  ; Input area
        ...
        MOV     AH,3FH       ; Request input
        MOV     BX,00        ; File handle for keyboard
        MOV     CX,20        ; Maximum 20 characters
        LEA     DX,INAREA    ; Input area
        INT     21H          ; Call DOS
```

คำสั่ง LEA เป็นการโหลดค่า offset address ของ INAREA ไว้ใน DX การทำงานของคำสั่ง INT จะคอยจนกว่าผู้ใช้ป้อนตัวอักขระ แต่ไม่มีการตรวจสอบจำนวนตัวอักขระที่เกินค่าสูงสุด

ใน CX การกดคีย์ ENTER (0DH) ซึ่งเป็นข้อมูลสุดท้าย สำหรับตัวอย่าง การป้อนตัวอักขระ "PC USERS GROUP" ไว้ใน INAREA

PC USERS GROUP ¦0DH¦0AH¦

การพิมพ์ตัวอักขระและตามด้วย ENTER(0DH) ซึ่งท่านพิมพ์เรียบร้อยแล้วและ linefeed(0AH) ซึ่งท่านไม่ได้พิมพ์ ซึ่งจำนวนค่าสูงสุดและความยาวของการพิมพ์ข้อมูลเข้าจะต้องมี 2 ตัวนี้คือ 0DH และ0AH ถ้าท่านพิมพ์ตัวอักขระน้อยกว่าค่าสูงสุด ตำแหน่งในหน่วยความจำจะต้องตามด้วยตัวอักขระ ENTER ตามหลังข้อมูลที่พิมพ์

การทำงานที่สมบูรณ์จะเป็นการเคลียร์แฟลกตัวทด และ เซ็ทค่า AX กับจำนวนของตัวอักขระที่รับ ในตัวอย่างความยาวของตัวอักษรคือ 14 บวกอีก 2 คือ 0DH,0AH ซึ่งเป็นการสิ้นสุดในการรับข้อมูลเข้า

ถ้าการทำงานไม่สมบูรณ์จะเกิดการผิดพลาด จะเป็นการเซ็ทค่าแฟลก และใส่รหัสผิดพลาดใน AX และ AX จะมีข้อมูลของความยาวตัวอักขระหรือรหัสผิดพลาด การตรวจสอบข้อผิดพลาดตามเงื่อนไข ได้จากการตรวจสอบแฟลก

ถ้าท่านคีย์ซื่อมากกว่าค่าสูงสุดใน CX การทำงานก็จะรับตัวอักขระทั้งหมด พิจารณาจากสถานะ ซึ่งข้อมูลใน CX=8 ถ้าผู้ใช้ป้อนตัวอักขระ "PC EXCHANGE" การทำงานจะเซ็ทตัวอักขระ 8 ตัวแรกใน INPUTAREA "PC EXCHA" ซึ่งไม่มี 0DH และ 0AH ตามหลัง และเซ็ทค่าใน AX=8 ซึ่งเป็นความยาวของตัวอักขระ การทำงานของ INT ต่อไปจะไม่ยอมรับซื่อโดยตรงจากคีย์บอร์ด เพราะมันยังคงมีสตริงในตัวบัฟเฟอร์ มันจะขับ "NGE" ตามด้วย 0DHและ0AH ไปที่ INPUTAREA และเซ็ทค่า AX=5 การทำงานทั้งสองคือ "NORMAL" และเคลียร์แฟลกตัวทด

FIRST INT :	PC EXCHA	AX=08
SECOND INT :	NGE,0D,0A	AX=05

การรับข้อมูล : ENTERING DATA

จากแบบฝึกหัดท่านสามารถแสดงผลของการป้อนข้อมูลจะใช้โปรแกรม DEBUG โปรแกรมรับข้อมูลได้ 12 ตัวอักขระ รวมทั้ง 0AH,0DH หลังจากโหลดโปรแกรม DEBUG แล้ว ใช้คำสั่ง A 100 เริ่มต้นการป้อนข้อมูลที่แอดเดรส 100

```
100     MOV     AH,3FH
102     MOV     BX,00
105     MOV     CX,0C
108     MOV     DX,10F
10B     INT     21
10D     JMP     100
10F     DB      ' '
```

โปรแกรมจะเซ็ทค่ารีจิสเตอร์ AH และ BX ตามการทำงานของ keyboard inputโดย CX เก็บค่าความยาวสูงสุดและเซ็ทค่า DX=10FH ซึ่งเป็นตำแหน่งของ DB อยู่จุดสุดท้ายของโปรแกรม การป้อนอักขระเริ่มต้นที่แอดเดรส 10FH

เมื่อท่านคีย์ในคำสั่ง ให้กด ENTER อีกครั้ง ใช้คำสั่ง U (U100,108) ในการ unassemble ของโปรแกรม ใช้คำสั่ง R และใช้คำสั่ง T ซ้ำๆกัน ในการ execute ของคำสั่ง MOV 4 คำสั่ง เมื่อถึงตำแหน่ง 10B ใช้คำสั่ง P ในการ execute ผ่าน INTERRUPT การอินเตอร์รัพท์จะหยุดและรอคอยการป้อนข้อมูลเข้า และตามด้วย ENTER ตรวจสอบข้อมูลในรีจิสเตอร์ AX แฟลกตัวทด ใช้คำสั่ง D 10F ที่แสดงการรับตัวอักขระในหน่วยความจำ คำสั่ง Q คือ QUIT

การใช้ 0DH,0AH และ 09H แสดงผล

อีกกรณีหนึ่งที่จะแสดงผลการ execute ได้เร็วกว่าโดยการใช้ 0DH,0AH และ 09H ซึ่งเป็นรหัส ASCII

	ASCII	HEX
CARRIAGE RETURN	13	0DH
LINE FEED	10	0AH
TAB	09	09H

การใช้ตัวอักขระเหล่านี้ ขณะที่ท่านแสดงผลและรับข้อมูลเข้า สำหรับcursorที่รอรับโดย อัตโนมัติ ในการเริ่มต้นของบรรทัดต่อไป

```
MESSAGE   DB   09,'PC USRES GROUP ANNUAL REPORT',13,10
          ...
          MOV  AH,40H          ; REQUEST DISPLAY
          MOV  BX,01           ; HANDLE
          MOV  CX,31           ; LENGTH
          LEA  DX,MESSAGE      ; MESSAGE
          INT  21H
```

การใช้ EQU ในการกำหนดการทำงานของโปรแกรม

```
          CR   EQU  13
          LF   EQU  10
          TAB  EQU  09
MESSAGE   DB   TAB,'PC USERS GROUP ANNUAL REPORT',CR,LF
```

บทสรุป

- หน่วยความจำสนับสนุนจอโมโนโครมมีขนาด 4 K ตัวอักขระ 2 K อาร์ทิบิวต์ 2 K
- หน่วยความจำสนับสนุนจอสีมีขนาด 16 K สามารถทำงานได้จอสีและจอโมโนโครม ท่านสามารถประมวลผลได้ทั้ง TEXT MODE สำหรับตัวอักขระธรรมดา แสดงผลใน GRAPHICS MODE
- คำสั่ง INT 10H ใช้ทำงานร่วมกับ BIOS ในการแสดงผล ตัวที่ใช้มากคือ Service 2 และ Service 6
- คำสั่ง INT 21H ใช้ทำงานร่วมกับ DOS ในการติดต่ออินพุตเอ้าพุต

แบบฝึกหัด

8-1. เลขฐานสิบหกที่มุมล่างขวามือสุดคือเลขอะไรของจอภาพ 80x25

8-2. จงเขียนคำสั่งในการเซ็ตเคอร์เซอร์ไปที่ตำแหน่งบรรทัดที่ 12 คอลัมน์ที่ 25

8-3. จงเขียนคำสั่งในการเคลียจอภาพเริ่มต้นบรรทัดที่ 12 คอลัมน์ 0 ถึงบรรทัดที่ 25 คอมลัมน์ 79

8-4. จงเขียนรายการข้อมูลและคำสั่งในการแสดงข่าวสารดั่งนี้ "what is the date?:"

8-5. จงเขียนคำสั่งในการรับข้อมูลจากคีย์บอร์ดตามรูปแบบดังต่อไปนี้

"WHAT IS THE DATE(MM/DD/YY)?:"

โดยใช้การบรีการของดอส

8-6. จงเขียนโปรแกรมในการรับข้อมูลจากคีย์บอร์ด และแสดงผลที่จอภาพกับเครื่องพิมพ์

8-7. จงเขียนโปรแกรมในการรับข้อมูลของบุคลากร โดยมีฟิลด์ต่างๆดังต่อไปนี้

NAME = ...

ADDRESS = ...

CITY = ...

TEL = ...

INPUT AGAIN (Y/N)

8-8. จงเขียนโปรแกรมในการรับตัวอักษรเลขฐานสิบหก A - F และแสดงผลในบรรทัดต่อไปใน รูปเลขฐานสิบ

ENTER A HEX DIGIT : C

IN DECIMAL IT IS : 12

8-9 จงเขียนโปรแกรมในการรับตัวอักขระ 3 ตัว แล้วพิมพ์ผลลัพธ์บรรทัดละตัวดังนี้

ENTER THREE CHARACTER : JFK

J

F

K