TẬP LỆNH CPU INTEL 8086/8088

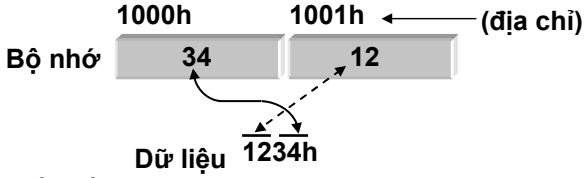
1. <u>Dạng lệnh</u> :

- Một lệnh của vi xử lý 86 có dạng tổng quát như sau :

  <Mã gợi nhớ> <Toán hạng đích>,<Toán hạng nguồn>

- **Mã gợi nhớ** giúp cho người sử dụng biết hoạt động của lệnh. Mã gợi nhớ thường là các chữ tiếng anh viết tắt như : MOV là lệnh chuyển, ADD là lệnh cộng, AND là lệnh và luận lý, JMP là lệnh nhảy . . .
- **Toán hạng đích** giữ kết quả (nếu có yêu cầu) sau khi thi hành lệnh. Toán hạng đích có thể là thanh ghi hay bộ nhớ.
- **Toán hạng nguồn** có thể là thanh ghi, bộ nhớ hay một số tức thời.
- Toán hạng thanh ghi là các thanh ghi của vi xử lý 86 gồm các thanh ghi tổng quát (8 bit lẫn 16 bit) và các thanh ghi đoạn đã biết.
- Toán hạng số tức thời có thể là số trong các hệ đếm khác nhau và được viết theo qui định như sau :
  - . Số hệ 2 : ××××××××B (× là 1 bit nhị phân).
    <u>*Ví dụ*</u> : 01101101B, 11111111B
  - . Số hệ 10 : ××××× , hay ×××××D (× là một số thuộc hệ 10).
    <u>*Ví dụ*</u> : 65535, 1000
  - . Số hệ 16 : ××××H và bắt đầu bằng số (là một số thuộc hệ 16).
    <u>*Ví dụ*</u> : 1A59H, 0E05BH
- Toán hạng bộ nhớ dùng trong tập lệnh vi xử lý 86 sử dụng phương pháp định địa chỉ tổng hợp được gọi là <u>*địa chỉ hiệu dụng*</u>.
- Địa chỉ hiệu dụng là tổ hợp của 3 nhóm sau được đặt trong dấu ngoặc vuông [ ]:
  - . Nhóm thanh ghi chỉ số : SI, DI
  - . Nhóm thanh ghi nền : BX, BP
  - . Địa chỉ trực tiếp : số 16 bit
- Các thanh ghi trong cùng một nhóm không được xuất hiện trong cùng một địa chỉ hiệu dụng.
- Ví dụ :
  - . Địa chỉ hiệu dụng hợp lệ :
    [1000h], [SI], [DI], [BX], [BP]
    [SI+BX], [SI+BP], [DI+BX], [DI+BP], [SI+1000h], [DI+100h], [BX+1], [BP+1]
    [SI][BX][1000h], [SI+ BP+1000h], [DI+BX][1000h], [DI+1000h][BP]
  - . Địa chỉ hiệu dụng <u>*không hợp lệ*</u> :
    [70000], [AX], [SI+DI+1000h], [BX][BP]
- Địa chỉ hiệu dụng chính là thành phần offset của địa chỉ luận lý bộ nhớ.
- Segment của địa chỉ hiệu dụng được mặc định như sau :
  . Nếu không sử dụng BP trong địa chỉ hiệu dụng thì mặc định theo thanh ghi DS.
  . Nếu có BP trong địa chỉ hiệu dụng thì mặc định theo thanh ghi SS.
- Các hoạt động thực hiện trên bộ nhớ thông qua địa chỉ hiệu dụng chia ra làm 2 trường hợp : hoạt động 8 bit và hoạt động 16 bit.
- Hoạt động bộ nhớ 8 bit làm việc trên 1 byte bộ nhớ ngay vị trí chỉ ra bởi địa chỉ hiệu dụng.

- Hoạt động bộ nhớ 16 bit sẽ làm việc trên 2 byte bộ nhớ có địa chỉ kế tiếp nhau và nội dung của chúng được ghép lại thành dữ liệu 16 bit theo qui tắc "***byte cao địa chỉ cao, byte thấp địa chỉ thấp***" như trong hình sau :



- Để thuận tiện trong vấn đề giải thích lệnh, ta qui ước thêm cách diễn tả sau :
  . Dữ liệu 8 bit của bộ nhớ : **[địa chỉ]**
  . Dữ liệu 16 bit của bộ nhớ : **[địa chỉ +1,địa chỉ]**
- Để xác định rõ hoạt động của bộ nhớ, ta phải dùng thêm toán tử PTR như sau :
  . Hoạt động 8 bit : **BYTE PTR** [1000h] là tham khảo 1 byte bộ nhớ có địa chỉ 1000h
  . Hoạt động 16 bit : **WORD PTR** [1000h] là tham khảo đến 2 byte bộ nhớ liên tiếp 1000h và 1001h
- Các chữ viết tắt dùng trong các nhóm lệnh :

reg	: thanh ghi tổng quát.
reg16	: thanh ghi 16 bit.
segreg	: thanh ghi đoạn.
accum	: thanh ghi bộ tích lũy AX hoặc AL.
mem	: bộ nhớ (địa chỉ hiệu dụng).
mem16	: bộ nhớ 2 byte liên tiếp (địa chỉ hiệu dụng).
mem32	: bộ nhớ 4 byte liên tiếp (địa chỉ hiệu dụng).
immed	: số tức thời.
immed8	: số tức thời 8 bit.
shortlabel	: nhãn ngắn (-128 byte +127 byte).
nearlabel	: nhãn trong đoạn (2 byte offset).
farlabel	: nhãn ngoài đoạn (4 byte : 2 byte segment và 2 byte offset).

2. <u>Nhóm lệnh chuyển dữ liệu</u> :

2.1 <u>Lệnh MOV</u> :

- Dạng lệnh :

```
MOV  reg,reg          MOV  reg,immed
MOV  mem,reg          MOV  mem,immed
MOV  reg,mem          MOV  mem16,segreg
MOV  reg16,segreg     MOV  segreg,mem16
MOV  segreg,reg16
```

- Giải thích : ***thđ ← thn***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Chép toán hạng nguồn vào toán hạng đích.
- *<u>Ví dụ</u>* :

```
MOV  AX,CX          ; AX ← CX
MOV  DL,BH          ; DL ← BH
MOV  [SI+1000h],BP  ; [SI+1001h, SI+1000h] ← BP
MOV  DX,[1000h]     ; DX ← [1001h,1000h]
```

```
MOV  DX,DS              ; DX ← DS
MOV  ES,BX              ; ES ← BX
MOV  DI,12h             ; DI ← 12h
MOV  AL,12h             ; AL ← 12h
MOV  BYTE PTR [1000h],12h     ; [1000h] ← 12h
MOV  WORD PTR [2000h],1200h ; [2001h,2000h] ← 1200h
MOV  [BX],DS            ; [BX+1,BX] ← DS
MOV  SS,[2000h]         ; SS ← [2001h,2000h]
```

2.2 Lệnh PUSH :

- Dạng lệnh : PUSH *reg16* PUSH *segreg*
  PUSH *mem16*
- Giải thích : ***SP ← SP-2***
  [SS:SP+1,SS:SP] ← thn
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Đẩy toán hạng nguồn 16 bit vào chồng (địa chỉ đỉnh chồng là SS:SP).
- *Ví dụ* :

```
PUSH  DI          ; [SS:SP+1,SS:SP] ← DI
PUSH  CS          ; [SS:SP+1,SS:SP] ← CS
PUSH  [SI]        ; [SS:SP+1,SS:SP] ← [SI+1,SI]
```

2.3 Lệnh POP :

- Dạng lệnh : POP *reg16* POP *segreg*
  POP *mem16*
- Giải thích : ***thđ ← [SS:SP+1,SS:SP]***
  SP ← SP+2
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Lấy dữ liệu từ đỉnh chồng vào toán hạng đích.
- *Ví dụ* :

```
POP  AX           ; AX ← [SS:SP+1,SS:SP]
POP  ES           ; ES ← [SS:SP+1,SS:SP]
POP  [BX+1]       ; [BX+2,BX+1] ← [SS:SP+1,SS:SP]
```

2.4 Lệnh XCHG :

- Dạng lệnh : XCHG *reg,reg* XCHG *mem,reg*
  XCHG *accum,reg16* XCHG *reg,mem*
- Giải thích : ***thđ ↔ thn***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Trao đổi nội dung hai toán hạng cho nhau.
- *Ví dụ* :

```
XCHG  AX,CX       ; AX ↔ CX
XCHG  AH,AL       ; AH ↔ AL
XCHG  [1000h],DX  ; [1001h,1000h] ↔ DX
```

2.5 Lệnh IN :

- Dạng lệnh : IN *accum,immed8*
  IN *accum,DX*
- Giải thích : ***btl ← [cổng IO]***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Nhập dữ liệu từ cổng xuất nhập vào thanh ghi bộ tích lũy AL hay AX. Trường hợp AX sẽ nhập byte thấp trước, byte cao sau.
- Dạng lệnh có *immed8* dùng trong trường hợp địa chỉ cổng xuất nhập 8 bit.
- *Ví dụ* : IN AL,61h
  IN AX,40h
- Dạng lệnh có thanh ghi DX dùng cho trường hợp địa chỉ cổng 16 bit. Tuy nhiên dạng này vẫn có thể dùng cho cổng xuất nhập có địa chỉ 8 bit và có lợi khi sử dụng địa chỉ cổng để nhập nhiều lần.
- *Ví dụ* : MOV DX,378h
  IN AL,DX

2.6 Lệnh OUT :

- Dạng lệnh : OUT *immed8,accum*
  OUT *DX,accum*
- Giải thích : ***[cổng IO] ← btl***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Xuất dữ liệu từ thanh ghi bộ tích lũy AL hoặc AX ra cổng xuất nhập có địa chỉ 8 bit là số tức thời *immed8* hay có địa chỉ 16 bit trong thanh ghi DX.
- *Ví dụ* : OUT 20h,AL
  MOV DX,2F8h
  OUT DX,AL

2.7 Lệnh XLAT :

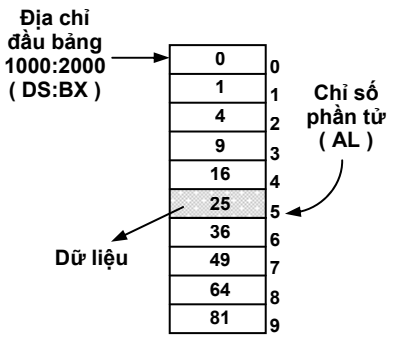
- Dạng lệnh : XLAT
- Giải thích : ***AL ← [DS:BX+AL]***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Tra bảng. Thanh ghi BX giữ địa chỉ đầu bảng. Thanh ghi AL giữ chỉ số của phần tử cần lấy ra.
- *Ví dụ* : bài toán tính bình phương một số nguyên có thể thực hiện bằng cách tra bảng như sau

```
MOV     CX,1000h
MOV     DS,CX
MOV     BX,2000h    ; địa chỉ đầu bảng
MOV     AL,5        ; chỉ số
XLAT                ; tra bảng
----------------------
```

Sau khi làm xong lệnh XLAT :
AL = 25 = 5^2



- Lệnh XLAT có ứng dụng trong mã hóa dữ liệu.

2.8 Lệnh LEA :
- Dạng lệnh : LEA *reg16,mem*
- Giải thích : ***thđ ← địa chỉ***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Nạp địa chỉ hiệu dụng vào thanh ghi 16 bit.
- *Ví dụ* : LEA BX,[1000h] ; *BX ← 1000h*
  LEA SI,[DI][BX][2000h] ; *SI ← DI+BX+2000h*

2.9 Lệnh LDS :
- Dạng lệnh : LDS *reg16,mem32*
- Giải thích : ***DS ← [địa chỉ+3,địa chỉ+2]***
  thđ ← [địa chỉ+1,địa chỉ]
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Nạp 4 byte bộ nhớ (con trỏ) vào thanh ghi DS và một thanh ghi tổng quát.
- *Ví dụ* : LDS BX,[1000h] ; *DS ← [1003h, 1002h]*
  ; *BX ← [1001h, 1000h]*

2.10 Lệnh LES :
- Dạng lệnh : LES *reg16,mem32*
- Giải thích : ***ES ← [địa chỉ+3,địa chỉ+2]***
  thđ ← [địa chỉ+1,địa chỉ]
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Nạp 4 byte bộ nhớ (con trỏ) vào thanh ghi ES và một thanh ghi tổng quát.
- *Ví dụ* : LES DI,[1000h] ; *ES ← [1003h, 1002h]*
  ; *SI ← [1001h, 1000h]*

2.11 Lệnh LAHF :
- Dạng lệnh : LAHF
- Giải thích : ***AH ← $Flags_L$***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Nạp 8 bit thấp của thanh ghi cờ vào thanh ghi AH.

2.12 Lệnh SAHF :
- Dạng lệnh : SAHF
- Giải thích : ***$Flags_L$ ← AH***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Cất thanh ghi AH vào 8 bit thấp của thanh ghi cờ.

2.13 Lệnh PUSHF :

- Dạng lệnh : PUSHF
- Giải thích : ***SP ← SP - 2***
  [SS:SP+1,SS:SP] ← Flags
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Đẩy thanh ghi cờ vào chồng.

2.14 Lệnh POPF :

- Dạng lệnh : POPF
- Giải thích : ***Flags ← [SS:SP+1,SS:SP]***
  SP ← SP + 2
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×	×	×	×	×	×	×	×

- Lấy thanh ghi cờ từ chồng ra.

3. Nhóm lệnh số học :

3.1 Lệnh ADD :

- Dạng lệnh : ADD *reg,reg* ADD *reg,immed*
  ADD *mem,reg* ADD *mem,immed*
  ADD *reg,mem* ADD *accum,immed*
- Giải thích : ***thđ ← thđ + thn***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	×

- Cộng toán hạng nguồn vào toán hạng đích. Kết quả cất vào toán hạng đích.
- *Ví dụ* :

```
ADD   CX,SI          ; CX  ← CX + SI
ADD   DH,BL          ; DH  ← DH + BL
ADD   [1000h],BX     ; [1001h,1000h]  ← [1001h,1000h] + BX
ADD   [2000h],CL     ; [2000h] ← [2000h] + CL
ADD   AL,[0000h]     ; AL  ← AL + [0000h]
ADD   BYTE PTR [SI+8],5   ; [SI+8] ← [SI+8] + 05h
```

3.2 Lệnh ADC :

- Dạng lệnh : ADC *reg,reg* ADC *reg,immed*
  ADC *mem,reg* ADC *mem,immed*
  ADC *reg,mem* ADC *accum,immed*
- Giải thích : ***thđ ← thđ + thn + CF***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	×

- Cộng toán hạng đích với toán hạng nguồn với cờ nhớ. Kết quả cất vào toán hạng đích. ADC dùng cho phép cộng 2 số có chiều dài nhiều byte.
- *Ví dụ* :

```
ADC   BX,AX                  ; BX ← BX + AX + CF
ADC   BYTE PTR [1000h],7Ah   ; [1000h] ← [1000h]+7Ah+CF
```

3.3 Lệnh INC :

- Dạng lệnh : INC *reg* INC *mem*

- Giải thích : ***thđ ← thđ + 1***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	

- Tăng tức là cộng 1 vào toán hạng đích nhưng không ảnh hưởng cờ nhớ.
- *Ví dụ :*
```
INC    CH
INC    WORD PTR [1000h]
```

3.4 Lệnh AAA :
- Dạng lệnh : AAA
- Giải thích : ***Nếu (b3b2b1b0 của AL) > 9 hoặc AF=1 thì***
  AL ← (AL+6) and 0Fh, AH ← AH+1, CF ← 1, AF ← 1
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
?			?	?	×	?	×

- Chỉnh ASCII sau phép cộng. Chỉnh kết quả trong AL thành 2 số BCD không nén trong AH và AL.
- *Ví dụ :*
```
kết quả :          AH=00, AL= 0Dh, AF=0, CF=0
sau khi chỉnh:     AH=01h, AL=03h, AF=1, CF=1
```

3.4 Lệnh DAA :
- Dạng lệnh : DAA
- Giải thích : ***Nếu (b3b2b1b0 của AL) > 9 hoặc AF=1 thì***
  AL ← (AL+6), AF ← 1
  Nếu AL > 9Fh hoặc CF=1 thì
  AL ← AL+60h, CF ← 1
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
?			?	?	×	?	×

- Chỉnh thập phân sau phép cộng. Chỉnh kết quả trong AL thành số BCD nén trong AL.
- *Ví dụ :*
```
kết quả :          AL= 0Dh, AF=0, CF=0
sau khi chỉnh:     AL=13h, AF=1, CF=0
kết quả :          AL= 9Dh, AF=0, CF=0
sau khi chỉnh:     AL=03h, AF=1, CF=1
```

3.5 Lệnh SUB :
- Dạng lệnh :
```
SUB   reg,reg          SUB   reg,immed
SUB   mem,reg          SUB   mem,immed
SUB   reg,mem          SUB   accum,immed
```
- Giải thích : ***thđ ← thđ - thn***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	×

- Trừ toán hạng đích cho toán hạng nguồn. Kết quả cất vào toán hạng đích.
- *Ví dụ* :
```
SUB   DL,AL        ; DL ← DL - AL
SUB   CX,[DI]      ; CX ← CX - [DI+1,DI]
SUB   BP,4         ; BP ← BP - 4
```

3.6 Lệnh SBB :
- Dạng lệnh :
```
SBB   reg,reg          SBB   reg,immed
```

SBB *mem,reg* SBB *mem,immed*
SBB *reg,mem* SBB *accum,immed*

- Giải thích : ***thđ ← thđ - thn - CF***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	×

- Trừ toán hạng đích cho toán hạng nguồn và cờ nhớ. Kết quả cất vào toán hạng đích.
- *Ví dụ* : SBB SI,BX ; *SI ← SI - BX -CF*
  SBB BYTE PTR [BX],2 ; *[BX+1,BX] ← [BX+1,BX] - 2 - CF*

3.7 Lệnh DEC :

- Dạng lệnh : DEC *reg* DEC *mem*
- Giải thích : ***thđ ← thđ - 1***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	

- Giảm tức là trừ 1 vào toán hạng đích nhưng không ảnh hưởng cờ nhớ.
- *Ví dụ :* DEC AX
  DEC BYTE PTR [SI][2000h]

3.8 Lệnh NEG :

- Dạng lệnh : NEG *reg* NEG *mem*
- Giải thích : ***thđ ← bù 2(thđ)***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	×

- Lấy bù 2 toán hạng đích.

3.9 Lệnh CMP :

- Dạng lệnh : CMP *reg,reg* CMP *reg,immed*
  CMP *mem,reg* CMP *mem,immed*
  CMP *reg,mem* CMP *accum,immed*
- Giải thích : ***thđ - thn***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	×

- So sánh. Thực hiện trừ toán hạng đích cho toán hạng nguồn, không lưu lại kết quả mà chỉ giữ lại tác động của phép trừ lên các cờ.
- *Ví dụ* : CMP AL,8 ; *AL - 8*
  CMP WORD PTR [1000h], 3 ; *[1001h,1000h] - 3*

3.10 Lệnh AAS :

- Dạng lệnh : AAS
- Giải thích : ***Nếu ($D_3D_2D_1D_0$ của AL) > 9 hoặc AF=1 thì***
  AL ← (AL - 6) and 0Fh, AH ← AH - 1, CF ← 1, AF ← 1
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
?			?	?	×	?	×

- Chỉnh ASCII sau phép cộng. Chỉnh kết quả trong AL thành 2 số BCD không nén trong AH và AL.
- *Ví dụ :* kết quả : AH=00h, AL= 0Dh, AF=0, CF=0

sau khi chỉnh: AH=01h, AL=03h, AF=1, CF=1

3.11 Lệnh DAS :

- Dạng lệnh : DAS
- Giải thích : ***Nếu ($D_3D_2D_1D_0$ của AL) > 9 hoặc AF=1 thì***
  AL ← (AL - 6), AF ← 1
  Nếu AL > 9Fh hoặc CF=1 thì
  AL ← AL - 60h, CF ← 1
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
?			×	×	×	×	×

- Chỉnh thập phân sau phép trừ. Chỉnh kết quả trong AL thành số BCD nén trong AL.
- *Ví dụ :* kết quả của (4 - 8) : AL= 0FCh, AF=1, CF=1
  sau khi chỉnh : AL=96h, AF=1, CF=1

3.12 Lệnh MUL :

- Dạng lệnh : MUL *reg* MUL *mem*
- Giải thích : ***Toán hạng nguồn 8 bit thì : AX ← AL * thn8***
  Toán hạng nguồn 16 bit thì : DX AX ← AX * thn16
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			?	?	?	?	×

- Nhân hai số không dấu 8 bit hay 16 bit. Số bit thực hiện được xác định bằng chiều dài của toán hạng nguồn.
  - ♣ Phép nhân 8 bit : thực hiện nhân AL với toán hạng nguồn, kết quả 16 bit cất trong thanh ghi AX.
  - ♣ Phép nhân 16 bit : thực hiện nhân AX với toán hạng nguồn, kết quả 32 bit cất trong 2 thanh ghi DX và AX. DX giữ 16 bit cao, AX giữ 16 bit thấp.
- *Ví dụ :* Nếu AL=5, CH=4, sau khi thực hiện lệnh
  MUL CH
  ta có AX = AL*CH = 0014h.
  Nếu AX=500h, [1001h,1000h]=401h, sau khi thực hiện lệnh
  MUL WORD PTR [1000h]
  ta có DXAX = AX * [1001h,1000h] = 500h * 401h = 00140500h
  Nghĩa là DX=0014h và AX=0500h.

3.13 Lệnh IMUL :

- Dạng lệnh : IMUL *reg* IMUL *mem*
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			?	?	?	?	×

- Nhân hai số có dấu. Thực hiện giống hệt như lệnh MUL, chỉ có kết quả được xem là số có dấu.

3.14 Lệnh AAM :

- Dạng lệnh : AAM
- Giải thích : ***AH ← (AL / 0Ah)***
  AL ← số dư của (AL / 0Ah)
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
?			×	×	?	×	?

- Chỉnh ASCII sau phép nhân. Có thể dùng để đổi số hex ra số BCD không nén.
- *Ví dụ :* kết quả : AH = 00, AL = 41h.
  sau khi chỉnh : AH = 06, AL = 05.

3.15 Lệnh DIV :

- Dạng lệnh : DIV *reg* DIV *mem*
- Giải thích : ***Toán hạng nguồn 8 bit thì : AL ← (AX / thn8)***
  AH ← số dư của (AX / thn8)
  Toán hạng nguồn 16 bit thì : AX ← (DXAX / thn16)
  DX ← số dư của (DXAX / thn16)
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
?			?	?	?	?	?

- Chia hai số không dấu.
- Nếu toán hạng nguồn là thanh ghi hay bộ nhớ 8 bit, thực hiện chia số 16 bit trong thanh ghi AX cho toán hạng nguồn 8 bit. Kết quả 8 bit cất trong thanh ghi AL. Số dư 8 bit cất trong thanh ghi AH.
- Nếu toán hạng nguồn là thanh ghi hay bộ nhớ 16 bit, thực hiện chia số 32 bit trong 2 thanh ghi DXAX cho toán hạng nguồn 16 bit. Kết quả 16 bit cất trong thanh ghi AX. Số dư 16 bit cất trong thanh ghi DX.
- *Ví dụ :* Nếu AX=0024h, [2000h]=05 thì sau khi thực hiện lệnh
  DIV BYTE PTR [2000h]
  ta có *AL=07 và AH=01*.
  Nếu DX=0001h, AX=0024h, BX=0200h thì sau khi thực hiện lệnh
  DIV BX
  ta có *AX=0008 và DX=0024h.*

3.16 Lệnh IDIV :

- Dạng lệnh : IDIV *reg* IDIV *mem*
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
?			?	?	?	?	?

- Chia hai số có dấu. Thực hiện giống như lệnh DIV nhưng kết quả coi là số có dấu.

3.17 Lệnh AAD :

- Dạng lệnh : AAD
- Giải thích : ***AL ← ((AH * 0Ah) + AL)***
  AH ← 0
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
?			×	×	?	×	?

- Chỉnh ASCII trước phép chia IDIV. Có thể dùng lệnh này để đổi số BCD không nén trong AX ra thành giá trị nhị phân trong AL.
- *Ví dụ :* nếu có : AL=03h, AH=05h
  sau khi chỉnh : AL=35h, AH=00h

3.18 Lệnh CBW :

- Dạng lệnh : CBW

- Giải thích : ***Nếu AL < 80h thì AH ← 00h***
  Nếu AL >= 80h thì AH ← 0FFh
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Mở rộng dấu trước khi dùng lệnh chia. Đối số 1 byte có dấu trong AL thành số 2 byte có dấu trong AX.

3.19 Lệnh CWD :
- Dạng lệnh : CWD
- Giải thích : ***Nếu AX < 8000h thì DX ← 0000h***
  Nếu AX >= 8000h thì DX ← 0FFFFh
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Mở rộng dấu trước khi dùng lệnh chia. Đối số 2 byte có dấu trong AX thành số 4 byte có dấu trong DXAX.

4. Nhóm lệnh luận lý :

4.1 Lệnh NOT :
- Dạng lệnh : NOT *reg* NOT *mem*
- Giải thích : ***thđ ← bÓ 1 của thđ***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Đảo hay lấy bù 1.
- *Ví dụ :*
```
NOT   AL
NOT   WORD PTR [BX+1000h]
```

4.2 Lệnh SHL/SAL :
- Dạng lệnh : SHL *reg,1* SHL *mem,1*
  SHL *reg,CL* SHL *mem,CL*
- Giải thích : ***thđ ← (thđ) dịch trái 1 hay nhiều bit.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	×

- Dịch trái. Dạng SHL reg,1 dùng để dịch trái 1 bit. Dạng SHL reg,CL dùng để dịch trái nhiều bit. Lúc đó thanh ghi CL chứa số bit cần dịch.
- *Ví dụ :*
```
SHL   DH,1
SAL   CX,1
MOV   CL,3
SHL   WORD PTR [1000h],CL      ; dịch trái 3 bit.
```

4.3 Lệnh SHR :
- Dạng lệnh : SHR *reg,1* SHR *mem,1*
  SHR *reg,CL* SHR *mem,CL*
- Giải thích : ***thđ ← (thđ) dịch phải luận lý 1 hay nhiều bit.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	×

- Dịch phải luận lý. Dạng có thanh ghi CL dùng để dịch nhiều bit.
- *Ví dụ :*
```
SHR   AX,1
```

```
MOV  CL,2
SHR  BYTE PTR [1000h],CL        ; dịch phải luận lý 2 bit.
```

4.4 Lệnh SAR :

- Dạng lệnh : SAR *reg,1* SAR *mem,1*
  SAR *reg,CL* SAR *mem,CL*
- Giải thích : ***thđ ← (thđ) dịch phải số học 1 hay nhiều bit.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	×

- Dịch phải số học. Dạng có thanh ghi CL dùng để dịch nhiều bit.
- *Ví dụ :*

```
SAR  DX,1
MOV  CL,7
SAR  WORD PTR [2000h],CL        ; dịch phải số học 7 bit.
```

4.5 Lệnh ROL :

- Dạng lệnh : ROL *reg,1* ROL *mem,1*
  ROL *reg,CL* ROL *mem,CL*
- Giải thích : ***thđ ← (thđ) quay trái không qua cờ nhớ 1 hay nhiều bit.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	×

- Quay trái không qua cờ nhớ. Dạng có thanh ghi CL dùng để quay nhiều bit.
- *Ví dụ :*

```
ROL  DL,1
MOV  CL,6
ROL  WORD PTR [1000h],CL  ; quay trái không qua cờ nhớ 6 bit.
```

4.6 Lệnh ROR :

- Dạng lệnh : ROR *reg,1* ROR *mem,1*
  ROR *reg,CL* ROR *mem,CL*
- Giải thích : ***thđ ← (thđ) quay phải không qua cờ nhớ 1 hay nhiều bit.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	×

- Quay phải không qua cờ nhớ. Dạng có thanh ghi CL dùng để quay nhiều bit.
- *Ví dụ :*

```
ROR  SI,1
MOV  CL,3
ROR  WORD PTR [3000h],CL  ; quay phải không qua cờ nhớ 3 bit
```

4.7 Lệnh RCL :

- Dạng lệnh : RCL *reg,1* RCL *mem,1*
  RCL *reg,CL* RCL *mem,CL*
- Giải thích : ***thđ ← (thđ) quay trái qua cờ nhớ 1 hay nhiều bit.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	×

- Quay trái qua cờ nhớ. Dạng có thanh ghi CL dùng để quay nhiều bit.
- *Ví dụ :*

```
RCL  BX,1
MOV  CL,4
RCL  BYTE PTR [1000h],CL        ; quay trái qua cờ nhớ 4 bit.
```

4.8 Lệnh RCR :

- Dạng lệnh : RCR *reg,1* RCR *mem,1*
  RCR *reg,CL* RCR *mem,CL*
- Giải thích : ***thđ ← (thđ) quay phải qua cờ nhớ 1 hay nhiều bit.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	×

- Quay phải qua cờ nhớ. Dạng có thanh ghi CL dùng để quay nhiều bit.
- *Ví dụ :*

```
RCR   CH,1
MOV   CL,2
RCR   BYTE PTR [1800h],CL      ; quay phải  qua cờ nhớ 2 bit.
```

4.9 Lệnh AND :

- Dạng lệnh : AND *reg,reg* AND *reg,immed*
  AND *mem,reg* AND *mem,immed*
  AND *reg,mem* AND *accum,immed*
- Giải thích : ***thđ ← thđ AND thn.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	0

- Và luận lý. Xóa cờ nhớ về 0.
- *Ví dụ :*

```
AND   CH,AH
AND   [SI],DX
AND   BYTE PTR [1000h],10000000b
AND   AX,0FFF0h
```

4.10 Lệnh TEST :

- Dạng lệnh : TEST *reg,reg* TEST *reg,immed*
  TEST *mem,reg* TEST *mem,immed*
  TEST *reg,mem* TEST *accum,immed*
- Giải thích : ***thđ AND thn.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
0			×	×	?	×	0

- Và luận lý hai toán hạng nhưng không giữ lại kết quả mà chỉ lập các cờ. Xóa cờ nhớ và cờ tràn về 0. Thường dùng để kiểm tra bit. Lúc đó toán hạng nguồn là một mặt nạ bit cần thiết.
- *Ví dụ :*

```
TEST  DX,1                      ; kiểm tra bit 0
TEST  BYTE PTR [2000h],10000000b ; kiểm tra bit 7
```

4.11 Lệnh OR :

- Dạng lệnh : OR *reg,reg* OR *reg,immed*
  OR *mem,reg* OR *mem,immed*
  OR *reg,mem* OR *accum,immed*
- Giải thích : ***thđ ← thđ OR thn.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	0

- Hay luận lý. Xóa cờ nhớ về 0.
- *Ví dụ :*

```
OR    DL,CH
```

```
OR      BP,[2000h]
OR      WORD PTR [1000h],000Fh
```

4.12 Lệnh XOR :

- Dạng lệnh : XOR *reg,reg* XOR *reg,immed*
  XOR *mem,reg* XOR *mem,immed*
  XOR *reg,mem* XOR *accum,immed*
- Giải thích : ***thđ ← thđ XOR thn.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	?	×	0

- Hay ngoại luận lý. Xóa cờ nhớ về 0.
- *Ví dụ :*

```
XOR   DL,80h          ; đảo bit 7
XOR   [2000h],AL      ; [2000h] ← [2000h] XOR AL
```

5. Xử lý chuỗi :

5.1 Tiếp đầu lệnh REP :

♣ Dạng 1 : REP *lệnh xử lý chuỗi*

- Giải thích : ***giảm CX, lặp lại lệnh theo sau Nếu CX ≠ 0***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Lặp lại không điều kiện. Thanh ghi CX giữ số lần lặp. Thường dùng với lệnh chép chuỗi MOVS.
- *Ví dụ :*

```
MOV   CX,10
REP   MOVSB   ; thực hiện lệnh MOVSB 10 lần.
```

♣ Dạng 2 : REPE / REPZ *lệnh xử lý chuỗi*

- Giải thích : ***giảm CX, lặp lại lệnh theo sau Nếu CX ≠ 0 và ZF = 1***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Lặp lại nếu bằng / nếu không. Hai điều kiện CX ≠ 0 và ZF = 1 phải thỏa đồng thời thì lệnh theo sau mới được lặp lại. Nếu không, làm qua lệnh kế. Thường dùng với lệnh so sánh chuỗi CMPS để tìm chuỗi con trong chuỗi lớn.
- *Ví dụ :*

```
MOV   CX,10
REPE  CMPSB     ; thực hiện lệnh CMPSB nếu chưa đủ 10
                ; lần và hai chuỗi vẫn còn bằng nhau.
```

♣ Dạng 3 : REPNE / REPNZ *lệnh xử lý chuỗi*

- Giải thích : ***giảm CX, lặp lại lệnh theo sau Nếu CX ≠ 0 và ZF = 0***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Lặp lại nếu bằng / nếu không. Hai điều kiện CX ≠ 0 và ZF = 1 phải thỏa đồng thời thì lệnh theo sau mới được lặp lại. Nếu không, làm qua lệnh kế. Thường dùng với lệnh quét chuỗi SCAS để dò tìm ký tự tong chuỗi.
- *Ví dụ :*

```
MOV    CX,20
REPNE  SCASB    ; thực hiện lệnh CMPSB nếu chưa đủ 20
                ; lần và chưa tìm ra ký tự trong chuỗi.
```

5.2 Lệnh MOVS :

- Dạng lệnh : MOVSB
  MOVSW
- Giải thích :
  - ♣ MOVSB : ***[ES:DI] ← [DS:SI]***
    Nếu DF=0 thì : SI ← SI + 1, DI← DI + 1
    ngược lại thì : SI ← SI - 1, DI← DI - 1
  - ♣ MOVSW : ***[ES:DI+1,ES:DI] ← [DS:SI+1,DS:SI]***
    Nếu DF=0 thì : SI ← SI + 2, DI← DI + 2
    ngược lại thì : SI ← SI - 2, DI← DI - 2
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Chép byte hay word từ nguồn sang chuỗi đích. Cặp thanh ghi DS:SI giữ địa chỉ chuỗi nguồn. Cặp thanh ghi ES:DI giữ địa chỉ chuỗi đích. Các địa chỉ chuỗi nguồn trong thanh ghi SI và địa chỉ chuỗi đích trong thanh ghi DI được tự động tăng hay giảm sau mỗi lần chép. Chiều tăng giảm địa chỉ tùy thuộc cờ định hướng DF. DF=0 xử lý tăng địa chỉ. DF=1 xử lý giảm địa chỉ.
- Lệnh này thường dùng kết hợp với tiếp đầu lệnh REP để thực hiện việc chép một chuỗi hay dãy. Lúc đó thanh ghi CX giữ chiều dài chuỗi nguồn.
- *Ví dụ 1 chép byte* : chép 80h byte từ địa chỉ 3000:1000 sang địa chỉ 4800:C200

```
MOV  AX,3000h
MOV  DS,AX
MOV  SI,1000h      ; địa chỉ chuỗi nguồn.
MOV  AX,4800h
MOV  ES,AX
MOV  DI,0C200h     ; địa chỉ chuỗi đích.
MOV  CX,80h        ; số lần chép.
CLD                ; xóa cờ DF, xử lý tăng địa chỉ.
REP  MOVSB
```

- *Ví dụ 2 chép word* : yêu cầu như ví dụ 1

```
MOV  AX,3000h
MOV  DS,AX
MOV  SI,1000h      ; địa chỉ chuỗi nguồn.
MOV  AX,4800h
MOV  ES,AX
MOV  DI,0C200h     ; địa chỉ chuỗi đích.
MOV  CX,40h        ; số lần chép.
CLD                ; xóa cờ DF, xử lý tăng địa chỉ.
REP  MOVSW
```

5.3 Lệnh CMPS :

- Dạng lệnh : CMPSB
  CMPSW
- Giải thích :
  - ♣ CMPSB : ***[DS:SI] - [ES:DI]***

Nếu DF=0 thì : SI ← SI + 1, DI← DI + 1
ngược lại thì : SI ← SI - 1, DI← DI - 1

♣ CMPSW : ***[DS:SI+1,DS:SI] - [ES:DI+1,ES:DI]***
Nếu DF=0 thì : SI ← SI + 2, DI← DI + 2
ngược lại thì : SI ← SI - 2, DI← DI - 2

- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	×

- So sánh byte hay word của chuỗi nguồn với chuỗi đích. Cặp thanh ghi DS:SI giữ địa chỉ chuỗi nguồn. Cặp thanh ghi ES:DI giữ địa chỉ chuỗi đích. Các thanh ghi giữ địa chỉ offset SI, DI được tự động tăng hay giảm sau mỗi lần so sánh. Chiều tăng giảm địa chỉ tùy thuộc cờ định hướng DF. DF=0 xử lý tăng địa chỉ. DF=1 xử lý giảm địa chỉ.
- Lệnh này thường dùng kết hợp với tiếp đầu lệnh REPE để thực hiện việc so sánh hai chuỗi hay hai dãy với nhau để tìm kiếm một chuỗi con trong một chuỗi lớn. Lúc đó thanh ghi CX giữ chiều dài chuỗi.
- Có thể có hai nguyên nhân làm ngừng lệnh so sánh chuỗi : hoặc hai chuỗi có byte hay word khác nhau (ZF = 0), hoặc hai chuỗi giống nhau (ZF = 1).
- *Ví dụ :* REPE CMPSB

5.4 Lệnh SCAS :

- Dạng lệnh : SCASB
  SCASW
- Giải thích :

♣ SCASB : ***AL - [ES:DI]***
Nếu DF=0 thì : DI← DI + 1
ngược lại thì : DI← DI - 1

♣ SCASW : ***AX - [ES:DI+1,ES:DI]***
Nếu DF=0 thì : DI← DI + 2
ngược lại thì : DI← DI - 2

- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×			×	×	×	×	×

- Quét chuỗi nghĩa là so sánh byte trong thanh ghi AL hay word trong thanh ghi AX với chuỗi đích. Cặp thanh ghi ES:DI giữ địa chỉ chuỗi đích. Địa chỉ chuỗi đích được tự động tăng hay giảm sau mỗi lần so sánh. Chiều tăng giảm địa chỉ tùy thuộc cờ định hướng DF. DF=0 xử lý tăng địa chỉ. DF=1 xử lý giảm địa chỉ.
- Lệnh này thường dùng kết hợp với tiếp đầu lệnh REPNE để thực hiện việc tìm kiếm một dữ liệu trong một chuỗi. Lúc đó thanh ghi CX giữ chiều dài chuỗi.
- Có thể có hai nguyên nhân làm ngừng lệnh quét chuỗi : hoặc tìm thấy dữ liệu trong chuỗi (ZF=1 hay CX 0), hoặc hết chuỗi mà vẫn chưa tìm thấy dữ liệu (ZF=0 hay CX=0).
- *Ví dụ :* REPNE SCASW

5.5 Lệnh LODS :

- Dạng lệnh : LODSB
  LODSW
- Giải thích :

♣ LODSB : ***AL ← [DS:SI]***
Nếu DF=0 thì : SI ← SI + 1
ngược lại thì : SI ← SI - 1

♣ LODSW : ***AX ← [DS:SI+1,DS:SI]***
Nếu DF=0 thì : SI ← SI + 2
ngược lại thì : SI ← SI - 2

- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Nạp chuỗi nguồn byte vào thanh ghi AL hay chuỗi nguồn word vào thanh ghi AX. Cặp thanh ghi DS:SI giữ địa chỉ chuỗi nguồn. Địa chỉ chuỗi nguồn được tự động tăng hay giảm sau mỗi lần nạp. Chiều tăng giảm địa chỉ tùy thuộc cờ định hướng DF. DF=0 xử lý tăng địa chỉ. DF=1 xử lý giảm địa chỉ.

5.6 Lệnh STOS :

- Dạng lệnh : STOSB
STOSW
- Giải thích :

♣ STOSB : ***[ES:DI] ← AL***
Nếu DF=0 thì : DI← DI + 1
ngược lại thì : DI← DI - 1

♣ STOSW : ***[ES:DI+1,ES:DI] ← AX***
Nếu DF=0 thì : DI← DI + 2
ngược lại thì : DI← DI - 2

- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Cất byte trong thanh ghi AL hay word trong thanh ghi AX vào chuỗi đích. Cặp thanh ghi ES:DI giữ địa chỉ chuỗi đích. Địa chỉ chuỗi đích được tự động tăng hay giảm sau mỗi lần cất. Chiều tăng giảm địa chỉ tùy thuộc cờ định hướng DF. DF=0 xử lý tăng địa chỉ. DF=1 xử lý giảm địa chỉ.

6. Chuyển điều khiển :

6.1 Lệnh CALL :

- Dạng lệnh : CALL *nearlabel* CALL *mem16*
CALL *farlabel* CALL *mem32*
CALL *reg16*
- Giải thích :

♣ *nearlabel* : ***PUSH IP***
IP ← địa chỉ lệnh kế + độ dời 2 byte

♣ *farlabel* : ***PUSH CS***
PUSH IP
CS ← địa chỉ segment
IP ← địa chỉ offset

♣ *reg16* : ***PUSH IP***
IP ← reg16

♣ *mem16* : ***PUSH IP***

IP ← [địa chỉ +1,địa chỉ]

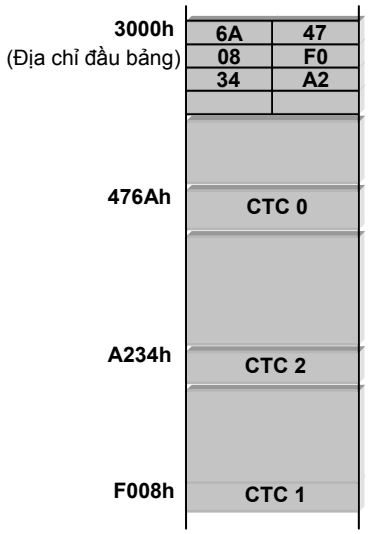
♣ *mem32* : ***PUSH CS***
PUSH IP
CS ← [địa chỉ +3,địa chỉ+2]
IP ← [địa chỉ +1,địa chỉ]

- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Gọi chương trình con. Quá trình gọi chương trình con được thực hiện qua 2 bước :
  - ♣ Cất địa chỉ trở về - chính là địa chỉ lệnh ngay sau lệnh CALL - vào chồng.
  - ♣ Chuyển sự thi hành chương trình đến địa chỉ lệnh đầu tiên của chương trình con.
- Địa chỉ trở về chính là nội dung hiện tại của cặp thanh ghi CS:IP.
- Lệnh gọi trực tiếp đến nhãn nearlabel chỉ cất nội dung IP, và nạp giá trị offset mới vào IP (nội dung CS không đổi) nên chỉ có thể dùng để gọi bên trong một segment. Lệnh này còn được gọi là lệnh gọi gần hay gọi trong segment. Nhãn nearlabel còn được gọi là nhãn gần và có kích thước 2 byte.
  Ví dụ : CALL 0F008h
- Lệnh gọi trực tiếp đến nhãn farlabel cất nội dung IP lẫn CS, sau đó nạp giá trị offset mới vào IP, nạp giá trị segment mới vào CS nên có thể dùng để gọi đến bất kỳ vị trí bộ nhớ nào cũng được. Lệnh này còn được gọi là lệnh gọi xa hay gọi ngoài segment. Nhãn farlabel còn được gọi là nhãn xa và có kích thước 4 byte.
  Ví dụ : CALL 3000:F008
- Lệnh gọi gián tiếp qua thanh ghi reg16 cũng là một lệnh gọi gần. Lúc đó địa chỉ chương trình con được nạp vào thanh ghi trước khi thực hiện lệnh gọi.
  Ví dụ : MOV DX,0F008h
  CALL DX
- Lệnh gọi gián tiếp qua bộ nhớ mem16 cũng là một lệnh gọi gần. Lúc đó địa chỉ chương trình con được đặt tại ô nhớ có địa chỉ hiệu dụng trong lệnh.
  Ví dụ : CALL [BX+3000h]
- Lệnh gọi gián tiếp qua bộ nhớ mem32 là một lệnh gọi xa. Lúc đó địa chỉ chương trình con đặt tại ô nhớ có địa chỉ hiệu dụng trong lệnh phải là địa chỉ 4 byte. Lúc đó cần phải chỉ rõ ra hoạt động bộ nhớ 32 bit bằng cách dùng toán tử DWORD PTR.
  Ví dụ : CALL DWORD PTR [SI+2000h]
- Với lệnh gọi gián tiếp qua bộ nhớ ta có thể tổ chức sắp xếp các địa chỉ chương trình con thành một bảng trong bộ nhớ gọi là bảng nhảy. Lúc đó mỗi chương trình con sẽ được gọi theo số thứ tự của nó trong bảng nhảy.
- *Ví dụ bảng nhảy gần :*

Chương trình con 0 ở địa chỉ 476Ah.
Chương trình con 1 ở địa chỉ 0F008h.
Chương trình con 2 ở địa chỉ 0A234h.



Để gọi chương trình con 2 ta thực hiện :

```
MOV   BX,2      ; số thứ tự chương trình con.
ADD   BX,BX     ; nhân 2.
CALL  [BX+3000h]  ; gọi chương trình con.
```

6.2 Lệnh JMP :

- Dạng lệnh : JMP *shortlabel* JMP *mem16*
  JMP *nearlabel* JMP *mem32*
  JMP *farlabel* JMP *reg16*
- Giải thích :
  - ♣ *shortlabel* : ***IP ← IP + độ dời (mở rộng dấu 16 bit)***
  - ♣ *nearlabel* : ***IP ← địa chỉ***
  - ♣ *farlabel* : ***CS ← địa chỉ segment***
    IP ← địa chỉ offset
  - ♣ *reg16 :* ***IP ← reg16***
  - ♣ *mem16* : ***IP ← [địa chỉ +1,địa chỉ]***
  - ♣ *mem32* : ***CS ← [địa chỉ +3,địa chỉ+2]***
    IP ← [địa chỉ +1,địa chỉ]
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Nhảy không điều kiện. Lệnh nhảy không điều kiện thực hiện giống như lệnh gọi nhưng không có bước lưu lại địa chỉ trở về.
- Lệnh nhảy đến nhãn ngắn shortlabel là lệnh nhảy tương đối. Nơi đến phải nằm trong phạm vi từ -128 đến +127 so với vị trí của lệnh nhảy. Toán hạng nguồn trong lệnh chỉ là byte độ dời để cộng thêm vào thanh ghi IP. Byte độ dời này được mở rộng dấu trước khi cộng vào thanh ghi IP.
- *Ví dụ :*
```
JMP   SHORT   18h
JMP   0F008h
JMP   DWORD   PTR   [3000h]
```

6.3 Lệnh RET :

- Dạng lệnh : RET RETF
  RET *immed8* RETF *immed8*
- Giải thích :
  - ♣ *RET* : ***POP IP***
  - ♣ *RETF* : ***POP IP***
    POP CS
  - ♣ *RET immed8* : ***POP IP***
    SP ← SP + immed8
  - ♣ *RETF immed8* : ***POP IP***
    POP CS
    SP ← SP + immed8
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Trở về từ chương trình con. Lệnh trở về là lệnh dùng để kết thúc một chương trình con.
- Lệnh RET để kết thúc một chương trình con gần.
- Lệnh RETF để kết thúc một chương trình con xa.
- Dạng lệnh trở về có toán hạng *immed8* dùng cho các chương trình con có sử dụng thông số trong chồng. Khi đó, toán hạng nguồn *immed8* sẽ được cộng vào thanh ghi

SP để chỉnh lại vị trí đỉnh chồng sau khi gọi chương trình con, tránh thất thoát bộ nhớ dùng cho chồng.

6.4 Lệnh nhảy có điều kiện :

- Dạng lệnh : J*cond* *shortlabel*
- Giải thích : ***Nếu thỏa điều kiện thì nhảy tương đối***
  IP ← địa chỉ lệnh kế + độ dời (mở rộng dấu 16 bit)
  ngược lại không làm gì cả (qua lệnh kế).
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Lệnh nhảy có điều kiện dùng trạng thái các cờ để làm điều kiện.
- Sau đây là bảng mã lệnh nhảy có điều kiện cùng với điều kiện nhảy.

Mã lệnh	Giải thích	Điều kiện
JE/JZ	**Nhảy nếu bằng/không**	**ZF = 1**
JL/JNGE	**Nhảy nếu nhỏ hơn/không lớn hơn hoặc bằng**	**(SF xor OF) = 1**
JLE/JNG	**Nhảy nếu nhỏ hơn hoặc bằng /không lớn hơn**	**((SF xor OF) or ZF) = 1**
JB/JNAE/JC	**Nhảy nếu dưới /không trên hoặc bằng/nhớ**	**CF = 1**
JBE/JNA	**Nhảy nếu dưới hoặc bằng /không trên**	**(CF or ZF) = 1**
JP/JPE	**Nhảy nếu kiểm tra / kiểm tra chẵn**	**PF = 1**
JO	**Nhảy nếu tràn**	**OF = 1**
JS	**Nhảy nếu dấu**	**SF = 1**
JNE/JNZ	**Nhảy nếu không bằng/khác không**	**ZF = 0**
JNL/JGE	**Nhảy nếu không nhỏ hơn/lớn hơn hoặc bằng**	**(SF xor OF) = 0**
JNLE/JG	**Nhảy nếu không nhỏ hơn hoặc bằng /lớn hơn**	**((SF xor OF) or ZF) = 0**
JNB/JAE/JNC	**Nhảy nếu không dưới /trên hoặc bằng/không nhớ**	**CF = 0**
JNBE/JA	**Nhảy nếu không dưới hoặc bằng /trên**	**(CF or ZF) = 0**
JNP/JPO	**Nhảy nếu không kiểm tra / kiểm tra lẻ**	**PF = 0**
JNO	**Nhảy nếu không tràn**	**OF = 0**
JNS	**Nhảy nếu không dấu**	**SF = 0**

- *Ví dụ :*

```
        MOV   CX,3          ; thực hiện một vòng lặp làm 3 lần.
        MOV   AX,0
Nhan:   ADD   AX,12
        DEC   CX
        JNZ   Nhan          ; nhảy đến lệnh tại vị trí "Nhan" nếu CX ≠ 0.
        MOV   [3000h],AX
```

6.5 Lệnh LOOP :

♣ Dạng lệnh 1 : LOOP *shortlabel*

- Giải thích : ***giảm CX, lặp vòng (nhảy) nếu CX ≠ 0***
  IP ← địa chỉ lệnh kế + độ dời (mở rộng dấu 16 bit)
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Lập vòng không điều kiện. CX giữ số lần lặp. Rất tiện dụng trong việc tạo ra các vòng lặp. Chẳng hạn như ví dụ trong phần lệnh nhảy có điều kiện có thể viết lại gọn hơn như sau :

```
        MOV   CX,3
```

```
        MOV     AX,0
Nhan:   ADD     AX,12
        LOOP    Nhan
        MOV     [3000h],AX
```

- Một trong những ứng dụng phổ biến của lệnh lặp vòng là tạo ra các vòng làm trễ.
- *Ví dụ :*

```
        MOV     CX,10
Nhan:   LOOP    Nhan      ; vòng lặp làm 10 lần lệnh LOOP

        MOV     CX,0
Nhan:   LOOP    Nhan      ; vòng lặp làm 65536 lần lệnh LOOP
```

♣ Dạng lệnh 2 : LOOPE/LOOPZ *shortlabel*

- Giải thích : ***giảm CX, lặp vòng (nhảy) nếu CX ≠ 0 và ZF = 1***
  IP ← địa chỉ lệnh kế + độ dời (mở rộng dấu 16 bit)
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Lập vòng nếu bằng / nếu không. Hai điều kiện CX ≠ 0 *và* ZF = 1 phải thỏa đồng thời thì lệnh mới lặp vòng. Nếu không, không làm gì cả (qua lệnh kế).
- Đôi khi người ta xét các điều kiện để không lặp còn gọi là điều kiện thoát khỏi vòng lặp : CX = 0 hay ZF = 0. Có thể xem đó là các nguyên nhân gây ra kết thúc vòng lặp.

♣ Dạng lệnh 3 : LOOPNE/LOOPNZ *shortlabel*

- Giải thích : ***giảm CX, lặp vòng (nhảy) nếu CX ≠ 0 và ZF = 0***
  IP ← địa chỉ lệnh kế + độ dời (mở rộng dấu 16 bit)
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Lập vÎng nău khéng bÙng / nău kh¾c khéng. Hai ½iảu kiền CX ≠ 0 và ZF = 0 phải thỏa đồng thời thì lệnh mới lặp vòng. Nếu không, không làm gì cả (qua lệnh kế).
- Đôi khi người ta xét các điều kiện để không lặp còn gọi là điều kiện thoát khỏi vòng lặp : CX = 0 hay ZF = 1. Có thể xem đó là các nguyên nhân gây ra kết thúc vòng lặp.

6.6 Lệnh JCXZ :

- Dạng lệnh : JCXZ *shortlabel*
- Giải thích : ***Nếu CX = 0 thì***
  IP ← địa chỉ lệnh kế + độ dời (mở rộng dấu 16 bit)
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Nhảy nếu CX=0. Thường dùng sau LOOPE, LOOPNE, REPE, REPNE để xác định nguyên nhân kết thúc vòng lặp.
- *Ví dụ :*

```
        REPE  CMPSB       ; so sánh 2 chuỗi
        JCXZ  nhan
        ----------------------------------------------------------------
        ( đoạn chương trình xử lý cho trường hợp chuỗi khác nhau)

        ----------------------------------------------------------------
nhan:   ( đoạn chương trình xử lý cho trường hợp chuỗi giống nhau)
```

6.7 Lệnh INT :

- Dạng lệnh : INT *immed8*
  INT *3*
- Giải thích : ***PUSHF***
  PUSH CS
  PUSH IP
  CS ← [(số ngắt * 4)+3, (số ngắt * 4)+2]
  IP ← [(số ngắt * 4)+1, (số ngắt * 4)]
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
		0					

- Ngắt quãng mềm. Thực chất của lệnh ngắt quãng là gọi đến một chương trình con đặc biệt gọi là chương trình phục vụ ngắt quãng.
- Cách thực hiện lệnh ngắt quãng chính là cách gọi xa gián tiếp qua bộ nhớ 32 bit.
- Số ngắt 1 byte *immed8* cung cấp trong lệnh chính là số thứ tự của chương trình con phục vụ ngắt quãng. Nhờ vậy nên mặc dù lệnh ngắt quãng là lệnh gọi xa nhưng lại rất ngắn.
- Bảng nhảy trong trường hợp này được gọi là bảng vector ngắt quãng. Vị trí của vector ngắt quãng được xác định bằng cách lấy số ngắt nhân 4. Kết quả này có thể xem là địa chỉ vật lý cũng được hoặc là địa chỉ offset lấy theo segment 0000 cũng được.
- Điểm khác biệt giữa lệnh ngắt quãng và lệnh gọi xa là thao tác cất thanh ghi trạng thái (cờ) vào chồng PUSHF. Chính vì thế nên chương trình con phục vụ ngắt quãng phải được kết thúc bằng một lệnh trở về khác là IRET.
- Các chương trình con phục vụ ngắt quãng thường được dùng cho các chương trình hệ thống (hệ điều hành, chương trình giao tiếp với các thiết bị, các chương trình con sử dụng thường xuyên, ...) hơn là dùng cho chương trình của người sử dụng.
- Số ngắt cũng theo qui ước của hệ thống như sau :
  00h ÷ 07h : ngắt hệ thống.
  08h ÷ 0Fh, 70h ÷ 77h : ngắt cứng.
  Còn lại : ngắt mềm.
- Một số ngắt thông dụng :
  INT 10h : màn hình.
  INT 13h : đĩa.
  INT 14h : thông tin liên lạc.
  INT 16h : bàn phím.
  INT 17h : máy in
  INT 21h : các phục vụ của MS-DOS.
  INT 20h : kết thúc chương trình, trở về DOS.

- Mỗi chương trình con phục vụ ngắt quãng có thể thực hiện nhiều chức năng bên trong nghĩa là các phục vụ được chia nhỏ ra nữa. Ví dụ ngắt phục vụ màn hình có chức năng chọn chế độ màn hình, chức năng định vị điểm nháy (cursor), chức năng xuất ký tự ra màn hình, chức năng đồ họa, . . .
- Thông số của chương trình phục vụ ngắt quãng thường được truyền thông qua các thanh ghi đầu vào (input) và k?t qu? thi hành chương trình con sẽ giữ trong các thanh ghi đầu ra (output).
- *Ví dụ :* dùng ngắt 17h, chức năng 0 để xuất ký tự ra máy in.

Input : AH = số chức năng (trong trường hợp này là 0).
AL = ký tự cần in (mã ASCII).
DX = số thứ tự máy in (0=LPT1, 1=LPT2, . . .)

Output : AL = trạng thái

Bit	***Ý nghĩa (nếu = 1)***
0	Quá thời gian đợi máy in.
1,2	Không dùng.
3	Lỗi xuất nhập.
4	Máy in đang được chọn.
5	Hết giấy.
6	Máy in đã nhận ký tự.
7	Máy in không bận.

- Như vậy để in ký tự 'A' ra máy in ta viết đoạn chương trình sau :

```
MOV  AH,0          ; nạp số chức năng.
MOV  AL,041h       ; mã ASCII của ký tự 'A'.
MOV  DX,0          ; nạp số thứ tự máy in LPT1.
INT  17h           ; gọi ngắt 17h để in.
MOV  [3000h],AL    ; cất trạng thái máy in.
```

6.8 Lệnh INTO :
- Dạng lệnh : INTO
- Giải thích : ***PUSHF***
  PUSH CS
  PUSH IP
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
		×	×				

- Ngắt quãng nếu tràn (OF = 1).

6.9 Lệnh IRET :
- Dạng lệnh : IRET
- Giải thích : ***POP IP***
  POP CS
  POPF
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
×	×	×	×	×	×	×	×

- Trở về từ chương trình phục vụ ngắt quãng.

7. Điều khiển bộ xử lý :

7.1 Lệnh CLC :
- Dạng lệnh : CLC
- Giải thích : ***CF ← 0***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
							0

- Xóa cờ nhớ về 0.

7.2 Lệnh STC :
- Dạng lệnh : STC
- Giải thích : ***CF ← 1***

- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
							1

- Lập cờ nhớ lên 1.

7.3 Lệnh CMC :
- Dạng lệnh : CMC
- Giải thích : ***CF ← bù 1 của CF***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
							×

- Lấy bù cờ nhớ.

7.4 Lệnh NOP :
- Dạng lệnh : NOP
- Giải thích : ***không làm gì cả***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
							0

- Không làm gì cả. Dùng để tạo ra các khoảng làm trễ ngắn.

7.5 Lệnh CLD :
- Dạng lệnh : CLD
- Giải thích : ***DF ← 0***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
	0						

- Xóa cờ định hướng về 0. Xử lý tăng địa chỉ trong các lệnh xử lý chuỗi.

7.6 Lệnh STD :
- Dạng lệnh : STD
- Giải thích : ***DF ← 1***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
	1						

- Lập cờ định hướng lên 1. Xử lý giảm địa chỉ trong các lệnh xử lý chuỗi.

7.7 Lệnh CLI :
- Dạng lệnh : CLI
- Giải thích : ***IF ← 0***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
		0					

- Xóa cờ ngắt quãng về 0. Cấm ngắt quãng cứng.

7.8 Lệnh STI :
- Dạng lệnh : STI
- Giải thích : ***IF ← 1***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF
		1					

- Lập cờ ngắt quãng lên 1. Cho phép ngắt quãng cứng.

7.9 Lệnh HLT :

- Dạng lệnh : HLT
- Giải thích : ***CPU vào trạng thái dừng.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Dừng CPU, chờ một ngắt quãng cứng xảy ra (INTR hay NMI).

7.10 Lệnh WAIT :

- Dạng lệnh : WAIT
- Giải thích : ***CPU vào trạng thái đợi.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- CPU vào trạng thái đợi cho đến khi ngỏ TEST tác động.

7.11 Tiếp đầu lệnh LOCK :

- Dạng lệnh : LOCK *lệnh*
- Giải thích : ***Khóa các tuyến trong khi thi hành lệnh theo sau.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Khóa các tuyến khi thi hành lệnh theo sau. Không cho phép các vi xử lý khác yêu cầu tuyến (chẳng hạn DMA).

7.12 Lệnh ESC :

- Dạng lệnh : ESC *immed,reg*
  ESC *immed,mem*
- Giải thích : ***đưa lệnh ra tuyến dữ liệu.***
- Tác động cờ :

OF	DF	IF	SF	ZF	AF	PF	CF

- Phát ra một lệnh cho vi mạch đồng xử lý 8087.
- *Ví dụ :* ESC 6,AL
  ESC 4,[2000h]