Khoa ĐIỆN TỬ - VIỄN THÔNG

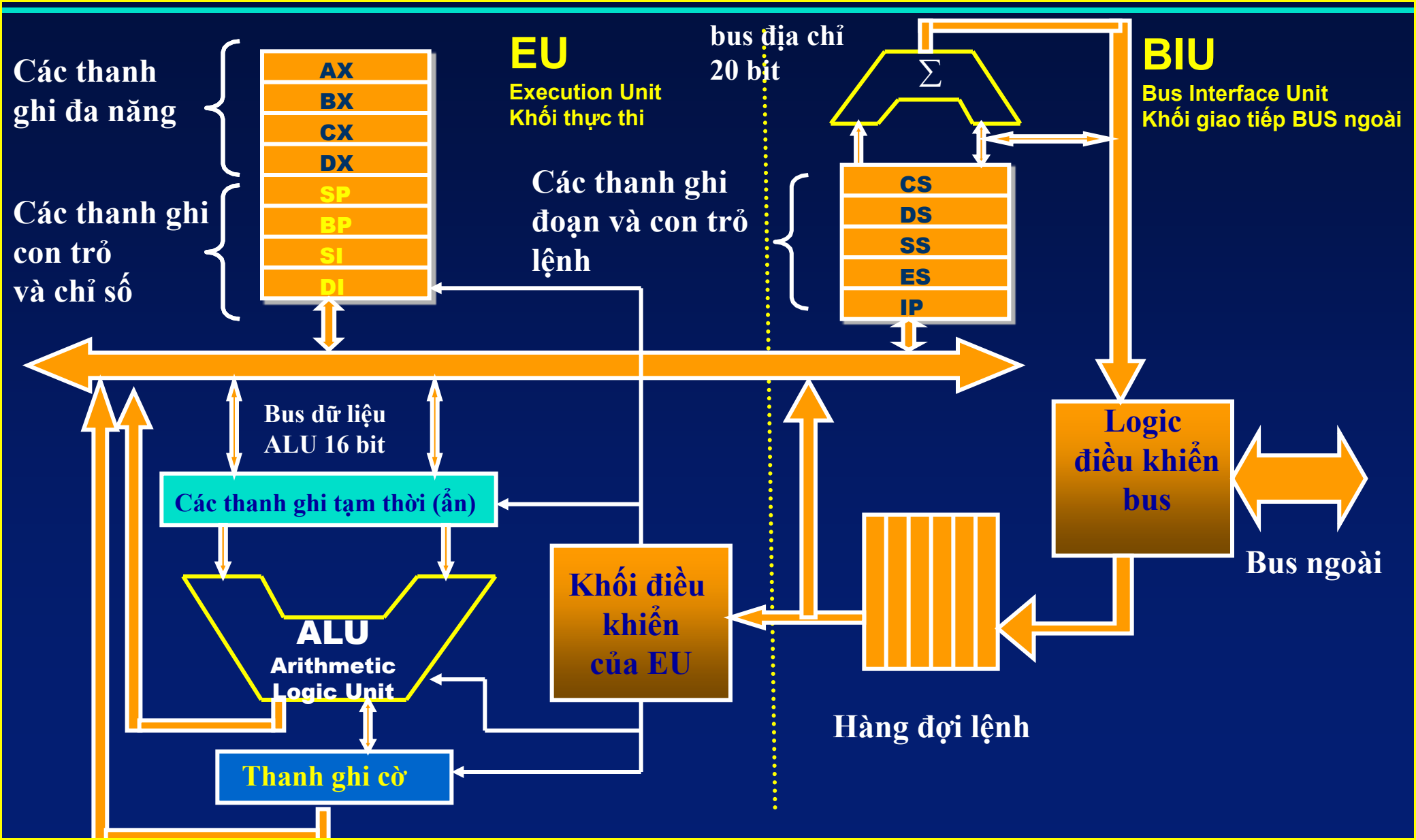
Bộ vi xử lý Intel 8088/8086

- Cấu trúc bên trong
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8086
- Cách mã hoá lệnh của 8086
- Mô tả tập lệnh của 8086
- số bộ vi xử lý khác
- Cách quản lý bộ nhớ ở chế độ bảo vệ ở các máy tính từ 80286

Bộ vi xử lý Intel 8088/8086

- Cấu trúc bên trong
  - Sơ đồ khối
  - Các thanh ghi đa năng
  - Các thanh ghi đoạn
  - Các thanh ghi con trỏ và chỉ số
  - Thanh ghi cờ
  - Hàng đợi lệnh
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8086
- Cách mã hoá lệnh của 8086
- Mô tả tập lệnh của 8086
- số bộ vi xử lý khác
- Cách quản lý bộ nhớ ở chế độ bảo vệ ở các máy tính từ 80286

Sơ đồ khối 8088/8086



Các thanh ghi đa năng của 8088/8086

	8 bit cao	8 bit thấp
AX	AH	AL
BX	BH	BL
CX	CH	CL
DX	DH	DL

- Thanh ghi chứa AX (accumulator): chứa kết quả của các phép tính. Kết quả 8 bit được chứa trong AL
- Thanh ghi cơ sở BX (base): chứa địa chỉ cơ sở, ví dụ của bảng dùng trong lệnh XLAT (Translate)
- Thanh ghi đếm CX (count): dùng để chứa số lần lặp trong các lệnh lặp (Loop). CL được dùng để chứa số lần dịch hoặc quay trong các lệnh dịch và quay thanh ghi
- Thanh ghi dữ liệu DX (data): cùng AX chứa dữ liệu trong các phép tính nhân chia số 16 bit. DX còn được dùng để chứa địa chỉ cổng trong các lệnh vào ra dữ liệu trực tiếp (IN/OUT)

Các thanh ghi đoạn

- Tổ chức của bộ nhớ 1 Mbytes
  - ❑ Vấn đề: Sử dụng 2 thanh ghi 16bit để xác định địa chỉ 20bit (1M)
  - ❑ Đoạn bộ nhớ (segment)
    - ⇨ 2^{16} bytes =64 KB
    - ⇨ Đoạn 1: địa chỉ 0000
    - ⇨ Đoạn 2: địa chỉ 0001
    - ⇨ Đoạn cuối cùng: FFFF
  - ❑ Ô nhớ trong đoạn:
    - ⇨ địa chỉ lệch: offset
    - ⇨ Ô 1: offset: 0000
    - ⇨ Ô cuối cùng: offset: FFFF
  - ❑ Địa chỉ vật lý:
    - ⇨ Segment : offset

Địa chỉ vật lý=Segment*16 + offset

Chế độ thực (real mode)

Các thanh ghi đoạn

- Ví dụ: Địa chỉ vật lý 12345H

Địa chỉ đoạn	Địa chỉ lệch
1000 H	?
1200 H	?
1004 H	?
0300 H	?

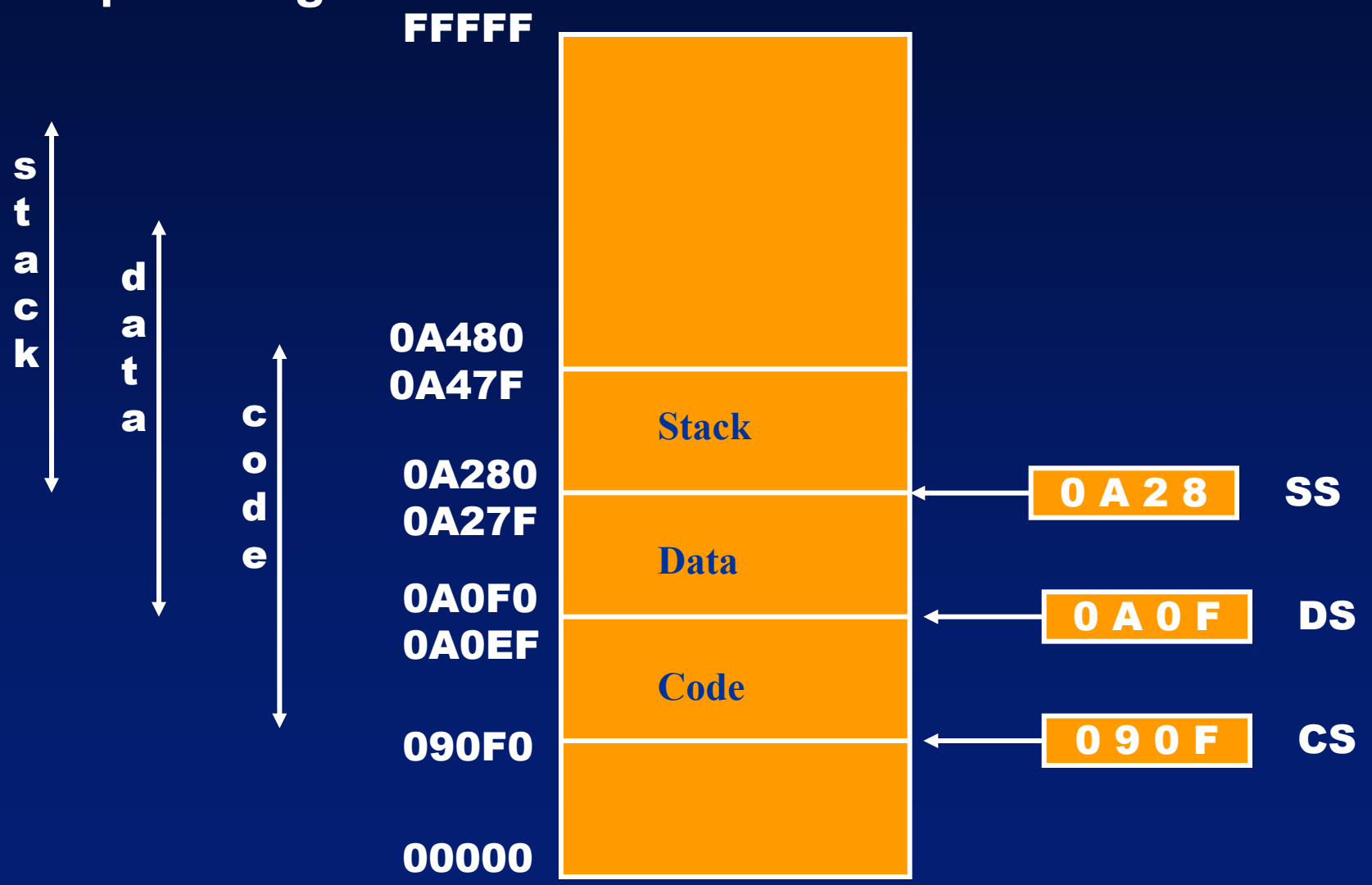
- Ví dụ: Cho địa chỉ đầu của đoạn: 49000 H, xác định địa chỉ cuối

Các thanh ghi đoạn

- Các thanh ghi đoạn: chứa địa chỉ đoạn

Các thanh ghi đoạn

- Các đoạn chồng nhau



Các thanh ghi con trỏ và chỉ số

- Chứa địa chỉ lệch (offset)
  - Con trỏ lệnh IP (instruction pointer): chứa địa chỉ lệnh tiếp theo trong đoạn mã lệnh CS.
    - ⇨ CS:IP
  - Con trỏ cơ sở BP (Base Pointer): chứa địa chỉ của dữ liệu trong đoạn ngăn xếp SS hoặc các đoạn khác
    - ⇨ SS:BP
  - Con trỏ ngăn xếp SP (Stack Pointer): chứa địa chỉ hiện thời của đỉnh ngăn xếp
    - ⇨ SS:SP
  - Chỉ số nguồn SI (Source Index): chứa địa chỉ dữ liệu nguồn trong đoạn dữ liệu DS trong các lệnh chuỗi
    - ⇨ DS:SI
  - Chỉ số đích (Destination Index): chứa địa chỉ dữ liệu đích trong đoạn dữ liệu DS trong các lệnh chuỗi
    - ⇨ DS:DI
  - SI và DI có thể được sử dụng như thanh ghi đa năng

Các thanh ghi con trỏ và chỉ số

- Thanh ghi đoạn và thanh ghi lệch ngầm định

Segment	Offset	Chú thích
CS	IP	Địa chỉ lệnh
SS	SP hoặc BP	Địa chỉ ngăn xếp
DS	BX, DI, SI, số 8 bit hoặc số 16 bit	Địa chỉ dữ liệu
ES	DI	Địa chỉ chuỗi đích

Thanh ghi cờ (Flag Register)

- 9 bit được sử dụng, 6 cờ trạng thái:
  - C hoặc CF (carry flag)): CF=1 khi có nhớ hoặc mượn từ MSB
  - P hoặc PF (parity flag): PF=1 (0) khi tổng số bít 1 trong kết quả là chẵn (lẻ)
  - A hoặc AF (auxilary carry flag): cờ nhớ phụ, AF=1 khi có nhớ hoặc mượn từ một số BCD thấp sang BCD cao
  - Z hoặc ZF (zero flag): ZF=1 khi kết quả bằng 0
  - S hoặc SF (Sign flag): SF=1 khi kết quả âm
  - O hoặc OF (Overflow flag): cờ tràn OF=1 khi kết quả là một số vượt ra ngoài giới hạn biểu diễn của nó trong khi thực hiện phép toán cộng trừ số có dấu

Thanh ghi cờ (Flag Register)

- 3 cờ điều khiển
  - T hoặc TF (trap flag)): cờ bẫy, TF=1 -> CPU làm việc ở chế độ chạy từng lệnh
  - I hoặc IF (Interrupt enable flag): cờ cho phép ngắt, IF=1 thì CPU sẽ cho phép các yêu cầu ngắt (ngắt che được) được tác động (Các lệnh: STI, CLI)
  - D hoặc DF (direction flag): cờ hướng, DF=1 khi CPU làm việc với chuỗi ký tự theo thứ tự từ phải sang trái (lệnh STD, CLD)

Thanh ghi cờ (Flag Register)

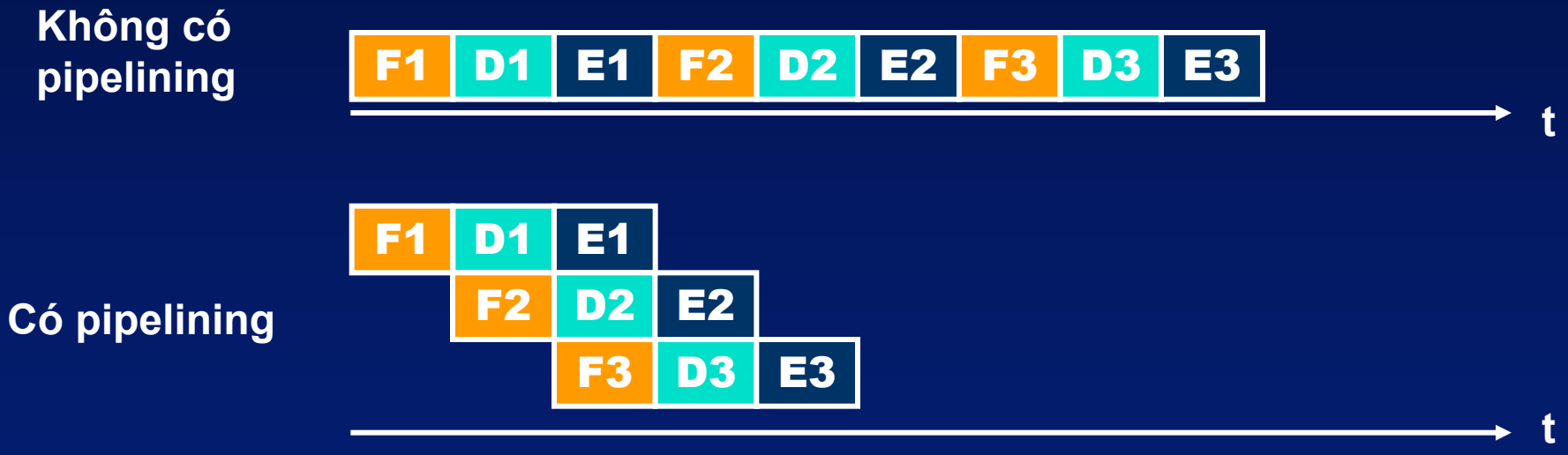
- Ví dụ:

$$\begin{array}{r} 80h \\ + \\ 80h \\ \hline 100h \end{array}$$

- ❑ SF=0 vì msb trong kết quả =0
- ❑ PF=1 vì có 0 bít của tổng bằng 1
- ❑ ZF=1 vì kết qủa thu được là 0
- ❑ CF=1 vì có nhớ từ bít msb trong phép cộng
- ❑ OF=1 vì có tràn trong phép cộng 2 số âm
  - ⇨ Cộng 2 số âm thu được kết quả dương

Hàng đợi lệnh

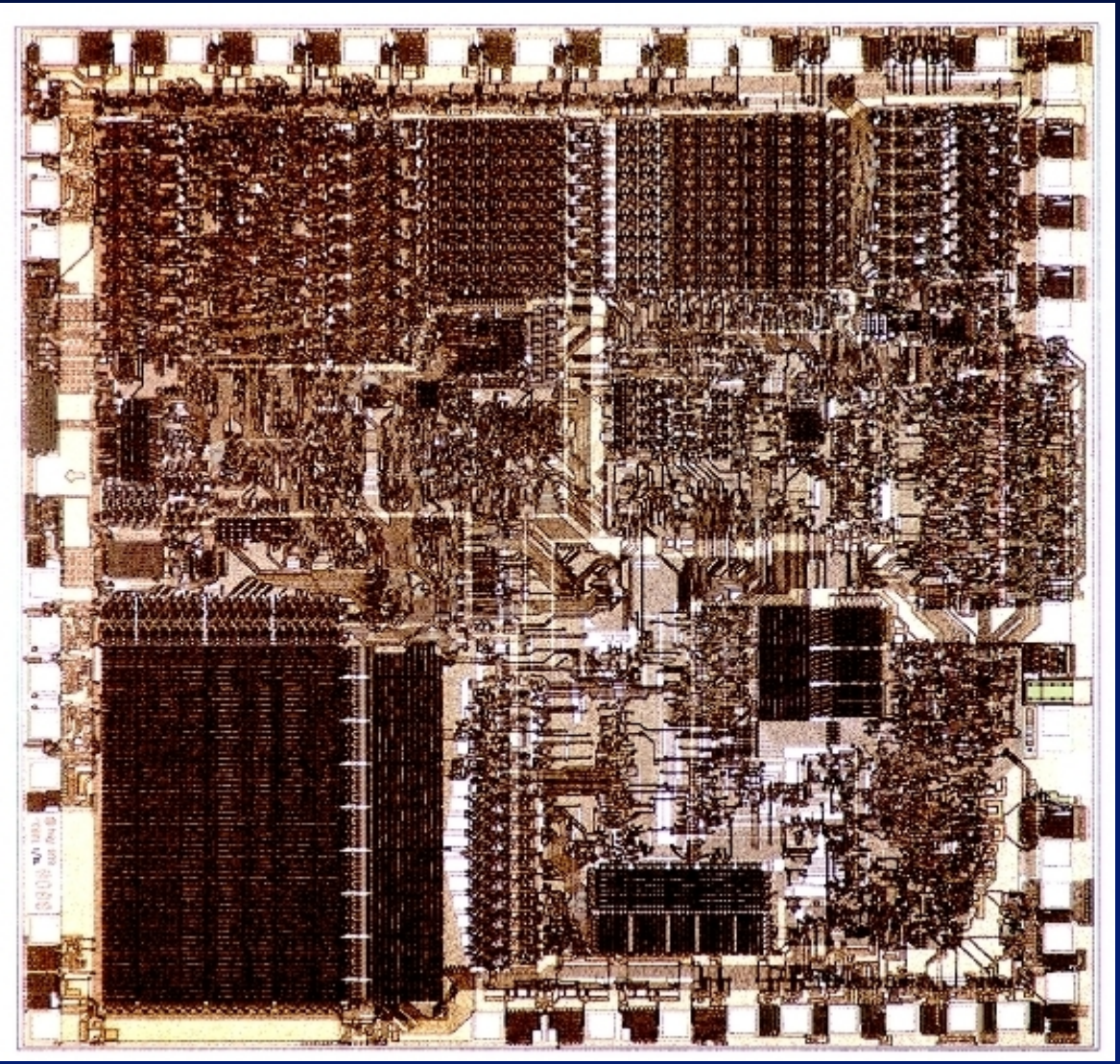
- 4 bytes đối với 8088 và 6 bytes đối với 8086
- Xử lý pipeline



Bộ vi xử lý Intel 8088/8086

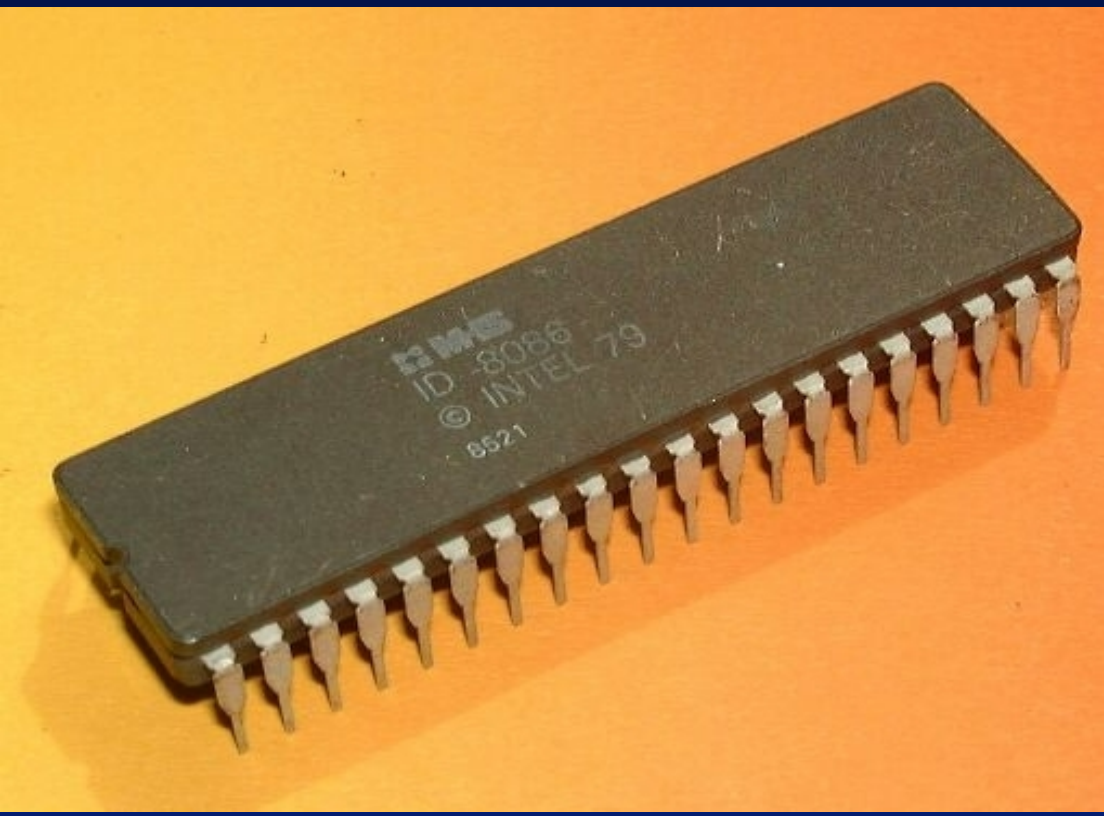
- Cấu trúc bên trong
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8086
- Cách mã hoá lệnh của 8086
- Mô tả tập lệnh của 8086
- số bộ vi xử lý khác
- Cách quản lý bộ nhớ ở chế độ bảo vệ ở các máy tính từ 80286

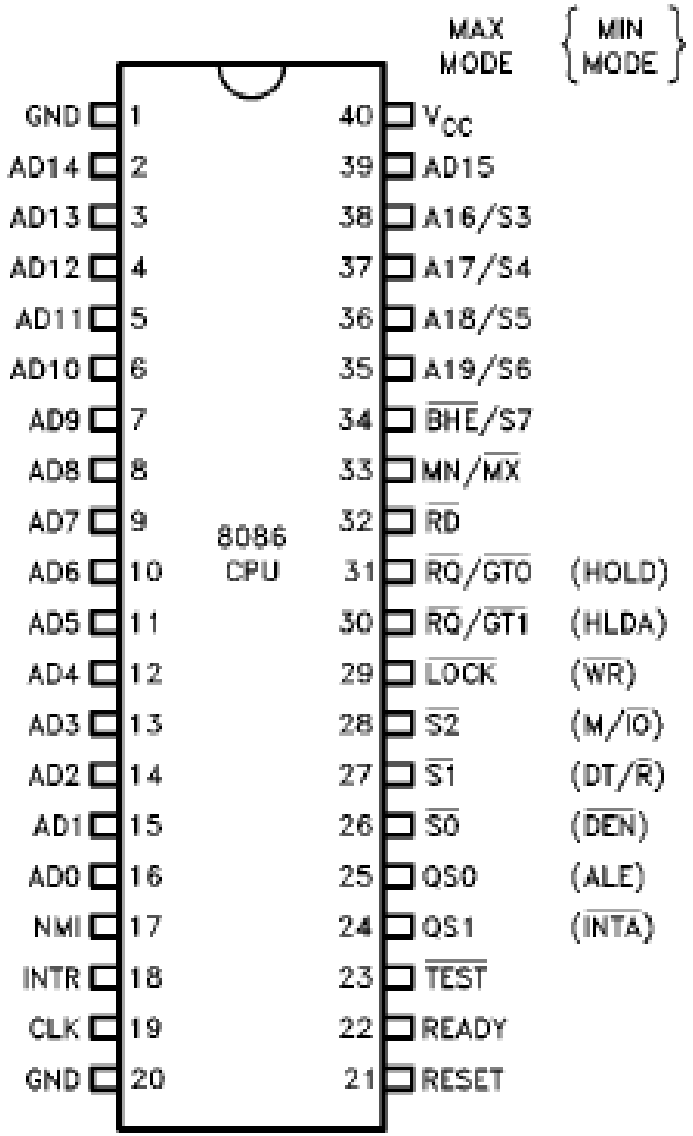
Sơ đồ chân Intel 8088



- 16-bit processor
- introduced in 1979
- 3 µm, 5 to 8 MHz, 29 KTOR, 0.33 to 0.66 MIPS

Sơ đồ chân Intel 8086



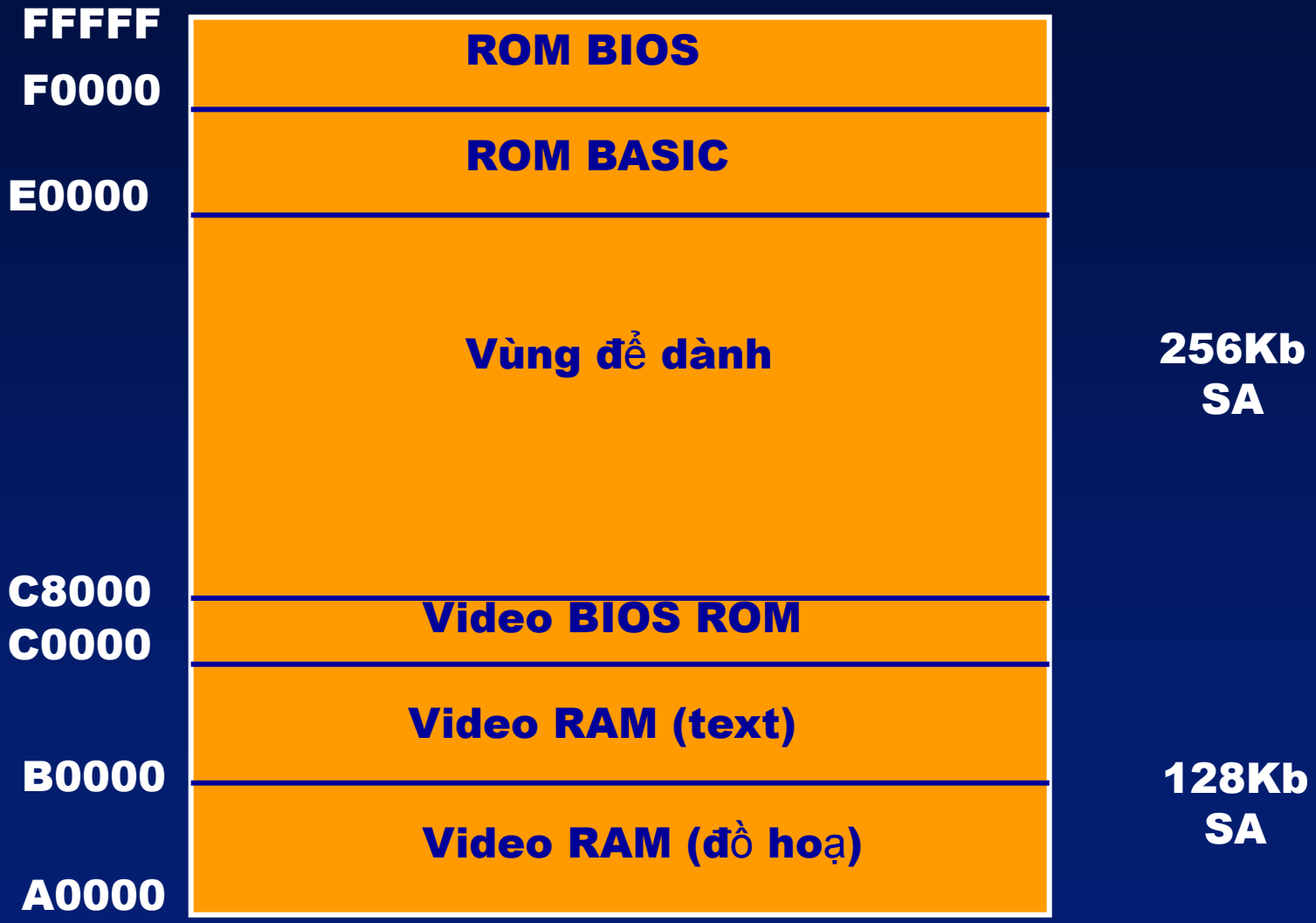


Bộ vi xử lý Intel 8088/8086

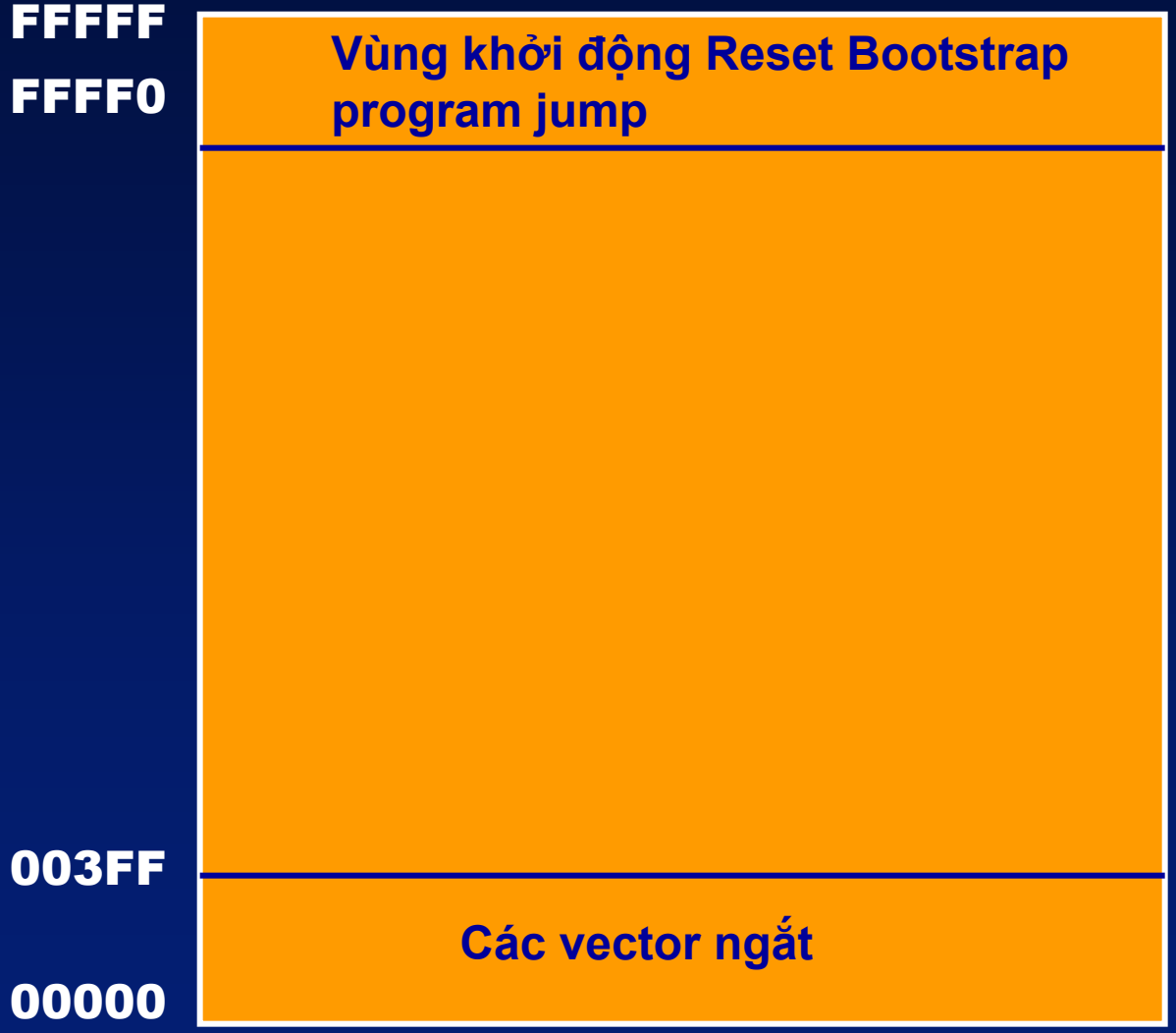
- Cấu trúc bên trong
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8086
- Cách mã hoá lệnh của 8086
- Mô tả tập lệnh của 8086
- số bộ vi xử lý khác
- Cách quản lý bộ nhớ ở chế độ bảo vệ ở các máy tính từ 80286

Bản đồ bộ nhớ máy tính IBM-PC
Vùng nhớ chương trình

Bản đồ bộ nhớ máy tính IBM-PC
Vùng nhớ hệ thống



Bản đồ bộ nhớ máy tính IBM-PC
Vùng nhớ dành riêng của 8088/8086



Khoa ĐIỆN TỬ - VIỄN THÔNG

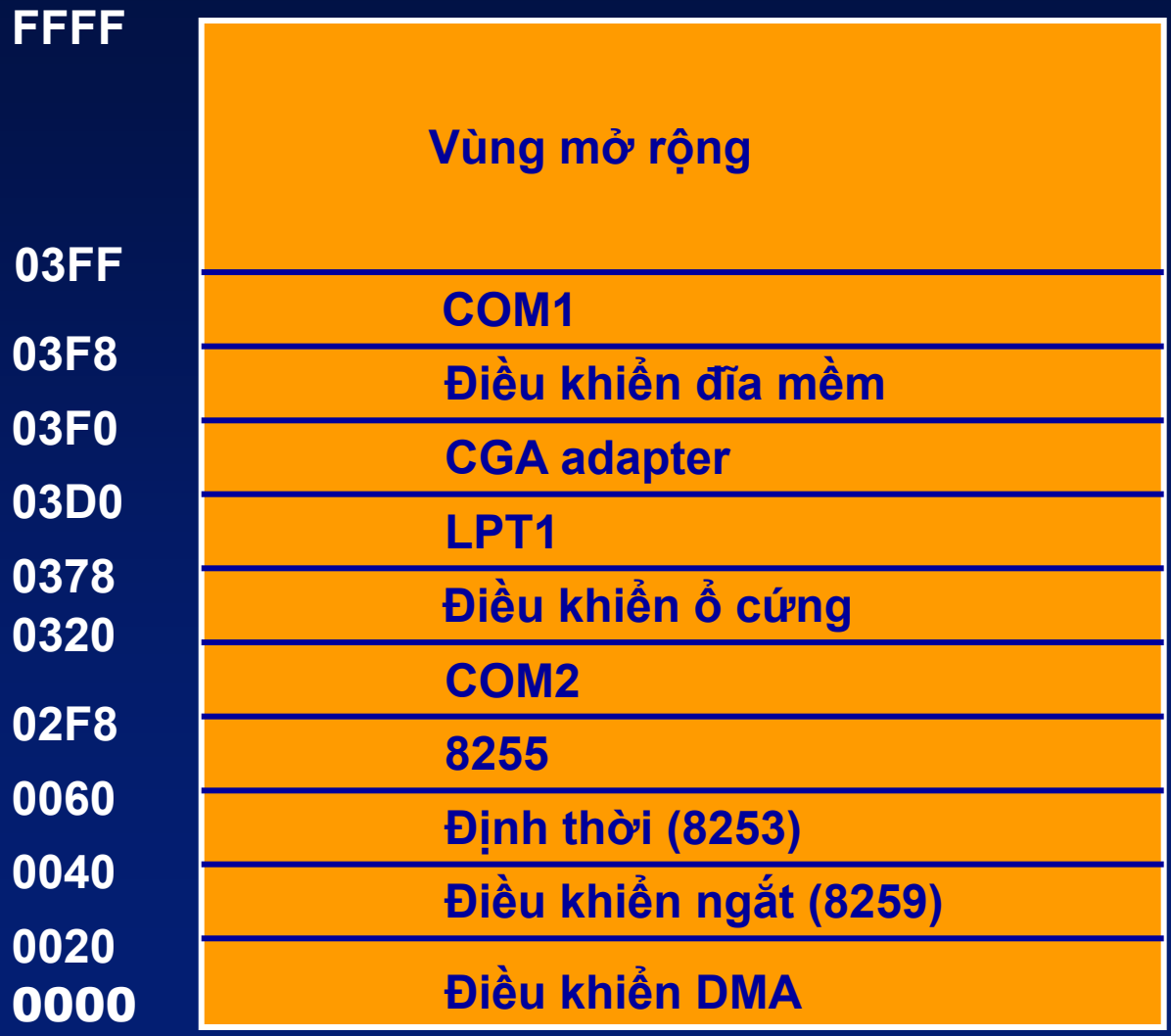
Trình tự khởi động

- Khi bật nguồn hoặc nhấn Reset
  - CS=FFFFh và IP=0000 => địa chỉ FFFF0 chứa chỉ thị chuyển điều khiển đến điểm khởi đầu của các chương trình BIOS
  - Các chương trình BIOS kiểm tra hệ thống và bộ nhớ
  - Các chương trình BIOS khởi tạo bảng vector ngắt và vùng dữ liệu BIOS
  - BIOS nạp chương trình khởi động (boot program) từ đĩa vào bộ nhớ
  - Chương trình khởi động nạp hệ điều hành từ đĩa vào bộ nhớ
  - Hệ điều hành nạp các chương trình ứng dụng

Bản đồ bộ nhớ của máy tính IBM PC

Bản đồ bộ nhớ máy tính IBM-PC
Các cổng vào ra

- Địa chỉ: 0000H –FFFFH, M/$\overline{IO}$ =0



Bộ vi xử lý Intel 8088/8086

- Cấu trúc bên trong
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8088/8086
  - Chế độ địa chỉ thanh ghi
  - Chế độ địa chỉ tức thì
  - Chế độ địa chỉ trực tiếp
  - Chế độ địa chỉ gián tiếp qua thanh ghi
  - Chế độ địa chỉ tương đối cơ sở
  - Chế độ địa chỉ tương đối chỉ số
  - Chế độ địa chỉ tương đối chỉ số cơ sở
- Cách mã hoá lệnh của 8088/8086
- Mô tả tập lệnh của 8088/8086
- Sơ đồ cấu trúc của 1 số bộ vi xử lý khác

Chế độ địa chỉ thanh ghi (Register Addressing Mode)

- Dùng các thanh ghi như là các toán hạng
- Tốc độ thực hiện lệnh cao

- Ví dụ:
  - MOV BX, DX ; Copy nội dung DX vào BX, nội dung DX được giữ nguyên
  - MOV AL, BL ; Copy nội dung BL vào AL
  - MOV AL, BX ; không hợp lệ vì các thanh ghi có kích thước khác nhau
  - MOV ES, DS ; không hợp lệ (segment to segment)
  - MOV CS, AX ; không hợp lệ vì CS không được dùng làm thanh ghi đích
  - ADD AL, DL ; Cộng nội dung AL và DL rồi đưa vào AL

Chế độ địa chỉ tức thì (Immediate Addressing Mode)

- Toán hạng đích là thanh ghi hoặc ô nhớ
- Toán hạng nguồn là hằng số
- Dùng để nạp hằng số vào thanh ghi (trừ thanh ghi đoạn và thanh cờ) hoặc vào ô nhớ trong đoạn dữ liệu DS
- Ví dụ:
  - MOV BL, 44 ; Copy số thập phân 44 vào thanh ghi BL
  - MOV AX, 44H ; Copy 0044H vào thanh ghi AX
  - MOV AL, 'A' ; Copy mã ASCII của A vào thanh ghi AL
  - MOV DS, 0FF0H ; không hợp lệ
  - MOV AX, 0FF0H ;
  - MOV DS, AX ;

  - MOV [BX], 10 ; copy số thập phân 10 vào ô nhớ DS:BX

Chế độ địa chỉ trực tiếp (Direct Addressing Mode)

- Một toán hạng là địa chỉ ô nhớ chứa dữ liệu
- Toán hạng kia chỉ có thể là thanh ghi

- Ví dụ:
  - MOV AL, [1234H] ; Copy nội dung ô nhớ có địa chỉ DS:1234 vào AL
  - MOV [4320H], CX ; Copy nội dung của CX vào 2 ô nhớ liên tiếp DS: 4320 và DS: 4321

Chế độ địa chỉ gián tiếp qua thanh ghi (Register indirect Addressing Mode)

- Một toán hạng là thanh ghi chứa địa chỉ của 1 ô nhớ dữ liệu
- Toán hạng kia chỉ có thể là thanh ghi

- Ví dụ:
  - MOV AL, [BX] ; Copy nội dung ô nhớ có địa chỉ DS:BX vào AL
  - MOV [SI], CL ; Copy nội dung của CL vào ô nhớ có địa chỉ DS:SI
  - MOV [DI], AX ; copy nội dung của AX vào 2 ô nhớ liên tiếp DS: DI và DS: (DI +1)

Chế độ địa chỉ tương đối cơ sở (Based relative Addressing Mode)

- Một toán hạng là thanh ghi cơ sở BX, BP và các hằng số biểu diễn giá trị dịch chuyển
- Toán hạng kia chỉ có thể là thanh ghi

- Ví dụ:
  - MOV CX, [BX]+10 ; Copy nội dung 2 ô nhớ liên tiếp có địa chỉ DS:BX+10 và DS:BX+11 vào CX
  - MOV CX, [BX+10] ; Cách viết khác của lệnh trên
  - MOV AL, [BP]+5 ; copy nội dung của ô nhớ SS:BP+5 vào thanh ghi AL

Chế độ địa chỉ tương đối chỉ số (Indexed relative Addressing Mode)

- Một toán hạng là thanh ghi chỉ số SI, DI và các hằng số biểu diễn giá trị dịch chuyển
- Toán hạng kia chỉ có thể là thanh ghi

- Ví dụ:
  - MOV AX, [SI]+10 ; Copy nội dung 2 ô nhớ liên tiếp có địa chỉ DS:SI+10 và DS:SI+11 vào AX
  - MOV AX, [SI+10] ; Cách viết khác của lệnh trên
  - MOV AL, [DI]+5 ; copy nội dung của ô nhớ DS:DI+5 vào thanh ghi AL

Chế độ địa chỉ tương đối chỉ số cơ sở (Based Indexed relative Addressing Mode)

- Kết hợp của 2 chế độ địa chỉ trước

- Ví dụ:
  - MOV AX, [BX] [SI]+8 ; Copy nội dung 2 ô nhớ liên tiếp có địa chỉ DS:BX+SI+8 và DS:BX+SI+9 vào AX
  - MOV AX, [BX+SI+8] ; Cách viết khác của lệnh trên
  - MOV CL, [BP+DI+5] ; copy nội dung của ô nhớ SS:BP+DI+5 vào thanh ghi CL

Tóm tắt các chế độ địa chỉ

Chế độ địa chỉ	Toán hạng	Thanh ghi đoạn ngầm định
Thanh ghi	Thanh ghi	
Tức thì	Dữ liệu	
Trực tiếp	[offset]	DS
Gián tiếp qua thanh ghi	[BX] [SI] [DI]	DS DS DS
Tương đối cơ sở	[BX] + dịch chuyển [BP] + dịch chuyển	DS SS
Tương đối chỉ số	[DI] + dịch chuyển [SI] + dịch chuyển	DS DS
Tương đối chỉ số cơ sở	[BX] + [DI]+ dịch chuyển [BX] + [SI]+ dịch chuyển [BP] + [DI]+ dịch chuyển [BP] + [SI]+ dịch chuyển	DS DS SS SS

Bỏ chế độ ngầm định thanh ghi đoạn (Segment override)

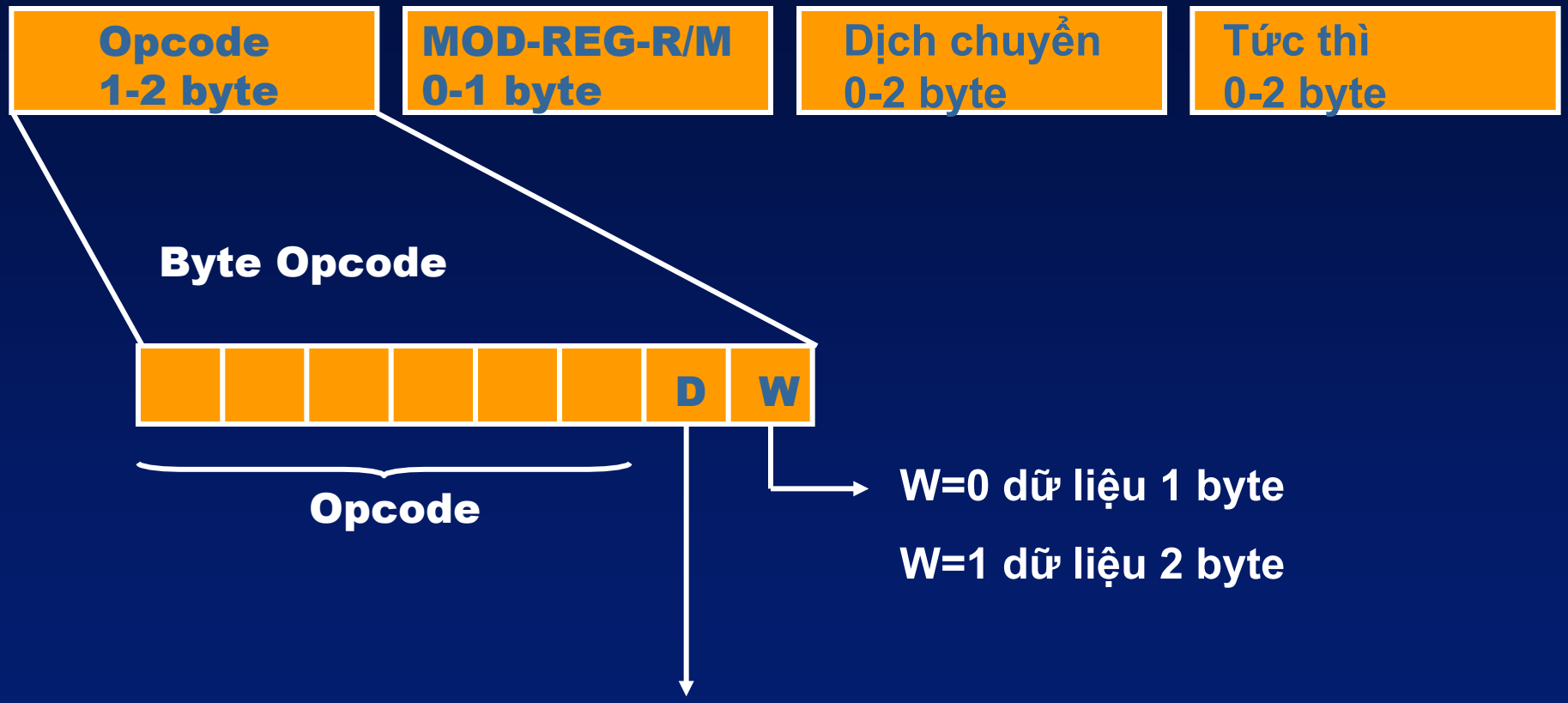
- Ví dụ:
  - MOV AL, [BX]; Copy nội dung ô nhớ có địa chỉ DS:BX vào AL
  - MOV AL, ES:[BX] ; Copy nội dung ô nhớ có địa chỉ ES:BX vào AL

Bộ vi xử lý Intel 8088/8086

- Cấu trúc bên trong
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8086
- **Cách mã hoá lệnh của 8086**
- Mô tả tập lệnh của 8086
- số bộ vi xử lý khác
- Cách quản lý bộ nhớ ở chế độ bảo vệ ở các máy tính từ 80286

Cách mã hoá lệnh của 8086

- Một lệnh có độ dài từ 1 đến 6 byte



Cách mã hoá lệnh của 8086

MOD REG R/M

MOD <> 11

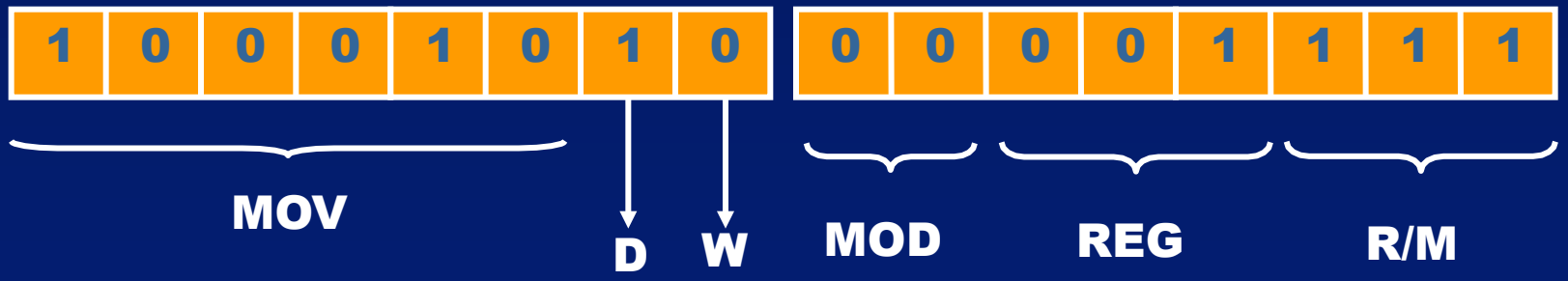
00	không có dịch chuyển
01	dịch chuyển 8 bit
10	dịch chuyển 16 bit
11	R/M là thanh ghi

Thanh ghi		Mã
W=1	W=0	
AX	AL	000
BX	BL	011
CX	CL	001
DX	DL	010
SP	AH	100
DI	BH	111
BP	CH	101
SI	DH	110

Mã	Chế độ địa chỉ
000	DS:[BX+SI]
001	DS:[BX+DI]
010	SS:[BP+SI]
011	SS:[BP+DI]
100	DS:[SI]
101	DS:[DI]
110	SS:[BP]*
111	DS:[BX]

Cách mã hoá lệnh của 8086

- Ví dụ: chuyển lệnh MOV CL, [BX] sang mã máy
  - opcode MOV: 100010
  - Dữ liệu là 1 byte: W=0
  - Chuyển tới thanh ghi: D=1
  - Không có dịch chuyển: MOD=00
  - [BX] nên R/M=111
  - CL nên REG=00



Ví dụ 2: chuyển lệnh MOV [SI+0F3H], CL sang mã máy

Cách mã hoá lệnh của 8086

Ví dụ 2: chuyển lệnh MOV [SI+0F3H], CL sang mã máy

Ví dụ 3: chuyển lệnh MOV BP,SP sang mã máy

Ví dụ 4: chuyển lệnh MOV DL,[DI+1] sang mã máy

Ví dụ 5: chuyển lệnh MOV BP,[DI+1000H] sang mã máy

Ví dụ 6: chuyển lệnh MOV [1000H],DL sang mã máy

Ví dụ 7: chuyển lệnh MOV [BP],DL sang mã máy

Bộ vi xử lý Intel 8088/8086

- Cấu trúc bên trong
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8086
- Cách mã hoá lệnh của 8086
- **Mô tả tập lệnh của 8086**
  - Các lệnh di chuyển (thay đổi) dữ liệu
  - Các lệnh số học và logic
  - Các lệnh điều khiển chương trình
  - Các lệnh xử lý chuỗi
  - Các lệnh điều khiển VXL
- Một số bộ vi xử lý khác
- Cách quản lý bộ nhớ ở chế độ bảo vệ ở các máy tính từ 80286

Các lệnh di chuyển dữ liệu
Làm việc với bộ nhớ và thanh ghi

MOV, XCHG, POP, PUSH, POPF, PUSHF, IN, OUT, LDS, LEA, LES, LAHF, SAHF,XLAT

- **MOV**
  - Dùng để chuyển giữa các thanh ghi, giữa 1 thanh ghi và 1 ô nhớ hoặc chuyển 1 số vào thanh ghi hoặc ô nhớ
  - Cú pháp: **MOV Đích, nguồn**
  - Lệnh này không tác động đến cờ
  - Ví dụ:

```
MOV AX, BX
MOV AH, 'A'
MOV AL, [1234H]
MOV AH, 1234H
MOV [1234H], AL
MOV [1234H], 1234H
MOV 1234H, [1234H]
MOV 1234H,AX
MOV 1234H, AL
MOV [SI], 1234H
MOV [DI] + 10, 1234H
MOV [SI], [DI] + 10
MOV 1234H, [SI] + [BX] + 10
MOV DS,12H
MOV ES, DS
MOV 12H, DS
```

Các lệnh di chuyển dữ liệu
Làm việc với bộ nhớ và thanh ghi

- Khả năng kết hợp toán hạng của lệnh MOV

Đích / Nguồn	Thanh ghi đa năng	Thanh ghi đoạn	ô nhớ	Hằng số
Thanh ghi đa năng	YES	YES	YES	NO
Thanh ghi đoạn	YES	NO	YES	NO
Ô nhớ	YES	YES	NO	NO
Hằng số	YES	NO	YES	NO

Các lệnh di chuyển dữ liệu
Làm việc với bộ nhớ và thanh ghi

- Lệnh XCHG (Exchange 2 operands)
  - ❑ Dùng để hoán chuyển nội dung giữa hai thanh ghi, giữa 1 thanh ghi và 1 ô nhớ
  - ❑ Cú pháp: XCHG Đích, nguồn
  - ❑ Giới hạn: toán hạng không được là thanh ghi đoạn
  - ❑ Lệnh này không tác động đến cờ
  - ❑ Ví dụ: Lệnh nào đúng ?

```
XCHG AX, BX
XCHG AL, [SI]
XCHG AX, [BX]
XCHG [SI], BH
XCHG [BX], AX
XCHG AX, DI
XCHG DI, SI
XCHG [SI], [BX]
XCHG DI, [BX]
XCHG DI, AX
XCHG DS, SI
XCHG [SI], DS
```

Các lệnh di chuyển dữ liệu
Làm việc với bộ nhớ và thanh ghi

- Lệnh LEA- Load Effective Address
  - Nạp địa chỉ hiệu dụng vào thanh ghi
  - Thực hiện: Đích=Địa chỉ lệch (hoặc hiệu dụng) của nguồn
  - Ví dụ: điều gì sẽ xảy ra khi thực hiện các câu lệnh sau đây

```
LEA     DX,[DI]
LEA     CX,[BX][DI]
LEA     AL, [BX + DI + 10]
LEA     DS, [DI]
LEA     DI, [DI]
LEA     BX, [DI]
LEA     [DI], [DI]
LEA     [DI], [SI]
```

Các lệnh di chuyển dữ liệu
Làm việc với bộ nhớ và thanh ghi

- **LES-Load Register and ES with Words from Memory**
  - Nạp 4 bytes hay 2 từ từ bộ nhớ vào 1 thanh ghi bất kì trước và ES sau
  - Thực hiện: Đích1 ← [địa chỉ + 1, địa chỉ]

    ES ← [địa chỉ + 3, địa chỉ + 2]

  ví dụ:

```
LES  BX, [1000h]
LES  DS, [1000h]
LES  ES, [1000h]
LES  DI, [1000h]
LES  AL, [1000h]
LES  BX, [DI]
LES  DS, [DI]
```

Các lệnh di chuyển dữ liệu
Làm việc với bộ nhớ và thanh ghi

- **LDS-Load Register and DS with Words from Memory**
  - Nạp 4 bytes hay 2 từ từ bộ nhớ vào 1 thanh ghi bất kì trước và DS sau
  - Thực hiện: Đích1 ← [địa chỉ + 1, địa chỉ]

    DS ← [địa chỉ + 3, địa chỉ + 2]

  ví dụ:

```
LDS  BX, [1000h]
LDS  DS, [1000h]
LDS  ES, [1000h]
LDS  DI, [1000h]
LDS  AL, [1000h]
LDS  BX, [DI]
LDS  DS, [DI]
```

Các lệnh di chuyển dữ liệu
Làm việc với bộ nhớ và thanh ghi

- LAHF – Load low byte of flag register into AH
  - ❑ Nạp 8 bit thấp của thanh ghi cờ vào thanh ghi AH
  - ❑ Thực hiện: AH <- FR (low)

  Ví dụ: LAHF

- SAHF – Store AH register into low byte of Flag register
  - ❑ Cất thanh ghi AH vào 8 bit thấp của thanh ghi cờ
  - ❑ Thực hiện: FR (low) <- AH

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- Lệnh PUSH
  - ❑ Dùng để cất 1 từ từ thanh ghi hoặc ô nhớ vào đỉnh ngăn xếp
  - ❑ Cú pháp: **PUSH Nguồn**
  - ❑ Mô tả: trước tiên SP=SP-2, sau đó Nguồn => {SP}
  - ❑ Giới hạn: thanh ghi 16 bit hoặc là 1 từ nhớ
  - ❑ Lệnh này không tác động đến cờ

Ví dụ1: Xác định vị trí con trỏ SP

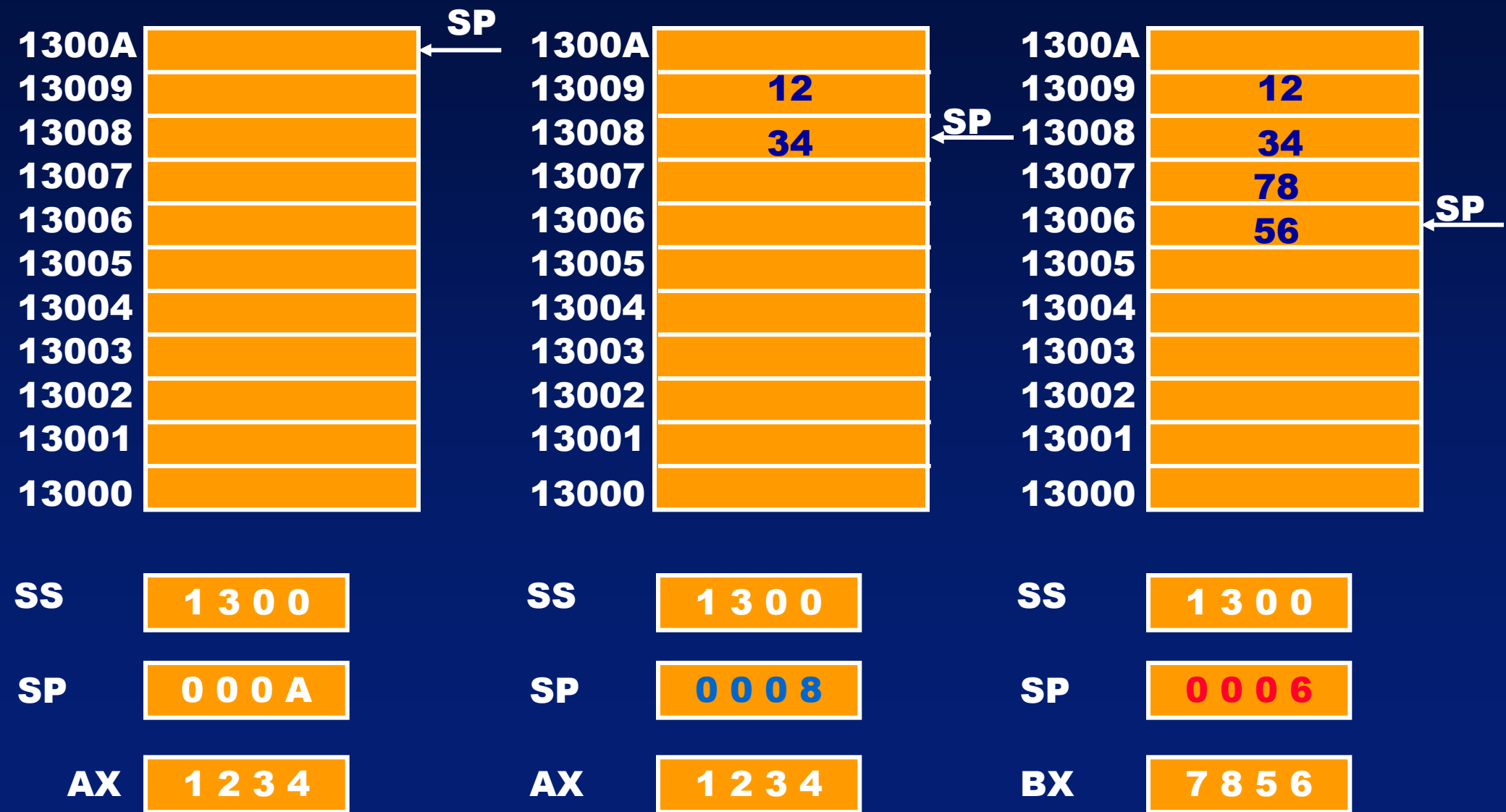
Cho SS = 1300h
SP = 000Ah
AX = 1234h
BX = 7856h

->

PUSH AX
PUSH BX

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- Ví dụ về lệnh PUSH



Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- **Lệnh PUSH**

  Ví dụ2: Xác định vị trí con trỏ SP

  Cho SS = 1300h
  SP = 000Ah
  AX = 1234h
  BX = 7856h
  DS = 1122h
  DI = 0011h
  DS : DI = 33h
  DS : DI + 1 = 44h

  ->

```
PUSH  AX
PUSH  BX
PUSH  DS
PUSH  DI
PUSH DS:DI
```

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- **Lệnh PUSH**

  Ví dụ3: Xác định vị trí con trỏ SP

  Cho SS = 1300h
  SP = 000Ah
  AX = 1234h
  BX = 7856h
  DS = 1122h
  DI = 0011h
  DS : DI = 33h
  DS : DI + 1 = 44h

  ->

  PUSH AX
  PUSH BX
  PUSH DS
  PUSH DI
  PUSH DS:DI
  PUSH DS

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- Lệnh POP
  - Dùng để lấy lại 1 từ vào thanh ghi hoặc ô nhớ từ đỉnh ngăn xếp
  - Cú pháp: POP Đích
  - Mô tả: {SP} => Đích, SP=SP+2
  - Giới hạn: thanh ghi 16 bit (trừ IP) hoặc là 1 từ nhớ
  - Lệnh này không tác động đến cờ
  - Ví dụ1:

    Cho

    SS = 1300h

    SP = 0006h

    SS:SP = 56h

    SS:SP+1 = 78h

    SS:SP + 2 = 34h

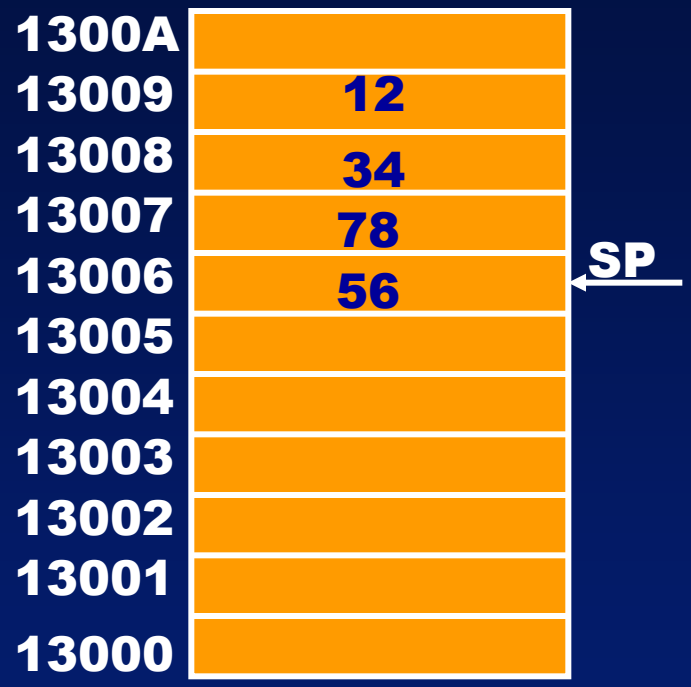
    SS:SP + 3 = 12h

    Tìm POP DX

    POP AX

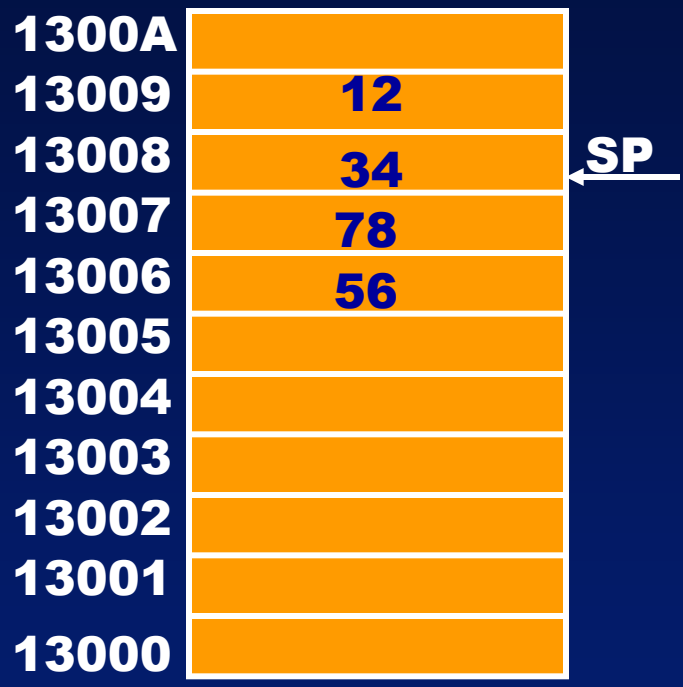
Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- Ví dụ lệnh POP DX



SS	1300
SP	0006
DX	7856

POP AX



SS	1300
SP	0008
AX	1234

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- Lệnh POP
  - ❑ Ví dụ2:

    Cho

    SS = 1300h

    SP = 0006h

    SS:SP = 56h

    SS:SP+1 = 78h

    SS:SP + 2 = 34h

    SS:SP + 3 = 12h

    Tìm POP DS

    POP SI

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- Lệnh POP
  - Ví dụ3:

    Cho

    SS = 1300h

    SP = 0006h

    SS:SP = 56h

    SS:SP+1 = 78h

    SS:SP + 2 = 34h

    SS:SP + 3 = 12h

    DS = 1122h

    DI = 0011h

    Tìm POP DS:DI

    POP DS:DI + 4

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- Lệnh POP
  - ❑ Ví dụ4:

    Cho

    SS = 1300h

    SP = 0006h

    SS:SP = 56h

    SS:SP+1 = 78h

    SS:SP + 2 = 34h

    SS:SP + 3 = 12h

    Tìm POP DL

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

- Lệnh PUSHF

❑ Cất nội dung của thanh ghi cờ vào ngăn xếp
❑ Thực hiện: trước tiên SP = SP – 2 , sau đó FR -> {SP}

ví dụ: PUSHF

- Lệnh POPF

❑ Lấy nội dung của ngăn xếp đưa vào thanh ghi cờ
❑ Thực hiện: trước tiên {SP} -> FR, sau đó SP = SP + 2

ví dụ: POPF

Các lệnh di chuyển dữ liệu
Tại sao lại cần bộ nhớ ngăn xếp

- Số lượng thanh ghi hữu hạn, nhiều chương trình (con)

```
MOV AX, 10
MOV BX, 20
CALL ChuongTrinhCon; gọi chương trình con
;AX=?, BX=?

ChuongTrinhCon:
    PUSH AX      ;cất AX vào ngăn xếp
    PUSH BX      ;cất BX vào ngăn xếp
    PUSHF
    MOV AX,0     ;Thuật toán của chương trình con, có sử dụng AX, BX
    POPF
    POP BX       ;trả lại giá trị cũ của AX, BX
    POP AX
    RET          ;Trở về chương trình chính
```

- Thuật toán sử dụng bộ nhớ LIFO

Các lệnh di chuyển dữ liệu
Làm việc với cổng

- Lệnh IN
  - Dùng để đọc 1 byte hoặc 2 byte dữ liệu từ cổng vào thanh ghi AL hoặc AX
  - Cú pháp: IN Acc, Port
  - Lệnh này không tác động đến cờ
  - Ví dụ:
    - ⇨ IN AX, 00H
    - ⇨ IN AL, F0H
    - ⇨ IN AX, DX

```
MOV DX,03F8h
IN AL,DX
```

- Lệnh OUT
  - Dùng để đưa 1 byte hoặc 2 byte dữ liệu từ thanh ghi AL hoặc AX ra cổng
  - Cú pháp: OUT Port, Acc
  - Lệnh này không tác động đến cờ
  - Ví dụ:
    - ⇨ OUT 00H, AX
    - ⇨ OUT F0H, AL
    - ⇨ OUT DX, AX
- Chú ý:
  - Nếu Port là hằng số: Port={0 . . . FFH}
  - Nếu muốn truy cập cổng có địa chỉ lớn hơn FFH thì phải dùng Port là DX

Các lệnh di chuyển dữ liệu
Làm việc với cổng

- Lệnh XLAT
  - ❑ Dùng để tra bảng dữ liệu
  - ❑ Địa chỉ bảng dữ liệu được xác đinh bởi: DS:BX
  - ❑ AL chứa vị trí của ô dữ liệu nằm trong bảng cần tra
  - ❑ Dữ liệu cần tra được lưu trong AL sau khi thực hiện lệnh

  Ví dụ:

  Cho DS = 1300h

  BX = 0006h

  DS : BX = 11h

  DS : BX + 1 = 22h

  DS : BX + 2 = 33h

  DS : BX + 3 = 44h

  DS : BX + 4 = 55h

  DS : BX + 5 = 66h

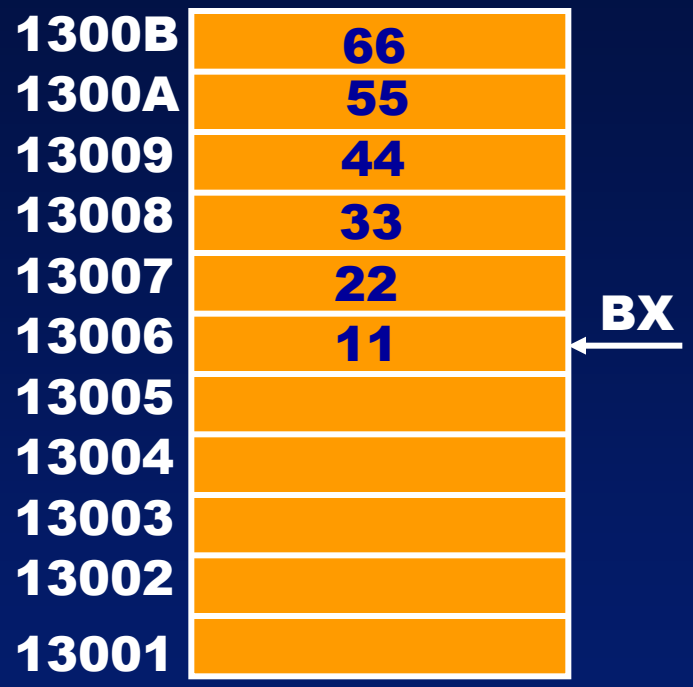
  AL = 03h

  Thực hiện lệnh XLAT

  tìm AL

Các lệnh di chuyển dữ liệu
Làm việc với ngăn xếp

Trước khi thực hiện lệnh



DS 1300

BX 0006

AL 03

XLAT

DS 1300

BX 0006

AL 44

Các lệnh số học và logic

- ADD, ADC, SUB, MUL, IMUL, DIV, IDIV, INC, DEC
- AND, OR, NOT, NEG, XOR
- Lệnh quay và dịch: RCL, RCR, SAL, SAR, SHL, SHR
- Lệnh so sánh: CMP, CMPS

- Lệnh ADD
  - ❑ Lệnh cộng hai toán hạng
  - ❑ Cú pháp: ADD Đích, nguồn
  - ❑ Thực hiện: Đích=Đích + nguồn
  - ❑ Giới hạn: toán hạng không được là 2 ô nhớ và thanh ghi đoạn
  - ❑ Lệnh này thay đổi cờ: AF, CF, OF, PF, SF, ZF

  Ví dụ1:

  Cho AX = 3488h -> thực hiện lệnh ADD AX,BX

  BX = 2212h -> tìm AX và xác định các bit của FR

Các lệnh số học và logic

- Lệnh ADD

Ví dụ2:

Cho AL = 87h

BL = DDh

ADD AL,BL

Tìm AL và các bit của thanh ghi cờ

Ví dụ 3:

Cho AX = 4687h

BX = 4A9Eh

ADD AX, BX

Xác đinh các bit thanh ghi cờ

Ví dụ 4:

Cho AX = 9131h

BX = 1A3Dh

ADD AX,BX Xác đinh các bit thanh ghi cờ

Các lệnh số học và logic

- **Lệnh ADD**

  Ví dụ 5:

  Cho AL = 87h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  ADD AL, [DI]

  Ví dụ 6:

  Cho AX = A387h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  ADD AX, DS

Các lệnh số học và logic
Các lệnh số học

- Lệnh ADC
  - ❑ Lệnh cộng có nhớ hai toán hạng
  - ❑ Cú pháp: ADC Đích, nguồn
  - ❑ Thực hiện: Đích=Đích + nguồn+CF
  - ❑ Giới hạn: toán hạng không được là 2 ô nhớ và thanh ghi đoạn
  - ❑ Lệnh này thay đổi cờ: AF, CF, OF, PF, SF, ZF

Ví dụ1:

Cho AL = 8Dh

DL = AEh

ADC AL, DL

Xác đinh các bit của FR

Ví dụ 2:

Cho AL = 34h

BL = 12h

Xác định các bit của FR

Các lệnh số học và logic
Các lệnh số học

- Lệnh SUB
  - ❑ Lệnh trừ
  - ❑ Cú pháp: SUB Đích, nguồn
  - ❑ Thực hiện: Đích=Đích - nguồn
  - ❑ Giới hạn: toán hạng không được là 2 ô nhớ và thanh ghi đoạn
  - ❑ Lệnh này thay đổi cờ: AF, CF, OF, PF, SF, ZF

  Ví dụ 1:

  Cho

  AX = 1245h

  BX = 11ADh

  SUB AX, BX

  Ví dụ 2:

  Cho

  AX = 1245h

  BX = 11ADh

Các lệnh số học và logic

- **Lệnh SUB**

Ví dụ 3:

Cho AL = 87h

BL = DDh

SUB AL,BL

Tìm AL và các bit của thanh ghi cờ

Ví dụ 4:

Cho AX = 4687h

BX = 4A9Eh

SUB AX, BX

Xác đinh các bit thanh ghi cờ

Ví dụ 5:

Cho AX = 9131h

BX = 1A3Dh

SUB AX, BX Xác đinh các bit thanh ghi cờ

Các lệnh số học và logic

- **Lệnh SUB**

  Ví dụ 6:

  Cho AL = 87h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  SUB AL, [DI]

  Ví dụ 7:

  Cho AX = A387h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  SUB AX, DS

Các lệnh số học và logic
Các lệnh số học

- Lệnh MUL
  - ❑ Lệnh nhân số không dấu
  - ❑ Cú pháp: MUL nguồn (nguồn phải là thanh ghi)
  - ❑ Thực hiện:
    - ⇨ AX=AL* nguồn8bit
    - ⇨ DXAX=AX*nguồn16bit
  - ❑ Lệnh này thay đổi cờ: CF, OF

Ví dụ 1:

Cho AL = 11h

BL = 23h

MUL BL

Ví dụ 2:

Cho AX = 1122h

DS = 2132h

MUL DS

Các lệnh số học và logic
Các lệnh số học

- Lệnh IMUL
  - Lệnh nhân số có dấu
  - Cú pháp: IMUL nguồn (nguồn phải là thanh ghi)
  - Thực hiện:
    - ⇨ AX=AL* nguồn8bit
    - ⇨ DXAX=AX*nguồn16bit
  - Lệnh này thay đổi cờ: CF, OF

Ví dụ 1:

Cho AL = 11h

BL = 23h

IMUL BL

Ví dụ 2:

Cho AX = 1122h

DS = 2132h

IMUL DS

Các lệnh số học và logic
Các lệnh số học

- **Lệnh DIV**
  - ❑ Lệnh chia 2 số không dấu
  - ❑ Cú pháp: DIV nguồn
  - ❑ Thực hiện:
    - ⇨ AL = thương (AX / nguồn8bit) ; AH=dư (AX / nguồn8bit)
    - ⇨ AX = thương (DXAX / nguồn16bit) ; DX=dư (DXAX / nguồn16bit)
  - ❑ Lệnh này không thay đổi cờ

Ví dụ 1:

Cho AL = 11h

BL = 23h

DIV BL

Ví dụ 2:

Cho AX = 1122h

DS = 2132h

Ví dụ 3:

Cho AX = 1122h

DX = 1100h

Các lệnh số học và logic
Các lệnh số học

- Lệnh IDIV
  - ❑ Lệnh chia 2 số có dấu
  - ❑ Cú pháp: IDIV nguồn
  - ❑ Thực hiện:
    - ⇨AL = thương (AX / nguồn8bit) ; AH=dư (AX / nguồn8bit)
    - ⇨AX = thương (DXAX / nguồn16bit) ; DX=dư (DXAX / nguồn16bit)
  - ❑ Lệnh này không thay đổi cờ

Ví dụ 1:

Cho AL = 11h

BL = 23h

IDIV BL

Ví dụ 2:

Cho AX = 1122h

DS = 2132h

Ví dụ 3:

Cho AX = 1122h

DX = 1100h

Các lệnh số học và logic
Các lệnh số học

- **Lệnh INC-Increase by 1**
  - ❑ Lệnh cộng 1 vào toán hạng là thanh ghi hoặc ô nhớ
  - ❑ Cú pháp: INC Đích
  - ❑ Thực hiện: Đích=Đích + 1
  - ❑ Lệnh này thay đổi cờ: AF, OF, PF, SF, ZF

Ví dụ 1:

Cho AX = 1232h

INC AX

Xác đinh AX và các bit cờ

Vi dụ 2:

Cho DS = 1000h

DI = 0000h

DS:DI = 13h

INC [DI]

Xác đinh kết quả phép tính và các bit cờ

Các lệnh số học và logic
Các lệnh số học

- Lệnh DEC-Decrease by 1
  - Lệnh trừ 1 từ nội dung một thanh ghi hoặc ô nhớ
  - Cú pháp: DEC Đích
  - Thực hiện: Đích=Đích - 1
  - Lệnh này thay đổi cờ: AF, OF, PF, SF, ZF

Ví dụ 1:

Cho AX = 1232h

DEC AX

Xác đinh AX và các bit cờ

Vi dụ 2:

Cho DS = 1000h

DI = 0000h

DS:DI = 00h

DEC [DI]

Xác đinh kết quả phép tính và các bit cờ

Các lệnh số học và logic
Các lệnh logic

- Lệnh AND
  - ❑ Lệnh AND logic 2 toán hạng
  - ❑ Cú pháp: AND Đích, nguồn
  - ❑ Thực hiện: Đích=Đích And nguồn
  - ❑ Giới hạn: toán hạng không được là 2 ô nhớ hoặc thanh ghi đoạn
  - ❑ Lệnh này thay đổi cờ: PF, SF, ZF và xoá cờ CF, OF

Ví dụ 1:

Cho

AX = 1245h

BX = 11ADh

AND AX, BX

Ví dụ 2:

Cho

AX = 1245h

BX = 11ADh

AND BX, AX

Các lệnh số học và logic

- **Lệnh AND**

Ví dụ 3:

Cho AL = 87h

BL = DDh

AND AL,BL

Tìm AL và các bit của thanh ghi cờ

Ví dụ 4:

Cho AX = 4687h

BX = 4A9Eh

AND AX, BX

Xác đinh các bit thanh ghi cờ

Ví dụ 5:

Cho AX = 9131h

BX = 1A3Dh

AND AX, BX Xác đinh các bit thanh ghi cờ

Các lệnh số học và logic

- **Lệnh AND**

  Ví dụ 6:

  Cho AL = 87h

  SS = 1000h

  DI = 0000h

  SS: DI = 11h

  AND AL, [DI]

  Ví dụ 7:

  Cho AX = A387h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  AND AX, DS

Các lệnh số học và logic
Các lệnh logic

- Lệnh XOR
  - ❑ Lệnh XOR logic 2 toán hạng
  - ❑ Cú pháp: XOR Đích, nguồn
  - ❑ Thực hiện: Đích=Đích XOR nguồn
  - ❑ Giới hạn: toán hạng không được là 2 ô nhớ hoặc thanh ghi đoạn
  - ❑ Lệnh này thay đổi cờ: PF, SF, ZF , xóa CF,OF

Ví dụ 1:

Cho

AX = 1245h

BX = 11ADh

XOR AX, BX

Ví dụ 2:

Cho

AX = 1245h

BX = 11ADh

XOR BX, AX

Các lệnh số học và logic

- Lệnh XOR

Ví dụ 3:

Cho AL = 87h

BL = DDh

XOR AL,BL

Tìm AL và các bit của thanh ghi cờ

Ví dụ 4:

Cho AX = 4687h

BX = 4A9Eh

XOR AX, BX

Xác đinh các bit thanh ghi cờ

Ví dụ 5:

Cho AX = 9131h

BX = 1A3Dh

XOR AX, BX Xác đinh các bit thanh ghi cờ

Các lệnh số học và logic

- Lệnh XOR

  Ví dụ 6:

  Cho AL = 87h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  XOR AL, [DI]

  Ví dụ 7:

  Cho AX = A387h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  XOR AX, DS

Các lệnh số học và logic
Các lệnh logic

- **Lệnh OR**
  - ❑ Lệnh OR logic 2 toán hạng
  - ❑ Cú pháp: OR Đích, nguồn
  - ❑ Thực hiện: Đích=Đích OR nguồn
  - ❑ Giới hạn: toán hạng không được là 2 ô nhớ hoặc thanh ghi đoạn
  - ❑ Lệnh này thay đổi cờ: PF, SF, ZF, xóa CF, OF

Ví dụ 1:

Cho

AX = 1245h

BX = 11ADh

OR AX, BX

Ví dụ 2:

Cho

AX = 1245h

BX = 11ADh

OR BX, AX

Các lệnh số học và logic

- **Lệnh OR**

Ví dụ 3:

Cho AL = 87h

BL = DDh

OR AL,BL

Tìm AL và các bit của thanh ghi cờ

Ví dụ 4:

Cho AX = 4687h

BX = 4A9Eh

OR AX, BX

Xác đinh các bit thanh ghi cờ

Ví dụ 5:

Cho AX = 9131h

BX = 1A3Dh

OR AX, BX Xác đinh các bit thanh ghi cờ

Các lệnh số học và logic

- **Lệnh OR**

  Ví dụ 6:

  Cho AL = 87h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  OR AL, [DI]

  Ví dụ 7:

  Cho AX = A387h

  DS = 1000h

  DI = 0000h

  DS: DI = 11h

  OR AX, DS

Các lệnh số học và logic
Các lệnh logic

- **Lệnh NOT**
  - ❑ Lệnh đảo 1 toán hạng
  - ❑ Cú pháp: NOT Đích
  - ❑ Thực hiện: Đích = đảo của Đích
  - ❑ Toán hạng: là 1 thanh ghi hoặc ô nhớ
  - ❑ Lệnh này thay đổi cờ: ko cờ nào

Ví du 1:

Cho AX = A23Ch

NOT AX

Ví dụ 2:

Cho DS = 1000h

DI = 0000h

DS:DI = 4Dh

NOT [DI]

Các lệnh số học và logic
Các lệnh logic

- **Lệnh NEG**
  - ❑ Lệnh tìm bù 2 của 1 toán hạng
  - ❑ Cú pháp: NEG Đích
  - ❑ Thực hiện: Đích = bù 2 của Đích
  - ❑ Toán hạng: là 1 thanh ghi hoặc ô nhớ
  - ❑ Lệnh này thay đổi cờ: PF, SF, ZF, CF, OF,AF

Ví du 1:

Cho AX = A23Ch

NEG AX

Ví dụ 2:

Cho DS = 1000h

DI = 0000h

DS:DI = 4Dh

NEG [DI]

Các lệnh số học và logic
Các lệnh so sánh

- Lệnh CMP-Compare 2 operands to Update the Flags
  - ❑ Lệnh so sánh 2 byte hoặc 2 từ
  - ❑ Cú pháp: **CMP Đích, nguồn**
  - ❑ Thực hiện: Đích – Nguồn sau đó update SF và ZF
    - ⇨ Đích = nguồn : SF=0 ZF=1
    - ⇨ Đích> nguồn : SF=0 ZF=0
    - ⇨ Đích < nguồn : SF=1 ZF=0
  - ❑ Giới hạn: toán hạng phải cùng độ dài và không được là 2 ô nhớ

Ví dụ 1:
Cho AX = 1123h
BX = 0054h
CMP AX, BX

Ví du 2:
Cho AL = ABh
DS = 1000h
DI = 0000h
DS:DI = BEh
CMP AL, [DI]

Các lệnh số học và logic
Các lệnh so sánh

- TEST - AND 2 operands to Update the Flags
  - Cú pháp: TEST Operand1,Operand2
  - Thực hiện: Đích And Nguồn -> sau đó update thanh ghi cờ
  - Ví dụ: Kiểm tra bit 0 của AL
    - ⇨ TEST AL,01h ;ZF=1 nếu AL.0=0, ZF=0 nếu AL.0=1
- Các lệnh thiết lập cờ
  - STC-Set the Carry Flag CF=1
  - STD-Set the Direction Flag DF=1
  - STI-Set the Interrupt Flag IF=1
- Các lệnh xóa cờ
  - CLC-Clear the Carry Flag CF=0
  - CLD-Clear the Direction Flag DF=0
  - CLI-Clear the Interrupt Flag IF=0
- CMC-Complement the Carry Flag CF=not(CF)

Các lệnh số học và logic
Dịch và quay

- Lệnh RCL-Rotate through Carry flag to the Left
  - ❑ Lệnh quay trái thông qua cờ nhớ
  - ❑ Cú pháp: RCL Đích, CL
    RCL Đích, Số lần quay
  - ❑ Thực hiện: quay trái đích CL lần
  - ❑ Lệnh này thay đổi cờ: CF, OF

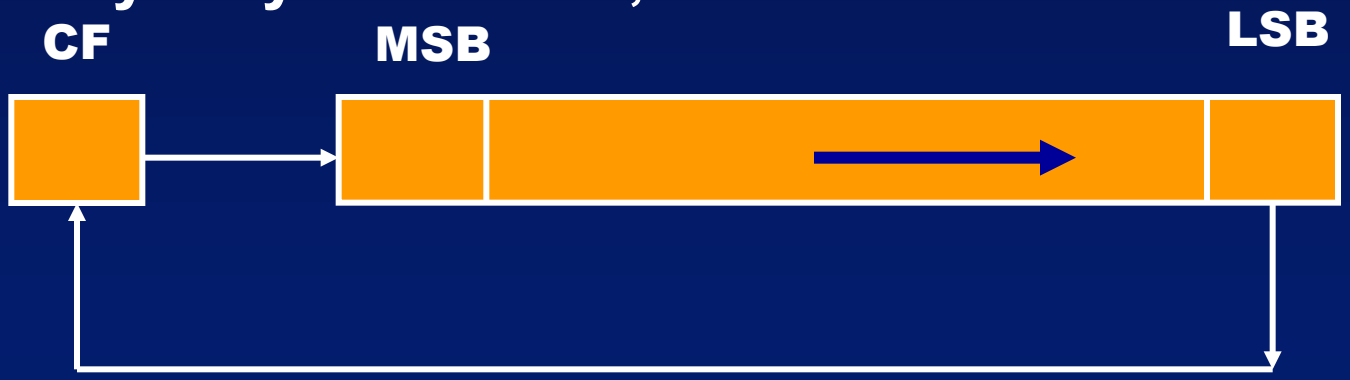
Các lệnh số học và logic
Dịch và quay

- Ví dụ 1: Cho AL = 4Dh & CF = 1 thực hiện RCL AL,1

- Ví dụ 2: Cho AX = 3D2Eh
  CL = 0004h
  CF = 0
  RCL AX,CL
  Tìm AX, CL ?

Các lệnh số học và logic
Dịch và quay

- Lệnh RCR-Rotate through Carry flag to the Right
  - ❑ Lệnh quay phải thông qua cờ nhớ
  - ❑ Cú pháp: RCR Đích, CL
    RCR Đích, Số lần quay
  - ❑ Thực hiện: quay phải đích CL lần
  - ❑ Lệnh này thay đổi cờ: CF, OF



Các lệnh số học và logic
Dịch và quay

- Ví dụ 1: Cho AL = 4Dh & CF = 1 thực hiện RCR AL,1

- Ví dụ 2: Cho AX = 3D2Eh
  CL = 0004h
  CF = 0
  RCR AX,CL
  Tìm AX, CL ?

Các lệnh số học và logic
Dịch và quay

- Lệnh ROL-Rotate All bits to the Left
  - ❑ Lệnh quay vòng trái
  - ❑ Cú pháp: ROL Đích, CL
    ROL Đích, Số lần quay
  - ❑ Thực hiện: quay vòng trái đích CL lần
  - ❑ Lệnh này thay đổi cờ: CF, OF

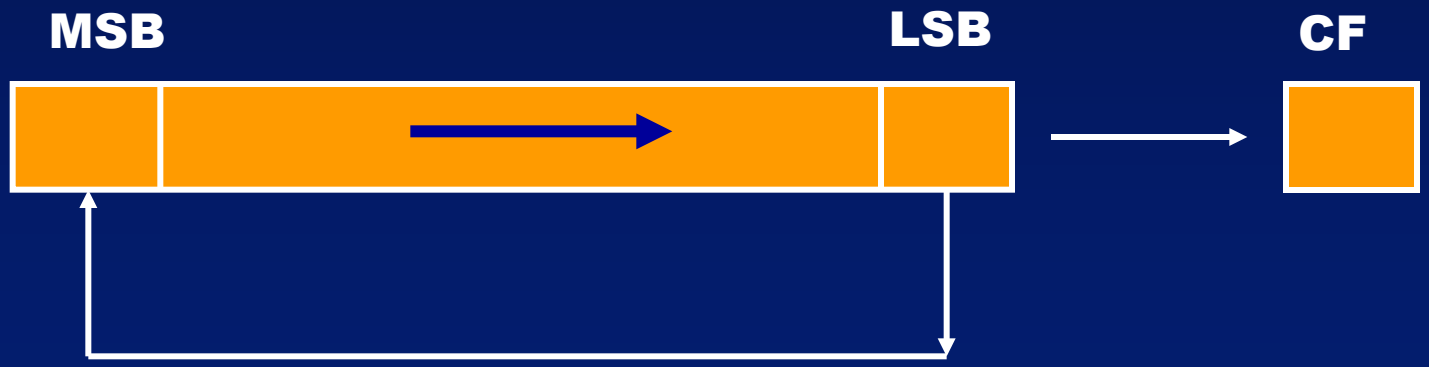
Các lệnh số học và logic
Dịch và quay

- Ví dụ 1: Cho AL = 4Dh & CF = 1 thực hiện ROL AL,1

- Ví dụ 2: Cho AX = 3D2Eh
  CL = 0004h
  CF = 0
  ROL AX,CL
  Tìm AX, CL ?

Các lệnh số học và logic
Dịch và quay

- Lệnh ROR-Rotate All bits to the Right
  - ❑ Lệnh quay vòng phải
  - ❑ Cú pháp: ROR Đích, CL
    ROR Đích, Số lần quay
  - ❑ Thực hiện: quay vòng phải đích CL lần
  - ❑ Lệnh này thay đổi cờ: CF, OF



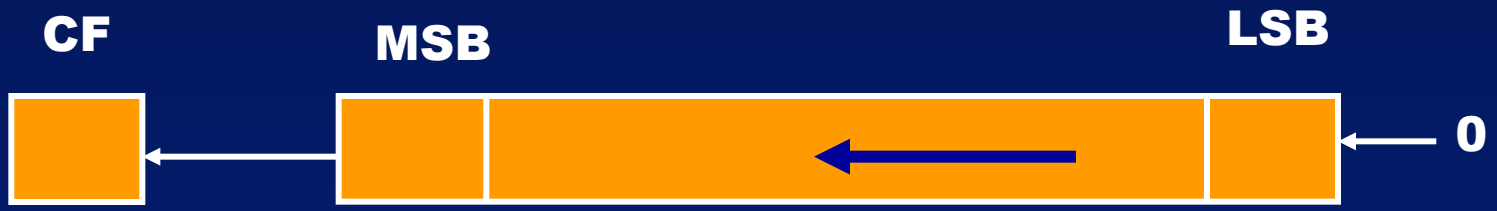
Các lệnh số học và logic
Dịch và quay

- Ví dụ 1: Cho AL = 4Dh & CF = 1 thực hiện ROR AL,1

- Ví dụ 2: Cho AX = 3D2Eh
  CL = 0004h
  CF = 0
  ROR AX,CL
  Tìm AX, CL ?

Các lệnh số học và logic
Dịch và quay

- Lệnh SAL-Shift Arithmetically Left
  - Lệnh dịch trái số học
  - Cú pháp: SAL Đích, CL hoặc SAL Đích, số lần dịch
  - Thực hiện: dịch trái đích CL bit tương đương với Đích=Đích*2^{CL}
  - Lệnh này thay đổi cờ OF, SF, ZF, PF



- Lệnh SHL-SHift Left
  - Lệnh dịch trái logic tương tự như SAL

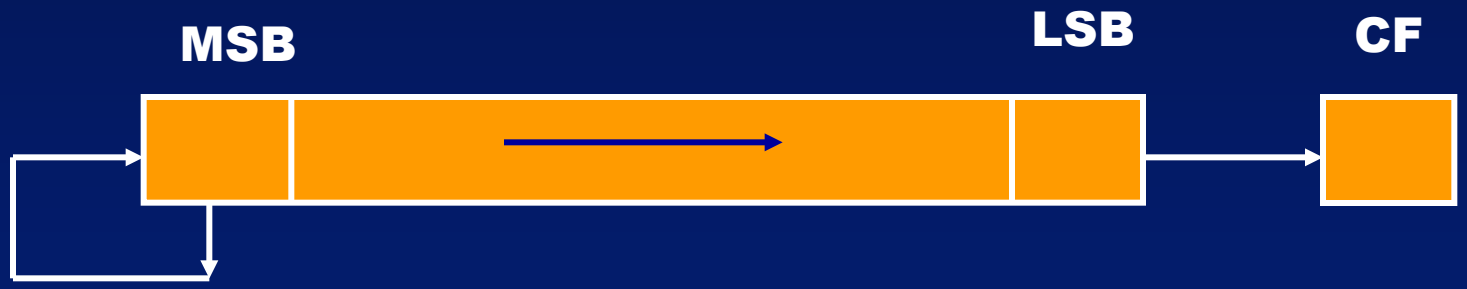
Các lệnh số học và logic
Dịch và quay

- Ví dụ 1: Cho AL = 4Dh & CF = 1 thực hiện SAL AL,1

- Ví dụ 2: Cho AX = 3D2Eh
  CL = 0004h
  CF = 0
  SHL AX,CL
  Tìm AX, CL ?

Các lệnh số học và logic
Dịch và quay

- Lệnh SAR-Shift Arithmetically Right
  - ❑ Lệnh dịch phải số học
  - ❑ Cú pháp: SAR Đích, CL hoặc SAR Đích, số lần dịch
  - ❑ Thực hiện: dịch phải đích CL bit
  - ❑ Lệnh này thay đổi cờ OF, SF, ZF, PF, CF mang giá trị của LSB



Các lệnh số học và logic
Dịch và quay

- Ví dụ 1: Cho AL = 4Dh & CF = 1 thực hiện SAR AL,1

- Ví dụ 2: Cho AX = 3D2Eh
  CL = 0004h
  CF = 0
  SAR AX,CL
  Tìm AX, CL ?

Các lệnh số học và logic
Dịch và quay

- Lệnh SHR-SHift Right
  - ❑ Lệnh dịch phải logic
  - ❑ Cú pháp: SHR Đích, CL hoặc SHR Đích, số lần dịch
  - ❑ Thực hiện: dịch phải đích CL bit
  - ❑ Lệnh này thay đổi cờ OF, SF, ZF, PF, CF mang giá trị của LSB

Các lệnh số học và logic
Dịch và quay

- Ví dụ 1: Cho AL = 4Dh & CF = 1 thực hiện SHR AL,1

- Ví dụ 2: Cho AX = 3D2Eh
  CL = 0004h
  CF = 0
  SHR AX,CL
  Tìm AX, CL ?

Các lệnh điều khiển chương trình
Lệnh nhảy không điều kiện JMP

- Dùng để nhảy tới một địa chỉ trong bộ nhớ
- 3 loại: nhảy ngắn, gần và xa
  - ❑ Lệnh nhảy ngắn (short jump)
    - ⇨ Độ dài lệnh 2 bytes:

E B	Độ lệch

    - ⇨ Phạm vi nhảy: -128 đến 127 bytes so với lệnh tiếp theo lệnh JMP
    - ⇨ Thực hiện: IP=IP + độ lệch
    - ⇨ Ví dụ:

```
        XOR AX, AX
Nhan:   MOV BX, 1
        ADD AX, BX
        JMP SHORT Nhan
```

Các lệnh điều khiển chương trình
Lệnh nhảy không điều kiện JMP

- Lệnh nhảy gần (near jump)
  - ⇨ Phạm vi nhảy: ± 32 Kbytes so với lệnh tiếp theo lệnh JMP
  - ⇨ Ví dụ:

```
XOR BX, BX
Nhan: MOV AX, 1
ADD AX, BX
JMP NEAR Nhan
```

```
XOR CX, CX
MOV AX, 1
ADD AX, BX
JMP NEAR PTR BX
```

```
XOR CX, CX
MOV AX, 1
ADD AX, BX
JMP WORD PTR [BX]
```

Thực hiện: IP=IP+ độ lệch

E 9	Độ lệchLo	Độ lệchHi

IP=BX IP=[BX+1] [BX]

Nhảy gián tiếp

Các lệnh điều khiển chương trình
Lệnh nhảy không điều kiện JMP

❑ Lệnh nhảy xa (far jump)

⇨ Độ dài lệnh 5 bytes đối với nhảy tới nhãn:

E A	d/c Offset Lo	d/c offset Hi	d/c segment Lo	d/c segment Hi

⇨ Phạm vi nhảy: nhảy trong 1 đoạn mã hoặc nhảy sang đoạn mã khác

⇨ Ví dụ:

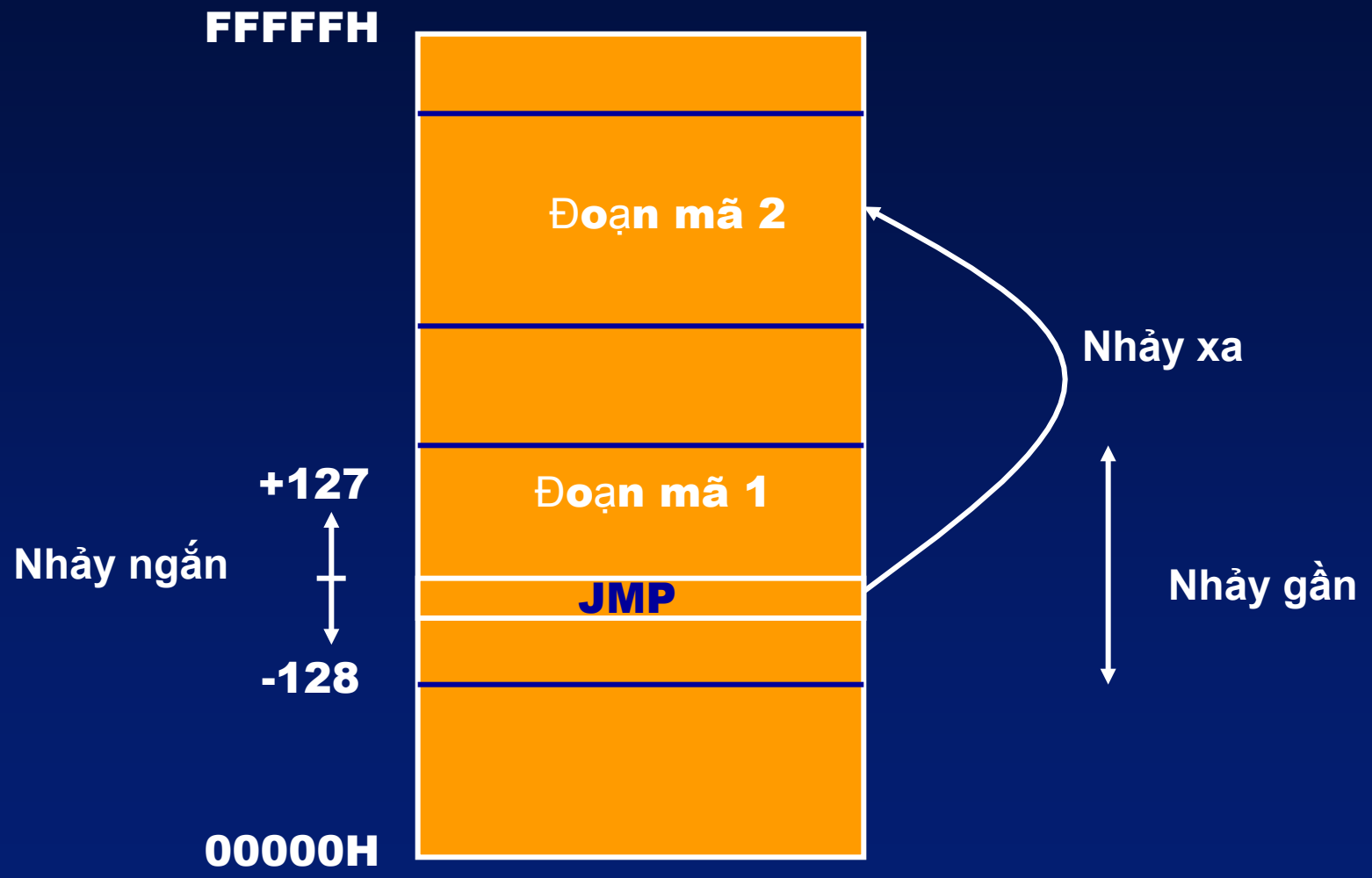
```
EXTRN   Nhan: FAR
Next: MOV AX, 1
      ADD AX, BX
      JMP FAR PTR Next
      .....
      JMP  FAR Nhan
```

Thực hiện: IP=IP của nhãn
CS=CS của nhãn

```
XOR CX, CX
MOV AX, 1
ADD AX, BX
JMP DWORD PTR [BX]
```

IP = [BX+1][BX]
CS= [BX+3][BX+2]

Các lệnh điều khiển chương trình
Tóm tắt lệnh JMP



Các lệnh điều khiển chương trình
Lệnh nhảy có điều kiện

- JE/JZ, JNE/JNZ, JG/JNLE, JGE/JNL, JL/JNGE, JLE/JNG (dùng cho số có dấu) và

  JA/JNBE, JB/JC/JNAE, JAE/JNB/JNC, JBE/JNA (dùng cho số không dấu) và

  JNP/JPO, JP/JPE, JNS
- Nhảy được thực hiện phụ thuộc vào các cờ
- Là các lệnh nhảy ngắn
- Ví dụ:

```
Nhan1: XOR BX, BX
Nhan2: MOV AX, 1
       CMP AL, 10H
       JNE Nhan1
       JE  Nhan2
```

Thực hiện: IP=IP + độ dịch

Các lệnh điều khiển chương trình
Lệnh lặp LOOP

- LOOP, LOOPE/LOOPZ, LOOPNE/LOOPNZ
- Là lệnh phối hợp giữa DEC CX và JNZ

```
        XOR AL, AL
        MOV CX, 16
Lap:    INC AL
        LOOP Lap
```

Lặp đến khi CX=0

```
        XOR AL, AL
        MOV CX, 16
Lap:    INC AL
        CMP AL, 10
        LOOPE Lap
```

Lặp đến khi CX=0
hoặc AL<>10 (ZF=0)

```
        XOR AL, AL
        MOV CX, 16
Lap:    INC AL
        CMP AL, 10
        LOOPNE Lap
```

Lặp đến khi CX=0
hoặc AL=10

Các lệnh điều khiển chương trình
Lệnh CALL (và RET)

- Dùng để gọi chương trình con
- Có 2 loại: CALL gần và CALL xa
  - ❑ CALL gần (near call): tương tự như nhảy gần
    - ⇨Gọi chương trình con ở trong cùng một đoạn mã

```
Tong  PROC NEAR
      ADD AX, BX
      ADD AX, CX
      RET
Tong ENDP
...
CALL Tong
```

Cất IP vào ngăn xếp
IP=IP + dịch chuyển
RET: lấy IP từ ngăn xếp

```
Tong  PROC NEAR
      ADD AX, BX
      ADD AX, CX
      RET
Tong ENDP
...
MOV BX, OFFSET Tong
CALL BX
```

Cất IP vào ngăn xếp
IP= BX
RET: lấy IP từ ngăn xếp

```
CALL WORD PTR [BX]
```

Cất IP vào ngăn xếp
IP= [BX+1] [BX]
RET: lấy IP từ ngăn xếp

Các lệnh điều khiển chương trình
Lệnh CALL

- CALL xa (far call): tương tự như nhảy xa

  ⇨Gọi chương trình con ở ngoài đoạn mã

```
Tong  PROC FAR
      ADD AX, BX
      ADD AX, CX
      RET
Tong ENDP
...
   CALL Tong
```

Cất CS vào ngăn xếp
Cất IP vào ngăn xếp
IP=IP của Tong
CS =CS của Tong
RET: lấy IP từ ngăn xếp
lấy CS từ ngăn xếp

```
CALL DWORD PTR [BX]
```

Cất CS vào ngăn xếp
Cất IP vào ngăn xếp
IP = [BX+1][BX]
CS= [BX+3][BX+2]
RET: lấy IP từ ngăn xếp
lấy CS từ ngăn xếp

Các lệnh điều khiển chương trình
Lệnh CALL

```
Tong  PROC FAR
      ADD AX, BX
      ADD AX, CX
      RET
Tong ENDP
      ...
      MOV BX,1
      CALL Tong
      MOV DX,AX
```

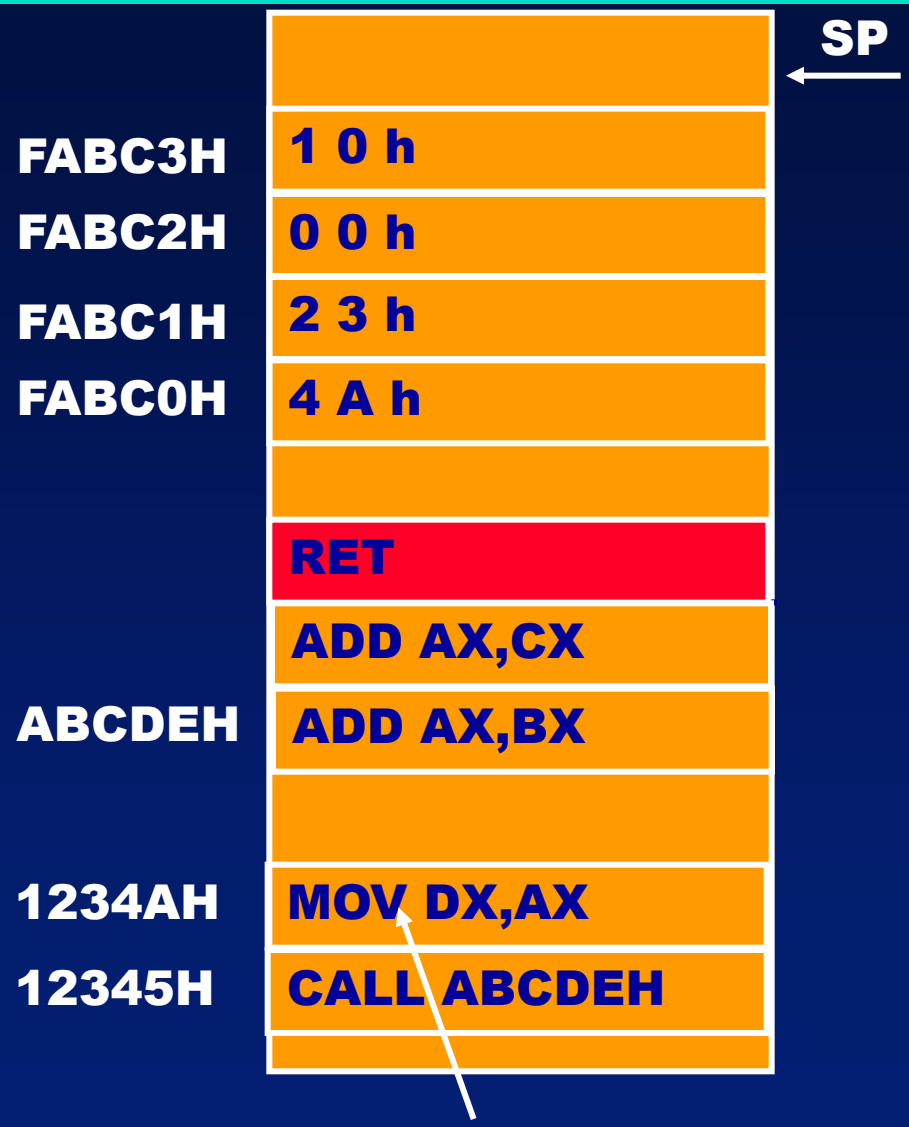
Biên dịch và thực hiện chương trình

CS	1000
IP	2345
SP	ABC4

Các lệnh điều khiển chương trình
Lệnh CALL

Các lệnh điều khiển chương trình
RET

CS ? IP ? SP A B C 0



CS 1 0 0 0 IP 2 3 4 A SP A B C 4

Các lệnh điều khiển chương trình
Lệnh ngắt INT và IRET

- INT gọi chương trình con phục vụ ngắt (CTCPVN)
- Bảng vector ngắt: 1 Kbytes 00000H đến 003FFH
  - 256 vector ngắt
  - 1 vector 4 bytes, chứa IP và CS của CTCPVN
  - 32 vector đầu dành riêng cho Intel
  - 224 vector sau dành cho người dùng

- Cú pháp: INT Number
- Ví dụ: INT 21H gọi CTCPVN của DOS

Các lệnh điều khiển chương trình
Lệnh ngắt INT và IRET

- Thực hiện INT:
  - Cất thanh ghi cờ vào ngăn xếp
  - IF=0 (cấm các ngắt khác tác động), TF=0 (chạy suốt)
  - Cất CS vào ngăn xếp
  - Cất IP vào ngăn xếp
  - IP=[N*4], CS=[N*4+2]
- Gặp IRET:
  - Lấy IP từ ngăn xếp
  - Lấy CS từ ngăn xếp
  - Lấy thanh ghi cờ từ ngăn xếp

Các lệnh
Làm việc với mảng, chuỗi

- Các lệnh di chuyển chuỗi MOVS, MOVSB, MOVSW
  - ❑ Dùng để chuyển một phần tử của chuỗi này sang một chuỗi khác
  - ❑ Cú pháp: MOVS chuỗi đích, chuỗi nguồn
    MOVSB
    MOVSW
  - ❑ Thực hiện:
    - ⇨ DS:SI là địa chỉ của phần tử trong chuỗi nguồn
    - ⇨ ES:DI là địa chỉ của phần tử trong chuỗi đích
    - ⇨ Sau mỗi lần chuyển SI=SI +/- 1, DI=DI +/- 1 hoặc SI=SI +/- 2, DI=DI +/- 2 tuỳ thuộc vào cờ hướng DF là 0/1
  - ❑ Lệnh này không tác động đến cờ
  - ❑ Ví dụ:
    - ⇨ MOVS byte1, byte2
- LODS/LODSB/LODSW-Load String Byte/Word into AL/AX

Lệnh di chuyển chuỗi

- **Bài toán thường gặp:**
  - ❑ Copy ChuoiNguon sang ChuoiDich, kích thước : N

Nạp địa chỉ ChuoiNguon vào 1 thanh ghi: SI
Nạp địa chỉ ChuoiDich vào 1 thanh ghi: DI
Lặp N lần

```
AL <- [SI]
[DI] <- AL
SI=SI+1
DI=DI+1
```

} MOVSB

- **Lệnh CMPS**
  - Dùng để so sánh từng phần tử của 2 chuỗi có các phần tử cùng loại
  - Cú pháp: **CMPS chuỗi đích, chuỗi nguồn**
    CMPSB
    CMPSW
  - Thực hiện:
    - ⇨ DS:SI là địa chỉ của phần tử trong chuỗi nguồn
    - ⇨ ES:DI là địa chỉ của phần tử trong chuỗi đích
    - ⇨ Sau mỗi lần so sánh SI=SI +/- 1, DI=DI +/- 1 hoặc SI=SI +/- 2, DI=DI +/- 2 tuỳ thuộc vào cờ hướng DF là 0/1
  - Cập nhật cờ AF, CF, OF, PF, SF, ZF

Các lệnh khác

- NOP-No Operation (tiêu tốn 3 chu kỳ đồng hồ)
- WAIT-Wait for TEST or INTR Signal
  - Chờ cho đến khi có tín hiệu mức thấp tác động vào chân TEST hoặc mức cao tác động vào chân INTR)
- HALT-Halt Processing
  - VXL dừng hoạt động. Để thoát khỏi trạng thái dừng phải tác động vào các chân INTR, NMI, RESET.
- ESC-Escape
  - Truyền dữ liệu cho bộ đồng xử lý toán học 8087

Bộ vi xử lý Intel 8088/8086

- Cấu trúc bên trong
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8086
- Cách mã hoá lệnh của 8086
- Mô tả tập lệnh của 8086
- **Một số bộ vi xử lý khác**
- Cách quản lý bộ nhớ ở chế độ bảo vệ ở các máy tính từ 80286

1 số bộ vi xử lý khác

VXL 80186

- Registers (16bit):

  General Reg: AX(AH+AL), BX(BH+BL), CX(CH+CL), DX(DH+DL)

  Pointer and index Reg: SP, BP, SI, DI

  Segment Reg: CS, DS, ES, SS

  Flag Reg: OF,CF,AF,SF,ZF,PF,IF,DF,TF

  Control Reg: IR,Timer, DMA, CS, Refresh CU,PowerSaving, Rellocation

- Memory map: similar to 8086
- Instruction set: can be viewed as existing on two levels including an assembly level and a machine level

1 số bộ vi xử lý khác

VXL 80286

- Registers (16bit):

  General Reg: AX(AH+AL), BX(BH+BL), CX(CH+CL), DX(DH+DL)

  Pointer and index Reg: SP, BP, SI, DI

  Segment Reg: CS, DS, ES, SS

  Flag Reg (32bit): OF,CF,AF,SF,ZF,PF,IF,DF,TF,NP,IOPL,TS,EM,PE,MP

  Control Reg: IR,Timer, DMA, CS, Refresh CU,PowerSaving, Rellocation

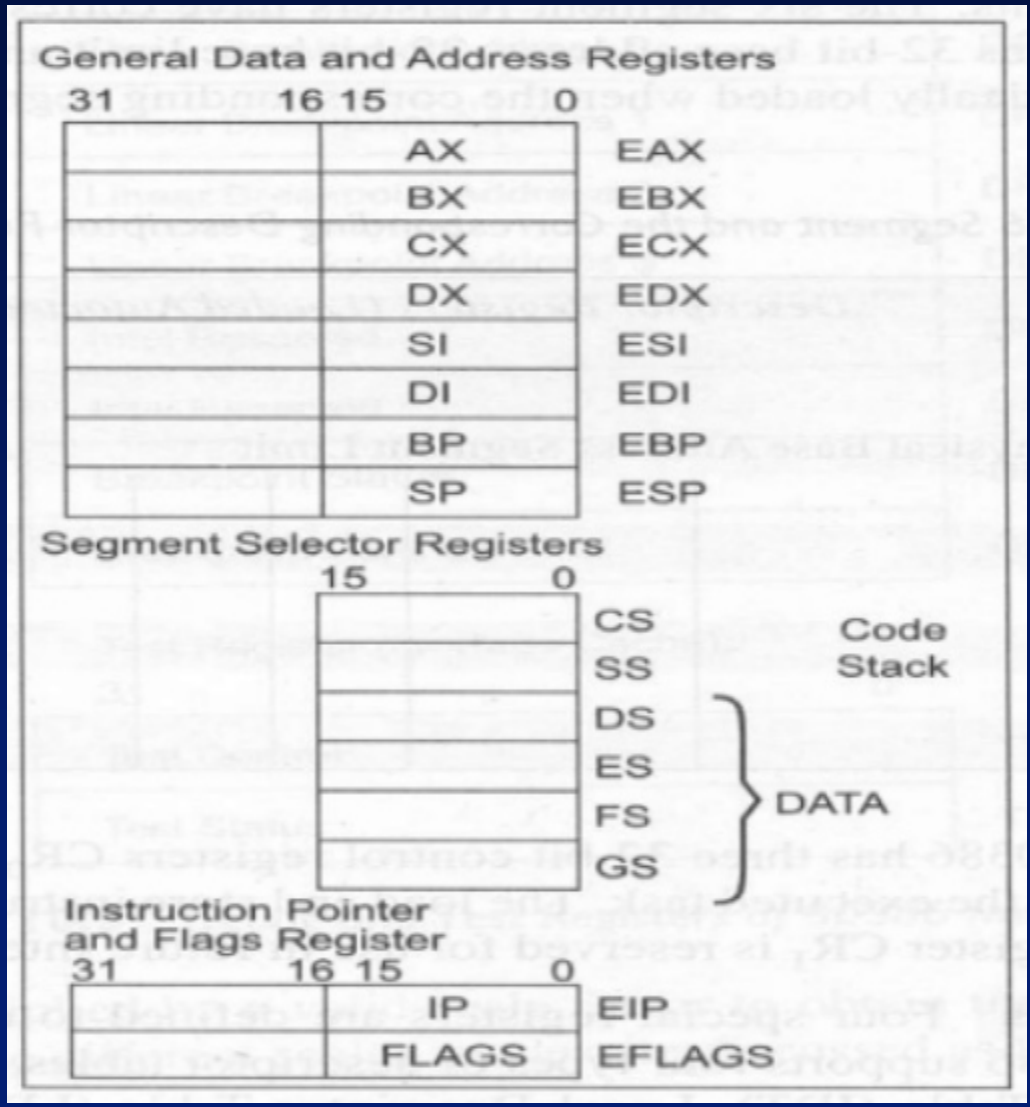
  Status Reg: Status Word

- Memory map: Real Address mode and Virtual memory protected mode
- Instruction set: similar to 8086 with some additional instructions which are used for virtual memory protected mode

1 số bộ vi xử lý khác

VXL 32bit - 80386

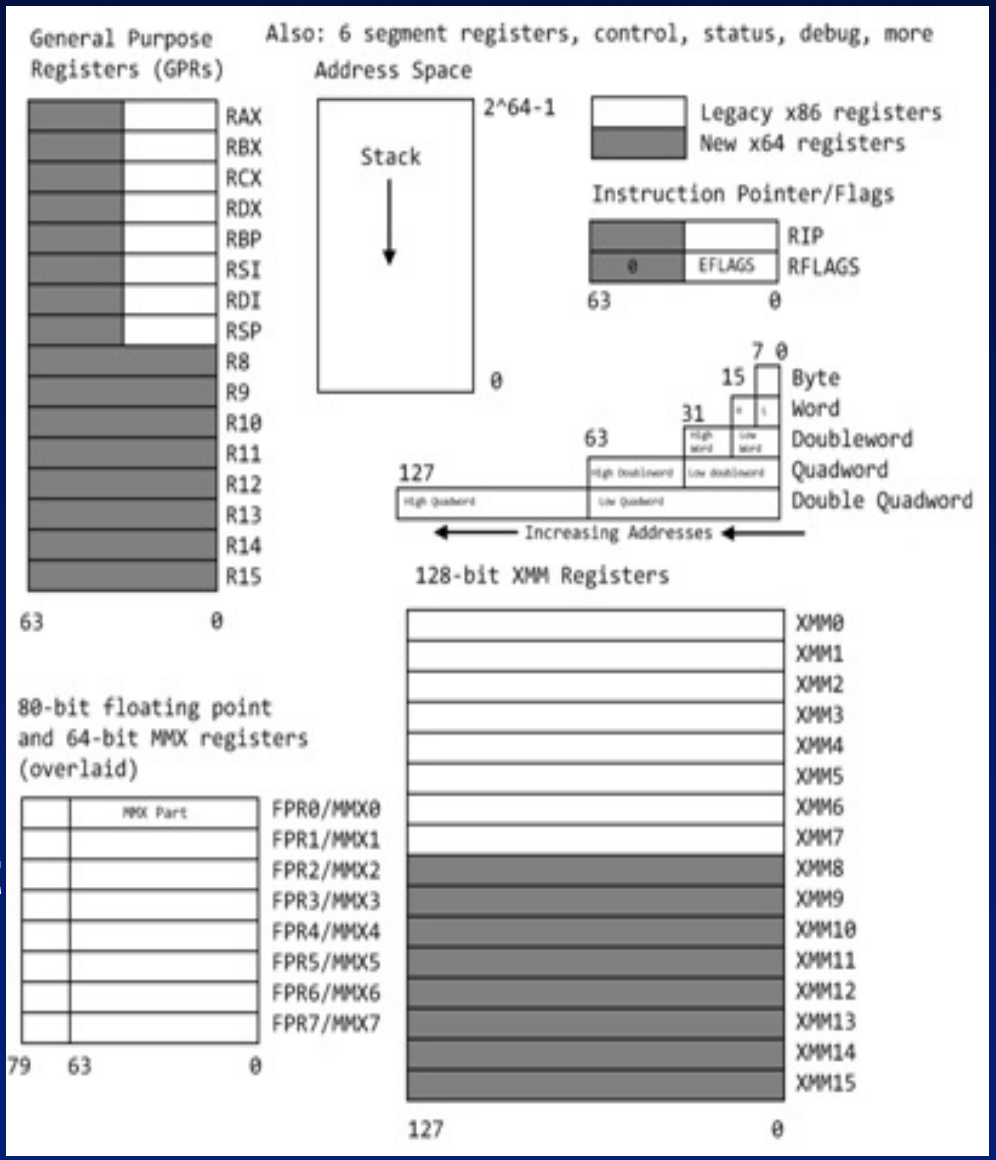
- General registers (32bit):
- Special registers: CR0-4, TR, GDTR,IDTR
- Memory management: using Paging for virtual memory management
- Instruction set: 2 formats (ModR/M or SIB byte) allow to implement 8bit, 16bit , 32bit operations



Sơ đồ cấu trúc 1 số bộ vi xử lý khác

VXL 64bit – Pentium 4

- General registers (64bit):
- Special registers: SSE, GDTR, LDTR, IDTR, TR
- Memory management: still using Paging for virtual memory management with larger size
- Instruction set: most 32bit operations can be used in 64 bit microprocessor.



Khoa ĐIỆN TỬ - VIỄN THÔNG

Bộ vi xử lý Intel 8088/8086

- Cấu trúc bên trong
- Sơ đồ chân
- Bản đồ bộ nhớ của máy tính IBM-PC
- Các chế độ địa chỉ của 8086
- Cách mã hoá lệnh của 8086
- Mô tả tập lệnh của 8086
- số bộ vi xử lý khác
- Cách quản lý bộ nhớ ở chế độ bảo vệ ở các máy tính từ 80286

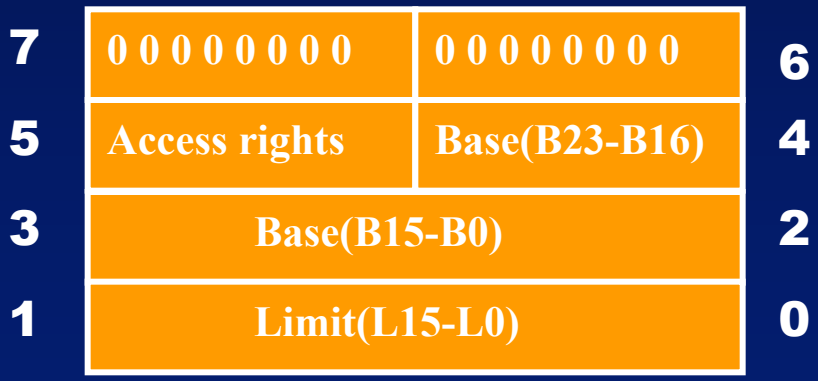
Đánh địa chỉ bộ nhớ ở chế độ bảo vệ

- Lý do:
  - Hỗ trợ đa nhiệm từ 80286.
  - Tương thích ngược với 8086.
  - Cho phép truy cập dữ liệu và chương trình ở vùng nhớ trên 1M
- Thanh ghi lệch chứa địa chỉ lệch
- Thanh ghi đoạn chứa từ chọn đoạn (segment selector)
  - từ chọn đoạn chọn 1 phần tử trong 1 trong 2 bảng mô tả đoạn (Descriptor Table), mỗi bảng có kích thước 64 KB
    - ⇨ Bảng mô tả đoạn toàn cục (Global DT): chứa thông tin về các đoạn của bộ nhớ mà tất cả các chương trình có thể truy nhập
    - ⇨ Bảng mô tả đoạn cục bộ (Local DT): chứa thông tin về các đoạn của 1 chương trình
  - Mô tả đoạn chứa thông tin về địa chỉ bắt đầu của đoạn

Đánh địa chỉ bộ nhớ ở chế độ bảo vệ

segment selector

RPL: mức ưu tiên yêu cầu, 00 cao nhất, 11 thấp nhất
TI=0, sử dụng bảng toàn cục, TI=1 sử dụng bảng cục bộ
Index: 13 bit chỉ số để chọn 1 trong 8K mô tả đoạn trong bảng mô tả đoạn



mô tả đoạn của 80286

mô tả đoạn từ 80386

Base: xác định địa chỉ bắt đầu của đoạn
Limit: giới hạn kích thước tối đa của đoạn

Đánh địa chỉ bộ nhớ ở chế độ bảo vệ

- Mỗi chương trình sử dụng một số vùng nhớ. CPU bảo vệ vùng nhớ đó, không cho phép chương trình khác truy cập vào.
- Không cho phép đọc mã lệnh ở đoạn dữ liệu và ngược lại.

Đánh địa chỉ bộ nhớ ở chế độ bảo vệ

- 80286
  - Base 24 bit: 000000H đến FFFFFFH (16 M đoạn)
  - Limit 16 bit: kích thước đoạn: từ 1 đến 64 KB
  - Địa chỉ vật lý= Base + độ lệch
  - 1 chương trình có thể sử dụng tối đa: 2*8K*64 K= 1GB bộ nhớ => bộ nhớ ảo (virtual memory)
- 80386/486/Pentium
  - Base 32 bit: 00000000H đến FFFFFFFFH (4 GB)
  - Limit 20 bit:
    - ⇨ G=0: kích thước đoạn: từ 1 đến 1MB
    - ⇨ G=1: kích thước đoạn từ 4K đến 4 GB
  - Địa chỉ vật lý= Base + độ lệch
  - 1 chương trình có thể sử dụng tối đa: 2*8K*4 GB= 64 Terabytes bộ nhớ