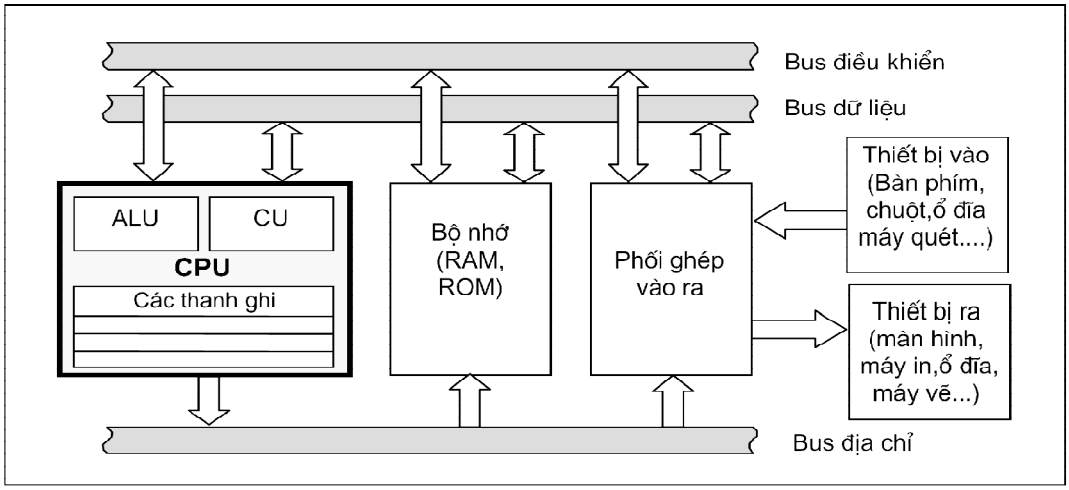
ĐỀ CƯƠNG ÔN TẬP MÔN HỌC CẤU TRÚC MÁY TÍNH.

CHƯƠNG 1: Cấu trúc hệ máy tính.

1. Máy tính làm việc theo nguyên lý Voneumann.

Von Neumann chia hoạt của máy tính thành 5 thành phần chính là:

- CPU (Bộ xử lý trung tâm) – Là thành phần chính của máy tính, đây là nơi sẽ thực hiện các phép tính số học và logic của quá trình xử lý thông tin, đồng thời là nơi sinh ra các tín hiệu để đồng bộ và điều khiển toàn bộ mọi hoạt động của máy tính.
- Bộ nhớ làm việc (RAM) – Là nơi tạo ra môi trường làm việc cho Hệ điều hành và các chương trình ứng dụng.
- Bộ nhớ vĩnh cửu (ROM) – Là nơi chứa các chương trình điều hành hoạt động máy máy tính ở mức độ cơ sở.
- Thiết bị vào (Input) – là các ngoại vi như bàn phím, con chuột, Scanner.v.v. giúp cho máy tính có nhiều khả năng phong phú khi thu thập số liệu và giao tiếp người máy.
- Thiết bị ra (Output) - là các ngoại vi như Màn hình, máy in, máy vẽ, loa bộ nhớ ngoài .v.v. giúp cho máy tính có khả năng phong phú khi xử lý và lưu trữ số liệu cũng như giao tiếp người máy và các thiết bị chuyên dụng khác.



Các thiết bị kể trên được kết nối với nhau thông qua hệ thống Bus bao gồm các tín hiệu:

- Tín hiệu địa chỉ: Tín hiệu này được sinh ra từ CPU hướng đến bộ nhớ và các ngoại vi. Cho phép CPU có khả năng địa chỉ hóa và quản lý được các ô nhớ,

các cổng vào và các cổng ra. Số lượng dây dẫn tạo nên các tín hiệu địa chỉ (độ rộng bus địa chỉ) cho thấy khả năng địa chỉ hóa được các ô nhớ và các cổng vào/ra trên máy tính. Nếu độ rộng của Bus địa chỉ là *k* bits thì máy tính đó có khả năng địa chỉ hóa được 2^k ô nhớ và tối đa 2^k cổng vào và 2^k cổng ra.

- Tín hiệu số liệu: Là tín hiệu 2 chiều cho phép CPU trao đổi thông tin với bộ nhớ hay cổng vào và cổng ra. Trên máy tính thường xuyên diễn ra 2 quá trình cơ bản là quá trình đọc và quá trình ghi. Ở quá trình đọc số liệu sẽ xuất phát từ bộ nhớ hay các cổng vào hướng đến CPU. Ở quá trình ghi, số liệu sẽ xuất phát từ CPU hướng đến bộ nhớ hay các cổng ra.
- Tín hiệu điều khiển: là các tín hiệu cho phép điều khiển khi nào thì CPU đọc hay ghi số liệu, cho phép máy tính thực hiện hay không thực hiện các chức năng như ngắt, DMA, biểu diễn trạng thái của máy tính hay mã hóa các quá trình thực hiện lệnh trên máy tính.

Có 3 tín hiệu điều khiển xuất phát từ CPU để điều khiển quá trình đọc/ghi trên máy tính: $M/\overline{IO}$, $\overline{RD}$, $\overline{WR}$. Từ 3 tín hiệu này máy tính có thể tạo được các tín hiệu điều khiển để đọc bộ nhớ, ghi bộ nhớ hay đọc và ghi vào/ra.

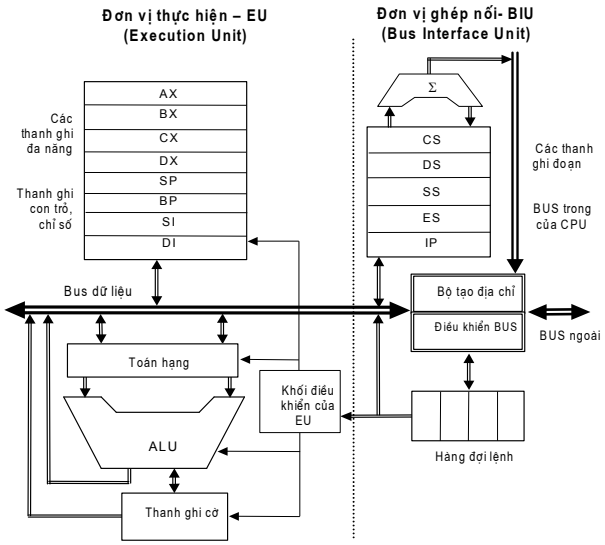
$M/\overline{IO}$	$\overline{RD}$	$\overline{WR}$		**Thao tác trên máy tính**
1	0	1	$\overline{MEMR}$	Đọc bộ nhớ (số liệu xuất phát từ bộ nhớ hướng đến CPU)
1	1	0	$\overline{MEMW}$	Ghi bộ nhớ (số liệu xuất phát từ CPU hướng đến bộ nhớ)
0	0	1	$\overline{IOR}$	Đọc cổng vào (số liệu xuất phát từ cổng vào hướng đến CPU)
0	1	0	$\overline{IOW}$	Ghi cổng ra (số liệu xuất phát từ CPU hướng đến cổng ra)

Tại một thời điểm trên bus điều khiển chỉ tồn tại một trong số 4 tín hiệu trên, đồng thời bus địa chỉ cũng chỉ tồn tại 1 giá trị, do đó tại 1 thời điểm máy chỉ có thể thực hiện 1 thao tác trong số 4 thao tác cơ bản của máy tính. Hay nói cách khác CPU chỉ có thể thực hiện trao đổi thông tin với 1 ô nhớ, 1 cổng vào ra hay 1 cổng ra trên máy tính. Nói về quá trình thực hiện lệnh thì CPU tại một thời điểm chỉ có thể thực hiện được 1 lệnh mà thôi. Quá trình này gọi là quá trình xử

lý tuần tự (step-by-step). Đây là nhược điểm lớn nhất của máy tính hoạt động theo nguyên lý Voneumann. Vì trong các bài toán xử lý thông tin trong thực tế, tại 1 thời điểm, máy tính thường xuyên cần phải đồng thời trao đổi thông tin với nhiều ngoại vi, nguyên lý Voneumann như đã trình bày ở trên không cho phép máy tính thực hiện được nhiệm vụ này. Để khắc phục nhược điểm này người ta phải tăng tốc độ làm việc của máy tính, xây dựng các mạng máy tính xử lý song song và đưa ra một mô hình máy tính mới hoạt động theo nguyên lý xứ lý song song – máy tính mạng nơron (Neural network).

2. Bộ xử lý trung tâm của máy tính.

Máy tính cá nhân đầu tiên mà IBM chế tạo (IBM PC XT) sử dụng Bộ VXL 8086/8088 của Intel. Đây là Bộ VXL 16 bits có cấu trúc như sau:



Với cấu trúc được chia thành hai phần: Đơn vị thực hiện (EU) và đơn vị giao tiếp (BIU) cho phép CPU có thể đồng thời vừa thực hiện lệnh và vừa đọc những lệnh tiếp theo sẽ thực hiện vào hàng đợi lệnh trong đơn vị giao tiếp, đây là một tiến bộ đáng kể mở đầu cho cấu trúc đường ống được xây dựng ở các BVXL tiên tiến của Intel.

Các thanh ghi đa năng của BVXL 8086/88

Để tương thích với các hệ 8 bits được chế tạo trước đó vẫn đang còn được sử dụng rộng rãi các thanh ghi đa năng của đơn vị thực hiện có thể được sử dụng như các thanh ghi 8 bits, khi đó thanh ghi 16 bits AX, BX, CX và DX được

chia thành các thanh ghi 8 bits tương ứng là AH-AL, BH-BL, CH-CL và DH-DL.

Các thanh ghi của CPU có các chức năng ngầm định được định nghĩa như sau:

Thanh ghi AX (AH – AL): Thanh ghi tích lũy (thực hiện các phép cộng, trừ, nhân chia và vào ra) .

Thanh ghi BX: Thanh ghi cơ sở (Thực hiện các phép lưu trữ, chuyển đổi số liệu)

Thanh ghi CX (CL): Là thanh ghi đếm cho các phép tính chuỗi, lặp, dịch chuyển, quay vòng .

Thanh ghi DX: Là thanh ghi tạm thời cho các phép nhân, chia và là nơi chứa địa chỉ cổng trong thao tác vào/ra với mode địa chỉ gián tiếp.

Địa chỉ vật lý – địa chỉ logic

Mặt khác, BVXL 8086/88 có 20 bits địa chỉ cho phép địa chỉ hóa tối đa được 2^{20} tương đương 1MB ô nhớ, trong khi đó các thanh ghi bên trong của BVXL chỉ là 16 bits vì vậy có hai phương pháp xác định địa chỉ:

Địa chỉ vật lý: Đây là một địa chỉ được biểu diễn bởi 1 số 20 bits cho phép biểu diễn chính xác vị trí của ô nhớ trong tổng số 2^{20} ô nhớ mà BVXL có khả năng địa chỉ hóa được. Ví dụ địa chỉ 275BA H hay 90000 H. Địa chỉ vật lý được sử dụng trong mạch giải mã để xác định ô nhớ nào hay cổng vào/ra nào sẽ trao đổi số liệu với BVXL.

Địa chỉ logic: Các thanh ghi của BVXL có độ dài chỉ 16 bits nên không thể biểu diễn được một địa chỉ 20 bits. Vì vậy, để biểu diễn một địa chỉ 20 bits người ta sử dụng các thanh ghi đoạn nằm trong thành phần của đơn vị giao tiếp (BUI) kết hợp với các thanh ghi con trỏ chỉ số trong đơn vị thực hiện (EU) theo quy tắc sau:

Thanh ghi đoạn	Thanh ghi con trỏ chỉ số, con trỏ lệnh (độ lệch – offset)	Thao tác
CS	IP	Nhận lệnh
DS	BX, SI, DI	Dữ liệu
SS	SP	Ngăn xếp
ES	DI, BP	Nơi gửi tới

Khi đó 1 địa chỉ 20 bits sẽ được biểu diễn như sau:

CS : IP, DS:BX, SS:SP .v.v. cách viết như thế được gọi là địa chỉ logic. Từ 1 địa chỉ logic người ta xác định địa chỉ vật lý bằng cách:

Dịch trái thanh ghi đoạn 4 bits rồi cộng số học với giá trị của thanh ghi độ lệch giữ nguyên. Ví dụ:

Giá trị của thanh ghi đoạn DS = 22B8 H.

Giá trị của thanh ghi độ lệch SI = 7820 H.

Địa chỉ logic được biểu diễn: DS:SI (22B8:7820)

Ta xác định được địa chỉ vật lý như sau:

Địa chỉ đoạn	22B8	Dịch trái 4 bits	2	2	B	8	0
Địa chỉ offset	7820	Giữ nguyên		7	8	2	0
Địa chỉ vật lý 20 bits			**2**	**A**	**3**	**A**	**0**

Việc dịch trái thanh ghi đoạn 4 bits tương đương với việc nhân giá trị thanh ghi đoạn với 10H (16 D).

Biểu diễn địa chỉ logic cho phép chia không gian của bộ nhớ máy tính thành 16 đoạn có số thứ tự từ 0 H- F H (tương đương với 4 bits cao nhất của địa chỉ 20 bits thay đổi từ 0000 B – 1111 B).

Với cách tính địa chỉ vật lý từ địa chỉ logic ta thấy với 1 địa chỉ vật lý sẽ có thế có nhiều địa chỉ logic (nếu tăng giá trị của thanh ghi đoạn lên 1 đơn vị hexa và giảm giá trị của thanh ghi độ lệch 10 đơn vị hexa hoặc ngược lại thì giá trị của địa chỉ vật lý không thay đổi). Giá trị lớn nhất của thanh ghi độ lệch là FFFF H (2^{16} giá trị) mỗi lần thay đổi (2^4 giá trị) như vậy với 1 địa chỉ vật lý ta có thể có tối đa $2^{16}:2^4 = 2^{12}$ giá trị địa chỉ logic. Trường hợp địa chỉ vật lý là 00000 H ta chỉ có 1 địa chỉ logic mà thôi.

Cấu trúc đường ống trong BVXL

Với cấu trúc hàng đợi lệnh ở BVXL 8086/88 ta thấy khi EU đang thực hiện lệnh, thì lệnh tiếp theo sẽ được thực hiện (có địa chỉ tại CS:IP) sẽ được nạp vào hàng đợi lệnh. Cấu trúc này cho phép quá trình đọc bộ nhớ được thực hiện đồng thời với quá trình xử lý lệnh. Xuất phát từ ý tưởng đó, ta thấy: nếu chia quá trình thực hiện một lệnh thành nhiều giai đoạn, cấu trúc hàng đợi lệnh sẽ cho phép thực hiện đồng thời các giai đoạn khác nhau của các lệnh kế tiếp nhau. Đó chính là bản chất của cấu trúc đường ống. Giả sử một lệnh máy được chia thành 5 giai đoạn:

Đọc lệnh (IF: Intruction Fetch).

Giải mã lệnh (ID: Intruction Decode).

Thi hành lệnh (EX: Execute).

Thâm nhập bộ nhớ trong hoặc nhảy (MEM: Memory access)

Lưu trữ kết quả (RS: Result Storing).

Mỗi giai đoạn có thể được thực hiện trong 1 hay nhiều xung Clock.

Chuỗi lệnh	**Chu kỳ xung đồng hồ**								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Lệnh thứ i	IF	ID	EX	MEM	RS				
Lệnh thứ i+1		IF	ID	EX	MEM	RS			
Lệnh thứ i+2			IF	ID	EX	MEM	RS		
Lệnh thứ i+3				I F	ID	EX	MEM	RS	
Lệnh thứ i+4					IF	ID	EX	MEM	RS

Trong tiến trình trên ta thấy trong chu kỳ của 1 xung clock nhiều giai đoạn khác nhau của lệnh được thực hiện đồng thời trên các phần tử khác nhau trong BVXL. Với 5 lệnh kế tiếp nhau, nếu không có cấu trúc đường ống sẽ cần đến 45 chu kỳ để thực hiện. Nhưng 5 lệnh đó chỉ thực hiện trong 9 chu kỳ nếu BVXL có cấu trúc đường ống đó là ưu điểm nổi bật của cấu trúc đường ống. Cấu trúc đường ống cũng là khởi đầu cho các cấu trúc vô hướng, siêu vô hướng, phân luồng và siêu phân luồng trên các BVXL tiên tiến của Intel.

CHƯƠNG 2: BỘ NHỚ VÀ HỆ THỐNG BUS.

1. Bộ nhớ trên máy tính.

Bộ nhớ của máy tính là nơi chứa các chương trình điều hành hoạt động của máy tính ở mức độ cơ sở (ROM BIOS) và là môi trường làm việc cho Hệ điều hành và các chương trình ứng dụng của máy tính (RAM).

Căn cứ vào đặc điểm làm việc của bộ nhớ theo nguyên lý Vonneumann có hai loại bộ nhớ:

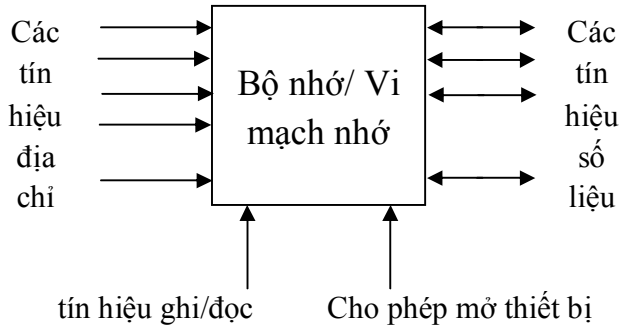
Bộ nhớ vĩnh cửu (ROM) là loại bộ nhớ mà thông tin ghi trong nó không bị mất đi khi cắt nguồn làm việc của máy tính. Do đó, các chương trình điều hành

hoạt động của máy được ghi vào bộ nhớ này và người ta thường gọi là ROM BIOS (Rom Base Input-Output System).

Bộ nhớ làm việc (RAM). Khác với bộ nhớ ROM, thông tin ghi trong bộ nhớ RAM sẽ bị mất khi ta cắt nguồn làm việc hay khởi động lại máy tính. Bộ nhớ RAM là môi trường làm việc của HĐH và các chương trình ứng dụng, để làm việc được hiệu quả thì máy tính cần có bộ nhớ RAM dung lượng lớn. Để có bộ nhớ RAM dung lượng lớn, giá thành hạ người ta sử dụng bộ nhớ RAM động, tuy nhiên do sử dụng tụ điện làm phần tử nhớ nên để tránh việc bị mất thông tin do hiện tượng tụ điện tự phóng điện phải có quá trình nạp lại điện áp cho tụ điện được gọi quá trình "làm tươi" bộ nhớ RAM động, quá trình này làm giảm tốc độ làm việc của bộ nhớ RAM động.

Bộ nhớ của máy tính được tổ hợp từ các vi mạch nhớ và có dung lượng được tính bằng đơn vị Byte, trong khi đó dung lượng của vi mạch nhớ được tính bằng đơn vị bit.

Bộ nhớ của máy tính được kết nối với CPU thông qua hệ thống BUS gồm các tín hiệu sau:



Số lượng các tín hiệu địa chỉ có trên bộ nhớ của máy tính biểu diễn dung lượng bộ nhớ máy tính (Nếu bộ nhớ máy tính có *k* bits địa chỉ thì dung lượng bộ nhớ sẽ là 2^k ô nhớ) mỗi ô nhớ trong bộ nhớ máy tính có chiều dài là 1, 2, 4, 8 hoặc 16 Byte số liệu. Khi đó dung lượng bộ nhớ máy tính sẽ là 2^k * *1 (2,4,8,16) Byte.*

Số lượng các tín hiệu địa chỉ có trên vi mạch nhớ biểu diễn dung lượng của vi mạch nhớ (Nếu vi mạch nhớ *n* bits địa chỉ thì dung lượng của vi mạch nhớ sẽ là 2^n ô nhớ) mỗi ô nhớ của vi mạch nhớ có chiều dài là 1, 2, 4, 8 bits số

liệu. Khi đó người ta nói dung lượng của vi mạch nhớ sẽ được xác định là 2^n * *1(2,4,8) bit*

Tổ hợp nhiều vi mạch nhớ sẽ tạo nên bộ nhớ máy tính, tùy theo số chân số liệu của vi mạch nhớ, người ta sẽ phải lắp song song các vi mạch nhớ với nhau để đảm bảo mỗi lần CPU truy cập bộ nhớ thì 1, 2, 4, 8 hay 16 Byte số liệu sẽ được ghi hoặc đọc.

Khi thực hiện truy cập bộ nhớ, tại 1 thời điểm CPU chỉ có thể đọc hay ghi 1 ô nhớ để xác định được ô nhớ sẽ thực hiện trao đổi số liệu với CPU người ta phải thực hiện giải mã bộ nhớ. Trước tiên là mạch giải mã bộ nhớ để xác định xem vi mạch nhớ nào chưa ô nhớ sẽ trao đổi số liệu với CPU và sau đó là giải mã trong nội bộ vi mạch nhớ để xác định được ô nhớ sẽ trao đổi số liệu với CPU.

Bộ nhớ của máy tính có *k* bits địa chỉ (máy tính có 2^k ô nhớ), vi mạch nhớ để tổ hợp nên bộ nhớ có *n* bits địa chỉ (tùy theo số chân số liệu mà vi mạch nhớ có được ta ghép song song các vi mạch nhớ với nhau đảm bảo mỗi lần truy cập thì 1, 2, 4, hoặc 8 Byte số liệu được ghi hay đọc) tạo được 2^n ô nhớ ($n < k$), quá trình giải mã chọn ô nhớ sẽ được tiến hành như sau:

Nếu ($k - n = m$)

m dây địa chỉ phần cao được nối đến mạch giải mã: $A_{k-1} A_{k-2} A_{k-3} ... A_n$.

n dây địa chỉ phần thấp được nối đến *n* chân địa chỉ của vi mạch nhớ: $A_{n-1} A_{n-2} A_{n-3} ... A_1 A_0$.

Với *m* tín hiệu địa chỉ đầu vào, mạch giải mã sẽ có 2^m đầu ra được dùng để chọn tối đa 2^m vi mạch nhớ tương ứng với 2^m tổ hợp của các tín hiệu địa chỉ $A_{k-1} A_{k-2} A_{k-3} ... A_n$. Tại vi mạch nhớ, với *n* tín hiệu địa chỉ sẽ có 2^n ô nhớ trong mỗi vi mạch. Với tổ hợp địa chỉ $A_{n-1} A_{n-2} A_{n-3} ... A_1 A_0$ mạch giải mã trong vi mạch nhớ sẽ có 2^n đầu ra, mỗi đầu ra tương ứng với 1 ô nhớ. Cách tổ chức như trên cho phép chọn được chính xác ô nhớ nào trong tổng số 2^k ô nhớ của máy tính sẽ thực hiện trao đổi số liệu với CPU.

Trên các máy tính hiện đại có dung lượng bộ nhớ RAM lớn, người ta thường tổ chức bộ nhớ RAM của máy tính từ các Modul, mỗi modul RAM là

tập hợp của nhiều vi mạch nhớ. Khi đó phương pháp giải mã để xác định ô nhớ sẽ trao đổi số liệu với CPU vẫn được thực hiện theo nguyên tắc trên.

Các modul bộ nhớ RAM của máy tính được chia thành các loại sau:

SIMM RAM:

DIMM RAM: SD RAM và DDR RAM

RD RAM (RAM BUS)

Máy tính IBM PC XT được xây dựng trên BVXL 8086/88 có 20 chân địa chỉ do đó dung lượng bộ nhớ của máy tính này là 2^{20} ô nhớ có địa chỉ từ 00000 h đến FFFFF h được phân chia như sau:

FFFFF FE000	8 K BIOS
FDFFF F6000	32 K Chương trình dịch BASIC
F5FFF F4000	8K Dành cho User
F3FFF CA000	18 K ROM mở rộng
C9FFF C8000	8 K ROM điều khiển đĩa
C7FFF C0000	32 K ROM mở rộng
BFFFF A0000	128 K Video RAM
9FFFF 00600	Dành cho hệ điều hành
005FF 00500	Dành cho DOS và BASIC chứa các tham số tạm thời
004FF 00400	Vùng dữ liệu tạm thời của BIOS
003FFF 000000	Bảng vector ngắt

Khi khởi động máy tính, bộ nhớ ROM BIOS sẽ phải được khởi tạo đầu tiên do đó trạng thái các thanh ghi khi khởi động máy tính sẽ như sau:

CPU	CS	DS	SS	ES	IP	Cờ	Hàng đợi
Nội dung	FFFF	0000	0000	0000	0000	Xóa	Trống

Lệnh đầu tiên sẽ được đọc từ ô nhớ có địa chỉ CS:IP là FFFF:0000 tương ứng với địa chỉ vật lý là FFFF0 h. Đó chính là lý do tại sao ROM BIOS lại được đặt tại vùng cao nhất của không gian bộ nhớ máy tính.

Vùng nhớ tiêu chuẩn của máy tính có dung lượng 1 MB có địa chỉ từ 00000 h đến FFFFF h được chia thành 2 loại:

Bộ nhớ quy ước có địa chỉ từ 00000h đến 9FFFFh có dung lượng là 640KB, đây là vùng nhớ RAM được dùng để chứa các tham số của HĐH, các tham số về cấu hình của máy (như trên bản đồ bộ nhớ của máy tính), phần lớn nhất của bộ nhớ quy ước được dùng làm môi trường cho các chương trình ứng dụng hoạt động.

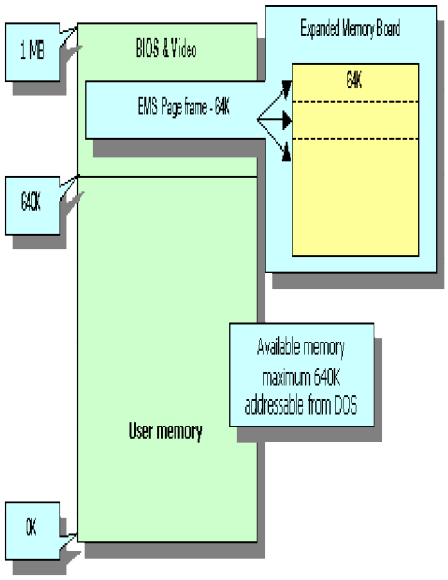
Bộ nhớ trên có địa chỉ từ A0000h đến FFFFFh được chia thành 2 phần: bộ nhớ video RAM có dung lượng 128 KB từ địa chỉ A0000h đến BFFFFh. Bộ nhớ ROM có dung lượng 256 KB từ địa chỉ C0000h đến FFFFFh trong đó vùng nhớ ROM BIOS bao giờ cũng được nằm tại vùng có địa chỉ cao nhất, đó là lý do mà khi khởi động máy tính giá trị của thanh ghi CS và IP phải là FFFF:0000 để chỉ đến ô nhớ có địa chỉ vật lý là FFFF0h.

Thực tế khi máy tính chạy các chương trình ứng dụng lớn thì dung lượng bộ nhớ RAM tại vùng nhớ quy ước không đủ để đáp ứng đòi hỏi của chương trình, mặt khác các bộ VXL từ 80286 trở đi, cấu trúc BVXL có nhiều hơn 20 đường địa chỉ cho phép địa chỉ hóa được không gian bộ nhớ nhiều hơn 1 MB theo quy định. Để đảm bảo có bộ nhớ RAM lớn hơn 1MB theo yêu cầu của các chương trình ứng dụng người ta cần phải có các tiêu chuẩn cho phép HĐH quản lý được không gian bộ nhớ này. Trên cơ sở đó người ta đưa ra 2 hình thức để quản lý bộ nhớ nằm ngoài không gian tiêu chuẩn này.

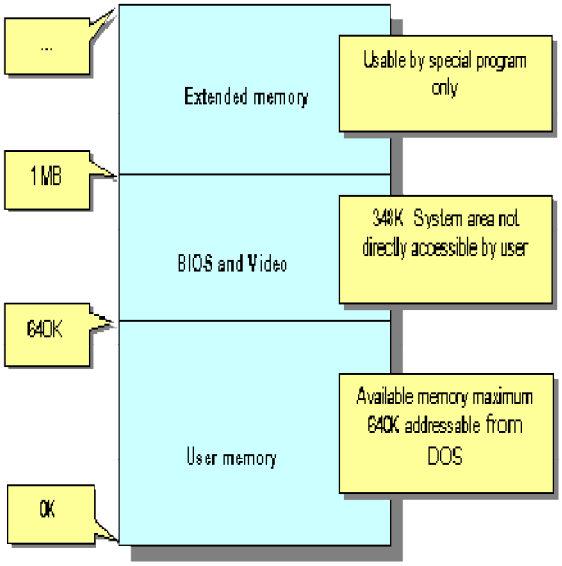
Bộ nhớ phát triển là bộ nhớ nằm ngoài không gian 1 MB tiêu chuẩn, để quản lý bộ nhớ này HĐH sẽ dùng 1 segment bộ nhớ có địa chỉ từ E0000h đến EFFFFh làm cửa sổ cho phép truy cập đến vùng nhớ này, khi đó một chương trình có tên là EMS.EXE sẽ được HĐH sử dụng để cho phép quản lý vùng nhớ

này. Ban đầu EMS chỉ cho phép quản lý được số liệu nằm trên bộ nhớ mở rộng, sau này EMS cho phép vùng nhớ mở rộng chứa được cả số liệu và mã lệnh.

Bộ nhớ phát triển, bộ VXL từ 80286 trở đi có số chân địa chỉ nhiều hơn 20, số chân địa chỉ như vậy cho phép máy tính có khả năng địa chỉ hóa được nhiều hơn 2^{20} ô nhớ như BVXL 80286 có 24 chân địa chỉ có thể địa chỉ hóa được 2^{24} ô nhớ, BVXL 80386 có 32 chân địa chỉ có thể địa chỉ hóa được 2^{32} ô nhớ.v.v. HĐH cần phải có một chương trình riêng để quản lý được các ô nhớ đó, chương trình XMS.EXE được dùng để quản lý bộ nhớ này và nó được gọi là bộ nhớ phát triển. Như vậy bộ nhớ phát triển là bộ nhớ nằm ngoài 1MB có địa chỉ bắt đầu từ 100000h.



Bộ nhớ mở rộng



Bộ nhớ phát triển

Vùng nhớ cao (High Memory area)

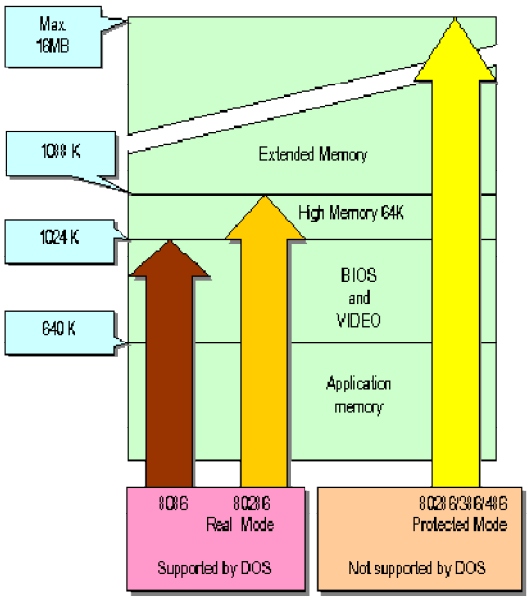
Ở chế độ thực tất cả các BVXL từ 80286 trở lên đều có thể truy cập 65520 bytes bộ nhớ có địa chỉ từ 100000 – 10FFEF (65520 ô nhớ) mà không cần hệ điều hành làm việc ở chế độ bảo vệ. Không gian nhớ này được gọi là vùng nhớ cao (HMA). Khi kết hợp giữa địa chỉ đoạn và offset để tạo địa chỉ vật lý, địa chỉ đoạn được dịch trái 4 bits. BVXL 8086/88 không có chân A20 nên không có nhớ. BVXL 80286 có chân A20 nên khi dịch trái phần có nhớ được chuyển sang

chân A20. HĐH thường sử dụng vùng nhớ cao để chứa các lệnh nội trú nhằm tăng không gian cho các chương trình ứng dụng bằng các lệnh:

DEVICE = HIMEM.SYS

DOS = HIGH

Lệnh DOS = HIGH chuyển các lệnh nội trú của HĐH lên vùng nhớ cao và giải phóng toàn bộ vùng nhớ quy ước cho các chương trình ứng dụng.



Vùng nhớ cao và bộ nhớ phát triển

2. Hệ thống Bus trên máy tính.

Bus trên máy tính là tập hợp các tín hiệu địa chỉ, số liệu, điều khiển cùng với các tín hiệu đồng bộ, nguồn được chế tạo tuân theo một tiêu chuẩn nhất định đảm bảo cho việc trao đổi số liệu giữa CPU với bộ nhớ và các thiết bị ngoại vi.

Một hệ Bus tiên tiến luôn đảm bảo được tốc độ truyền số liệu giữa CPU bộ nhớ và ngoại vi là nhanh nhất, đồng thời có khả năng nhận dạng được sự thay đổi của thiết bị ngoại vi hay bộ nhớ để đảm bảo cho việc truyền số liệu không bị xung đột. Trên cơ sở đó người ta có các khái niệm:

Dải thông

Tốc độ mà bus có thể truyền dữ liệu từ thiết bị chủ tới thiết bị tớ gọi là *dải thông* hay *độ rộng dải* của bus (bandwidth) hoặc *năng suất truyền* của bus

(throughput). Đơn vị đo là MB/sec. Dải thông phụ thuộc vào tốc độ, độ rộng và giao thức của bus:

Dải thông của Bus = tần số làm việc của bus * độ rộng bus số liệu.

Mục đích phát triển của các hệ bus trên máy tính IBM PC và tương thích là các loại bus thế hệ sau luôn có dải thông lớn hơn thế hệ trước để đảm bảo truyền số liệu nhanh hơn.

Plug and play

Microsoft và Intel cộng tác với nhau để trang bị bus ISA thêm khả năng của cấu hình tự động. Đặc điểm này còn gọi là *cắm là chạy* (plug and play). PCI với cầu hình tự động có thể làm việc hoàn chỉnh chỉ sau khi card EISA và BIOS được trang bị cấu hình tự động. ISA và MCA đều đã được bổ sung đặc tính quan trọng này từ trước. Có 3 trường hợp liên quan tới Plug and Play như sau:

1. Cả BIOS trên board mẹ lẫn card bổ sung đều không phải là Plug and play. Trường hợp này đôi khi còn được gọi "cắm xong là cầu Chúa" (plug and pray), vì bạn sẽ gặp lỗi.
2. BIOS trên board mẹ được trang bị Plug and play nhưng card bổ sung thì không. Trường hợp này phần mềm cài đặt sẽ giúp bạn sắp xếp địa chỉ I/O, IRQ và các kênh DMA.
3. Cả BIOS trên board mẹ lẫn card bổ sung đều trang bị Plug and play. Trường hợp này, cấu hình tự động sẽ thực hiện mọi công việc. Nó sẽ tự sắp xếp địa chỉ I/O, IRQ và các kênh DMA mà không cần người dùng can thiệp.
4. Hot Plug and Play là khả năng bus hoàn toàn có thể nhận biết sự thay đổi (ngắt và kết nối ngoại vi đến bus) và cài đặt điều khiển cho ngoại vi ngay cả khi máy tính vẫn đang làm việc. Khả năng này cho phép máy tính có thể làm việc liên tục không phải dừng khi thay thế các ngoại vi.

Máy tính sẽ sử dụng vùng nhớ RAM khi thực hiện chương trình ứng dụng. Các file chạy thường có phần mở rộng *.COM hoặc *.EXE.

File có phần mở rộng *.COM có kích thước nhỏ hơn 64KB, để thực hiện một file *.com, hệ điều hành chỉ sử dụng 1 segment (64KB) bộ nhớ để thực hiện chương trình này. Khi đó HĐH sẽ khởi tại các đoạn trong segment đó như sau:

Các thanh ghi đoạn có cùng 1 giá trị: CS = DS = SS = ES

Vùng số liệu sẽ được khởi tạo từ giá trị thấp nhất của segment đó với địa chỉ là 0000 h cho đến 00FF h.

Vùng mã lệnh sẽ được bắt đầu từ ô nhớ 0100 h.

Vùng ngăn xếp sẽ được bắt đầu từ ô nhớ FFFE h.

Như vậy điểm vào của một chương trình *.com sẽ bắt đầu từ địa chỉ CS:0100, vùng số liệu sẽ bắt đầu từ địa chỉ DS:0000, vùng ngăn xếp sẽ bắt đầu từ địa chỉ SS:FFFE.

Để thực hiện 1 file *.EXE HĐH sử dụng nhiều segment, khi đó đoạn mã lệnh, đoạn số liệu, đoạn ngăn xếp và đoạn số liệu mở rộng được nằm trên các đoạn khác nhau CS ≠ DS ≠ SS ≠ ES.

Điểm vào của chương trình *.EXE có thể nằm ở bất cứ điểm nào trong đoạn mã lệnh.

Vùng số liệu sẽ bắt đầu từ địa chỉ DS:0000, vùng ngăn xếp sẽ bắt đầu từ địa chỉ SS:FFFE.

Khi thực hiện các lệnh asm, người ta thường dùng 1 số ngắt mềm để thực hiện một số yêu cầu. Ví dụ như ngắt int 21h (ngắt của DOS) với các giá trị ah = 01h, ah = 08h, ah =02h, ah = 4ch, ngắt int 10h (ngắt màn hình) và ngắt int 16h (ngắt bàn phím).

Để xem xét nội dung bên trong các thanh ghi của CPU, các ô nhớ, các sector hay các file trên đĩa người ta thường dùng chương trình debug. Debug là một chương trình tương đướng với 1 lệnh ngoại trú của HĐH ta có thể sử dung debug để chạy các đoạn chương trình ngắn, để xem và thay đổi nội dung của các thanh ghi bên

trong CPU và ô nhớ . Ví dụ như để kiểm tra sự có mặt của các cổng COM hay cổng LPT của máy tính ta có thể xem nội dung của vùng nhớ chứa tham số của HĐH tại các ô nhớ từ 00400h đến 0040f h bằng lệnh D.

D 0040:0000 l 10

Vào ra và quản lý quá trình vào ra.

Vào/ ra là quá trình trao đổi số liệu giữa CPU và các thiết bị ngoại vi thông qua bus. Trong các BVXL hiện nay có loại có hỗ trợ cho quá trình vào ra, có loại không hỗ trợ cho quá trình vào/ra.

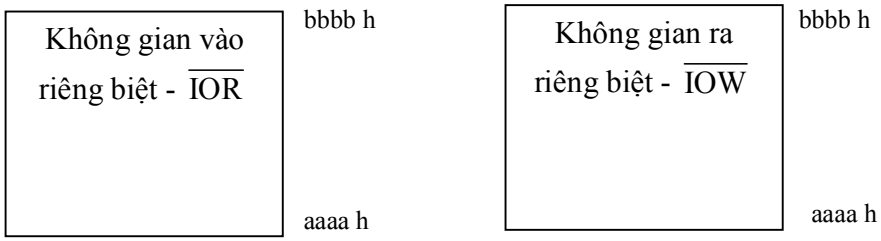
BVXL có hỗ trợ vào ra: Trong cấu trúc của CPU về phần mềm có các lệnh hỗ trợ vào ra và về phần cứng có các Modul đảm bảo cho quá trình vào ra bằng các xung trạng thái $M/\overline{IO}$ kết hợp với 2 tín hiệu điều khiển là $\overline{RD}$ và $\overline{RW}$ để tạo ra các tín hiệu cho phép máy tính có không gian vào/ra riêng biệt.

$M/\overline{IO}$	$\overline{RD}$	$\overline{WR}$		**Thao tác trên máy tính**
1	0	1	$\overline{MEMR}$	Đọc bộ nhớ
1	1	0	$\overline{MEMW}$	Ghi bộ nhớ
0	0	1	$\overline{IOR}$	Đọc cổng vào
0	1	0	$\overline{IOW}$	Ghi cổng ra

Khi tín hiệu trạng thái $M/\overline{IO}$ có mức logic "0" khi đó CPU sẽ thực hiện trao đổi số liệu với không gian vào/ra cụ thể: khi tín hiệu $\overline{IOR}$ có mức logic "0" máy tính sẽ đọc số liệu từ vùng không gian các cổng vào có địa chỉ được xác định bởi số lượng các bits địa chỉ tương ứng với chế độ làm việc ở chế độ vào/ra này. Khi tín hiệu $\overline{IOW}$ có mức logic "0" máy tính sẽ ghi số liệu đến vùng không gian các cổng ra có địa chỉ được xác định bởi số lượng các bits địa chỉ tương ứng với chế độ làm việc ở chế độ vào/ra này.

BVXL không hỗ trợ vào ra: Trong cấu trúc của không có các modul cho phép hỗ trợ vào ra. Cụ thể quá trình đọc/ghi của máy tính chỉ tồn tại hai tín hiệu điều khiển là $\overline{MEMR}$ và $\overline{MEMW}$. Khi đó người ta phải sử dụng 1 phần không gian

của bộ nhớ để làm không gian vào/ra cho các thiết bị ngoại vi



Sử dụng không gian vào/ra riêng biệt

Khi sử dụng không gian vào ra riêng biệt, máy tính sẽ tạo ra 1 vùng địa chỉ cho các cổng vào từ *aaaa* h đến *bbbb* h với tín hiệu điều khiển là $\overline{IOR}$, sử dụng lệnh IN và 1 vùng địa chỉ cho các cổng ra từ *aaaa* h đến *bbbb* h với tín hiệu điều khiển là $\overline{IOW}$, sử dụng lệnh OUT.

Các lệnh hỗ trợ vào ra trên máy tính IBM PC có hai chế độ địa chỉ:

Chế độ địa chỉ trực tiếp: không gian địa chỉ vào/ra được địa chỉ hóa bởi 8 bits địa chỉ từ A_0 đến A_7 ($2^8 = 256$ cổng vào và $2^8 = 256$ cổng ra – tương ứng với aaaa h = 0000h và bbbbh = 00FF h). Khi đó địa chỉ cổng được nằm trực tiếp ngay trong lệnh. Cụ thể:

Lệnh vào

Khi trao đổi số liệu 8 bits – Thanh ghi AL là thanh ghi thao tác (giao tiếp trên phần 62 chân của bus ISA)

IN AL, #địa chỉ cổng - Ví dụ IN AL, 55h

Khi trao đổi số liệu 16 bits – Thanh ghi AX là thanh ghi thao tác (giao tiếp trên phần 36 chân của bus ISA)

IN AX, #địa chỉ cổng – Ví dụ IN AX, 2Fh

Lệnh ra

Khi trao đổi số liệu 8 bits – Thanh ghi AL là thanh ghi thao tác (giao tiếp trên phần 62 chân của bus ISA)

OUT #địa chỉ cổng, AL - Ví dụ IN 55h, AL

Khi trao đổi số liệu 16 bits – Thanh ghi AX là thanh ghi thao tác (giao tiếp trên phần 36 chân của bus ISA)

OUT #địa chỉ cổng, AX – Ví dụ IN 2Fh, AX

Chế độ địa chỉ gián tiếp: không gian địa chỉ vào/ra được địa chỉ hóa bởi 16 bits địa chỉ từ A_0 đến A_{15} (2^{16} = 65536 cổng vào và 2^{16} = 65536 cổng ra – tương ứng với aaaa h = 0000h và bbbbh = FFFF h). Khi đó thanh ghi DX được sử dụng để chứa địa chỉ của cổng vào/ra. Cụ thể:

Lệnh vào

Khi trao đổi số liệu 8 bits – Thanh ghi AL là thanh ghi thao tác (giao tiếp trên phần 62 chân của bus ISA)

```
MOV     DX, #địa chỉ cổng
IN      AL, DX
Ví dụ
MOV     DX,03F8h    ; nạp địa chỉ cổng vào lên thanh ghi DX
IN      AL, DX      ; đọc số liệu từ cổng có địa chỉ 03F8h
```

Khi trao đổi số liệu 16 bits – Thanh ghi AX là thanh ghi thao tác (giao tiếp trên phần 36 chân của bus ISA)

```
MOV     DX, #địa chỉ cổng
IN      AX, DX
Ví dụ
MOV     DX, 012Ah   ; nạp địa chỉ cổng vào lên thanh ghi DX
IN      AX, DX      ; đọc số liệu từ cổng có địa chỉ 012Ah
```

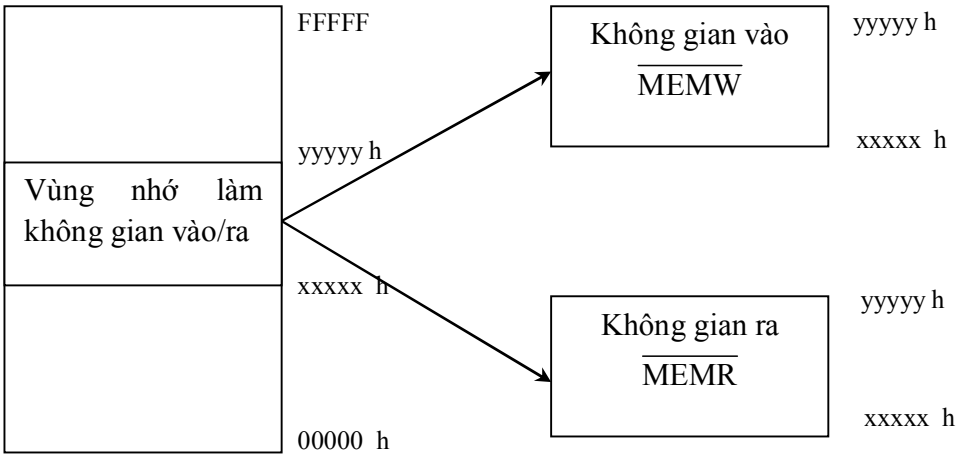
Lệnh ra

Khi trao đổi số liệu 8 bits – Thanh ghi AL là thanh ghi thao tác (giao tiếp trên phần 62 chân của bus ISA)

```
MOV     DX, 03F8h   ; nạp địa chỉ cổng vào lên thanh ghi DX
OUT     DX, AL      ; ghi số liệu ra cổng có địa chỉ 03F8h
```

Khi trao đổi số liệu 16 bits – Thanh ghi AX là thanh ghi thao tác (giao tiếp trên phần 36 chân của bus ISA)

```
MOV     DX, 012Ah   ; nạp địa chỉ cổng vào lên thanh ghi DX
OUT     DX, AX      ; ghi số liệu ra cổng có địa chỉ 012Ah
```



Sử dụng 1 phần không gian bộ nhớ làm không gian vào/ra

Khi sử dụng 1 phần không gian bộ nhớ làm không gian vào ra, máy tính sẽ sử dụng 1 vùng không gian bộ nhớ để làm không gian vào/ra và tạo ra vùng không gian vào có địa chỉ từ *xxxxx* h đến *yyyyy* h với tín hiệu điều khiển là $\overline{\text{MEMR}}$ sử dụng lệnh chuyển số liệu từ một cổng vào có địa chỉ ghi trong lệnh tới thanh ghi bên trong của BVXL. Vùng không gian ra cũng có địa chỉ từ *xxxxx* h đến *yyyyy* h với tín hiệu điều khiển là $\overline{\text{MEMW}}$ sử dụng lệnh chuyển số liệu từ thanh ghi bên trong của BVXL đến một cổng ra có địa chỉ ghi trong lệnh. Vùng không gian địa chỉ vào/ra là một vùng bộ nhớ được biểu diễn bởi 20 bits địa chỉ do đó cần phải có thủ tục khai báo địa chỉ đoạn cho các lệnh vào/ra. Ví dụ: Cần đọc số liệu từ địa chỉ cổng 35000h vào máy tính, ta cần các bước sau: Địa chỉ vật lý 35000h có thể được biểu diễn bằng 1 địa chi logic là 3000:5000 (DS:offset)

```
MOV     AX, 3000h   ; khởi tạo thanh ghi đoạn số liệu
MOV     DS, AX
MOV     AL,[5000]   ; đọc số liệu từ cổng có địa chỉ 35000h.
```

Cần ghi số liệu ra cổng có địa chỉ 35000h

```
MOV     AX,3000h    ; khởi tạo thanh ghi đoạn số liệu
```

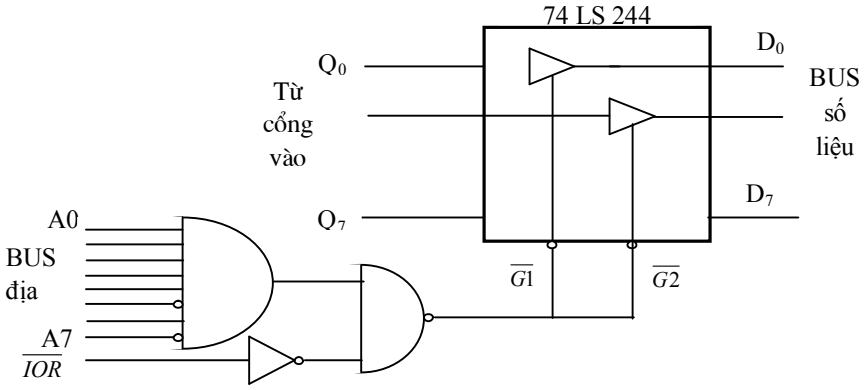
```
MOV     DS, AX
MOV     [5000], AL   ; ghi số liệu ra cổng 35000h.
```

Lúc này số liệu để ghi ra cổng ra hoặc đọc từ cổng vào không nhất thiết phải nằm trong thanh ghi tích lũy mà có thể là 1 trong bất kỳ các thanh ghi đa năng nào của CPU như AL, BL, CL, DL (trao đổi số liệu 8 bits) hay AX, BX, CX, DX (trao đổi số liệu 16 bits)

Nối ghép vào/ra

Để nối ghép các cổng vào/ra với bus có thể dùng các vi mạch đơn giản như vi mạch 74LS244 cho các cổng vào, 74LS374 cho các cổng ra hay sử dụng vi mạch lập trình vào/ra song song như 8255A.

Nối ghép cổng vào sử dụng vi mạch 74LS244

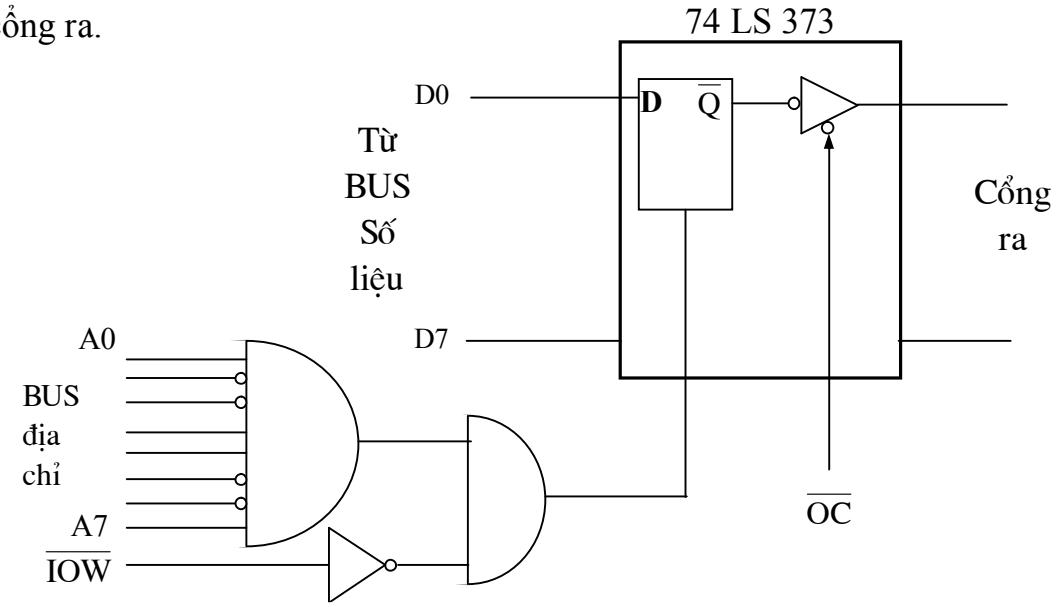


Thực hiện lệnh IN AL, 5Fh

Vi mạch 74LS244 là bộ đệm 8 cổng 1 chiều có 2 chân điều khiển $\overline{G_1}$, $\overline{G_2}$, tám đầu vào của bộ đệm được nối đến các cổng vào và 8 đầu ra của bộ đệm được nối đến các chân D_0 đến D_7 của bus số liệu. Mạch giải mã bao gồm các tín hiệu địa chỉ từ A_0 đến A_7 kết hợp với tín hiệu $\overline{\text{IOR}}$ cho phép đọc số liệu từ cổng qua bộ đệm 74LS244 đến BVXL thông qua bus số liệu.

Nối ghép cổng ra sử dụng vi mạch 74LS374:

Vi mạch 74LS374 là mạch chốt 8 cổng có chân điều khiển $\overline{OC}$, bao gồm 8 triger D, tám đầu vào của mạch chốt được nối đến các chân D_0 đến D_7 của bus số liệu, 8 đầu ra của mạch chốt được nối đến các cổng ra. Mạch giải mã bao gồm các tín hiệu địa chỉ từ A_0 đến A_7 kết hợp với tín hiệu $\overline{IOW}$ cho phép ghi số liệu từ thanh ghi Al của CPU qua bus đến các đầu vào của mạch chốt 74LS374 đến các cổng ra.



Thực hiện lệnh OUT 99h, Al

Quản lý vào/ra

Máy tính thường quản lý các thiết bị Vào/Ra theo các phương pháp sau:

Xử lý tuần tự: Máy tính thực hiện hỏi lần lượt trạng thái của các cổng Vào/Ra, khi một thiết bị ngoại vi nào đó được yêu cầu phục vụ, trạng thái cổng sẽ thay đổi, máy tính sẽ phục vụ quá trình vào/Ra đó.

Xử lý theo ngắt: Khi có một ngoại vi yêu cầu được được phục vụ, nó sẽ phát tín hiệu yêu cầu ngắt (IRQ) đến BVXL, BVXL sẽ dừng tất cả mọi công việc để phục vụ yêu cầu này. Nếu đồng thời có nhiều ngoại vi cùng yêu cầu được phục vụ. Máy tính sẽ phục vụ chúng lần lượt theo thứ tự ưu tiên.

Trong máy tính IBM PC, tất cả các ngoại vi tiêu chuẩn được quản lý bằng phương pháp ngắt. Yêu cầu ngắt trong máy tính bao gồm: Ngắt cứng, ngắt mềm.

Ngắt mềm được sinh ra trong quá trình thực hiện chương trình, khi mà chương trình cần thực hiện một chức năng nào đó liên quan đến Hệ điều hành. Ngắt mềm tương ứng với một lệnh ASM – int #số hiệu ngắt.

Ngắt cứng là đáp ứng tức thời của máy tính với một yêu cầu được phục vụ của một thiết bị ngoại vi tiêu chuẩn gắn trên máy tính như bàn phím, ổ đĩa, máy in..... Các ngắt này được linh kiện 8259A quản lý.

Máy tính IBM PC XT có 8 ngoại vi tiêu chuẩn gắn với các ngắt cứng là:

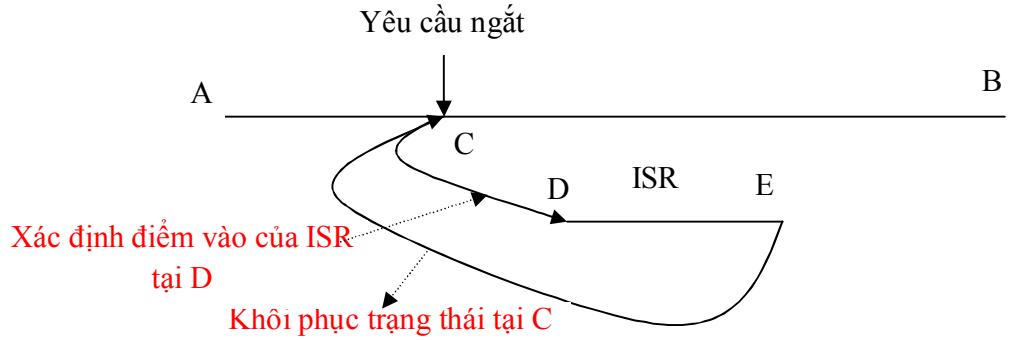
IRQ0 Timer
IRQ1 Keyboard
IRQ2 Dùng trong máy AT
IRQ3 Serial port 2
IRQ4 Serial port 1
IRQ5 Hardisk
IRQ6 Floppy disk
IRQ7 Parallel port 1

Máy tính IBM PC cho phép đáp ứng được 256 ngắt trong đó có 15 ngắt cứng còn lại là ngắt mềm và dự trữ. Mỗi ngắt cứng có một chương trình kèm theo nó gọi là chương trình phục vụ ngắt (ISR). Mỗi ngắt trong 256 ngắt trên máy tính được đánh số thứ tự theo số hexa từ 00 h – FF h được gọi là số hiệu ngắt, các số hiệu ngắt từ 00 h – FF h có mức ưu tiên giảm dần: Ngắt có số hiệu là 00 h có mức ưu tiên cao nhất và ngắt có số hiệu là FF h có mức ưu tiên thấp nhất.

Quá trình đáp ứng của máy tính với 1 yêu cầu ngắt: Khi có 1 hay một vài yêu cầu ngắt được sinh ra. CPU sẽ thực hiện các thao tác sau:

Cất toàn bộ trạng thái của máy tính tại thời điểm hiện tại vào 1 vùng nhớ gọi là ngăn xếp. Trạng thái đó sẽ là giá trị của thanh ghi cờ trạng thái (RF – 2 bytes) và địa chỉ của lệnh tiếp theo sẽ được thực hiện được nằm trong CS:IP.

Xác định số hiệu ngắt, qua đó xác định được điểm vào của chương trình phục vụ ngắt, nạp điểm vào đó vào thanh ghi CS và IP để tiến hành thực hiện chương trình phục vụ ngắt. Sau khi thực hiện xong chương trình phục vụ ngắt, trạng thái của máy tính tại thời điểm diễn ra quá trình phục vụ ngắt được khôi phục và máy tính lại tiếp tục thực hiện công việc mà trước đó nó đang làm.



Để xác định điểm vào của một chương trình phục vụ ngắt, máy tính sử dụng bảng vector ngắt, trong đó ghi sẵn điểm vào của 256 ngắt. Điểm vào của chương trình phục vụ ngắt là ô nhớ có địa chỉ logic (Đoạn: Độ lệch) là 2 thanh ghi 16 bits, như vậy để lưu trữ được địa chỉ điểm vào của một chương trình phục vụ ngắt ta cần 4 bytes gọi là 1 phần tử của vector ngắt. Với 256 chương trình ta cần 256 phần tử và chiếm 1024 bytes bộ nhớ. Bảng vetor ngắt của máy tính IBM PC được bố trí nằm trong vùng nhớ có địa chỉ từ 00000 h – 003FF h tại vùng thấp nhất trên bản đồ bộ nhớ. Bốn ô nhớ 00000 – 00003 chứa điểm vào của chương trình phục vụ ngắt có số hiệu ngắt 00, bốn ô nhớ tiếp 00004 – 00007 chứa điểm vào của chương trình phục vụ ngắt có số hiệu ngắt 01, bốn ô nhớ tiếp 00008 – 0000B chứa điểm vào của chương trình phục vụ ngắt có số hiệu ngắt 02 .v.v. Như vậy ô nhớ đầu tiên chứa điểm vào của 1 chương trình phục vụ ngắt bao giờ

cũng gấp 4 lần số hiệu ngắt, do đó chỉ cần xác định được số hiệu ngắt là ta có thể xác định được điểm vào của chương trình phục vụ ngắt. Đối với ngắt mềm, số hiệu ngắt được nằm ngay trong lệnh INT, ví dụ ta có lệnh INT 21h, lệnh INT 10h thì 21h và 10h chính là số hiệu ngắt của yêu cầu ngắt. Với ngắt cứng để gán số hiệu ngắt cho các tín hiệu yêu cầu ngắt IRQ, người ta sử dụng việc lập trình vào thanh ghi ICW2 của vi mạch điều khiển ngắt 8259 để quản lý các ngắt cứng.

	Ô nhớ	**Đ/c ô nhớ**
Chứa địa chỉ đầu của chương trình phục vụ ngắt có số hiệu ngắt FF		003FF
		003FE
		003FD
		003FC = 0FF *4
……	……..	…….
Chứa địa chỉ điểm vào của chương trình phục vụ ngắt có số hiệu ngắt 02		0000B
		0000A
		00009
		00008 = 02 *4
Chứa địa chỉ đầu của chương trình phục vụ ngắt có số hiệu ngắt 01		00007
		00006
		00005
		00004 = 01 *4
Chứa địa chỉ đầu của chương trình phục vụ ngắt có số hiệu ngắt 00		00003
		00002
		00001
		00000

Để xác định điểm kết thúc của một chương trình phục vụ ngắt, người ta quy định lệnh cuối cùng của chương trình phục vụ ngắt phải là IRET.

Để cất giữ và khôi phục trạng thái của máy tính tại thời điểm diễn ra quá trình phục vụ ngắt, máy tính sử dụng ngăn xếp.

Ngăn xếp là 1 segment (đoạn bộ nhớ) được sử dụng để cất giữa trạng thái của máy tính khi phục vụ các thao tác ngắt và các lệnh PUSH, POP của ASM. Địa chỉ của ô nhớ nơi sẽ diễn ra quá trình hồi phục và cất giữ được gọi là đỉnh của ngăn xếp SP. Địa chỉ vật lý đỉnh của ngăn xếp là SS:SP. Khi máy tính bắt đầu khởi động Hệ điều hành sẽ lựa chọn 1 segment để làm ngăn xếp và khởi

động giá trị của SP = FFFE. Khi diễn ra quá trình ngắt, trạng thái của máy tính sẽ được cất vào ngăn xếp và đỉnh của ngăn xếp sẽ liên tục thay đổi để chỉ đến ô nhớ là nơi sẽ bắt đầu diễn ra quá trình cất giữ và hồi phục như sau:

Quá trình cất giữ		Ô nhớ		Quá trình hồi phục	Địa chỉ ô nhớ
					FFFF
Giá trị ban đầu của SP	→	Cất giữ giá trị của RF	←	SP = SP +2	FFFE
					FFED
SP = SP -2	→	Cất giữ giá trị của CS tại lúc ngắt	←	SP = SP +2	FFFC
					FFFB
SP = SP -2	→	Cất giữ giá trị của IP tại lúc ngắt	←	SP = SP +2	FFFA
					FFF9
SP = SP -2	→				FFF8
					FFF7
					FFF6
					FFF5

Ta thấy trong quá trình cất giữ, thanh ghi con trỏ lệnh (IP) được cất giữ sau cùng nhưng lại được lấy ra đầu tiên trong quá trình khôi phục. Thanh ghi RF được cất đầu tiên nhưng lại được lấy ra sau cùng, do đó nguyên tắc làm việc của ngăn xếp được gọi là quá trình LIFO (last in – first out).

Tóm tắt quá trình xử lý ngắt:

1. Cất thanh ghi cờ (FR) vào ngăn xếp và giảm SP đi 2 vì FR là thanh ghi 2 byte.

2. Xoá cả hai cờ cho phép ngắt IF và cờ bẫy TF (IF=0 và TF=0). Khi đó sẽ che các yêu cầu ngắt khác đưa đến chân INTR và huỷ bỏ chế độ chạy từng lệnh trong khi CPU đang thực hiện chương trình con phục vụ ngắt ISR. Tuỳ thuộc vào thủ tục ngắt mà người lập trình có thể huỷ che chân INTR bằng lệnh STI.

3. Cất CS hiện hành vào ngăn xếp và giảm SP đi 2.

4. Cất IP hiện hành vào ngăn xếp và giảm SP đi 2.

5. Nhân số hiệu INT với 4 để xác định địa chỉ vật lý của ngắt trong bảng vec tơ ngắt, qua đó có được CS và IP của chương trình phục vụ ngắt ISR.

6. Với CS:IP mới, CPU bắt đầu nhận và thực hiện các lệnh của chương trình phục vụ ngắt ISR.

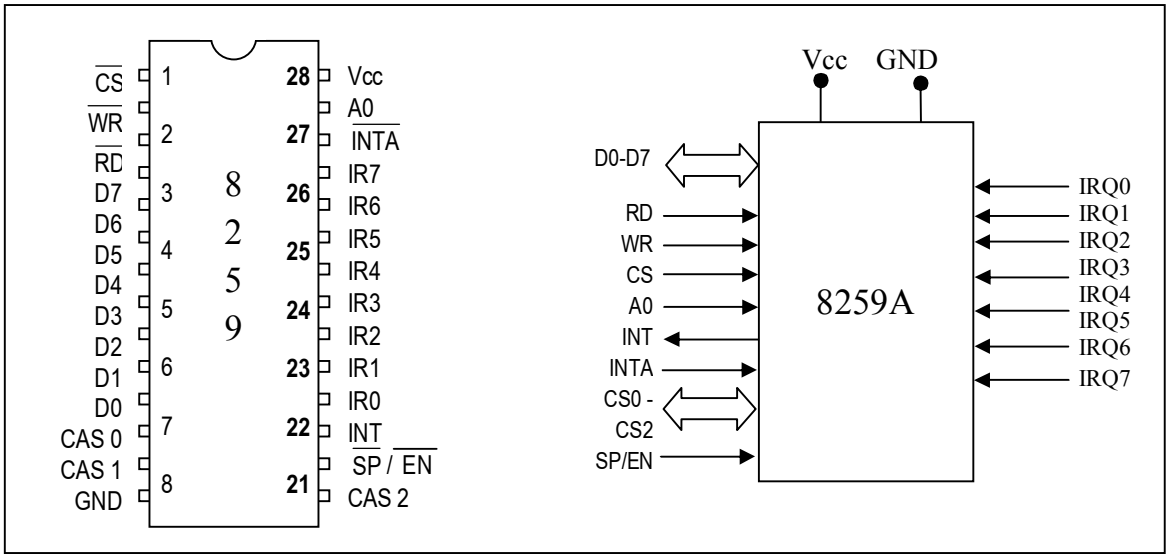
7. Lệnh cuối cùng của ISR là lệnh IRET. Lệnh này thông báo để CPU tải lại giá trị IP, CS và FR từ ngăn xếp và nhờ vậy CPU có thể chạy tiếp chương trình tại nơi nó đã bị ngắt.

Vi mạch điều khiển ngắt cứng 8259

Vi mạch điều khiển ngắt cứng dùng trong máy tính IBM PC cho phép quản lý được 8 ngắt cứng tương ứng với 7 ngoại vi tiêu chuẩn là:

IRQ0 Timer
IRQ1 Keyboard
IRQ2 Dùng trong máy AT
IRQ3 Serial port 2
IRQ4 Serial port 1
IRQ5 Hardisk
IRQ6 Floppy disk
IRQ7 Parallel port 1

Vi mạch 8259 có 8 đầu vào yêu cầu ngắt IRQ0 – IRQ7 với IRQ0 có mức ưu tiên cao nhất và IRQ7 có mức ưu tiên thấp nhất.



Để lập trình cho 8259 A, cần thiết lập các từ điều khiển của 8259 để thiết lập các chế độ làm việc cho 8259 như chế độ làm việc độc lập, chế độ nối tầng và thiết lập các mức ưu tiên cho các tín hiệu yêu cầu ngắt IRQ.

8259 có 4 từ điều khiển là ICW1, ICW2, ICW3 và ICW4. Việc lựa chọn một trong 4 từ điều khiển là sử dụng $\overline{CS}$ kết hợp với A0.

$\overline{CS}$	A0	Từ lệnh khởi tạo
0	0	**ICW1**
0	1	ICW2, ICW3 và ICW4

Trong đó thanh ghi ICW2 là thanh ghi cho phép xác định số hiệu ngắt

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	T7	T6	T5	T4	T3	T2	T1	T0

Chức năng của ICW2 là gán các số hiệu ngắt cho IRQ0 – IRQ7. Các bits D2 D1 D0 thay đổi từ 000 – 111 cùng với D3 – D0 để tạo ra các số dạng INT nn để gán cho IRQ0 – IRQ7 tương ứng. Nhìn vào bảng Vector ngắt ta thấy IRQ0 – IRQ7 sẽ lần lượt tương ứng với các số hiệu ngắt INT 08 H – INT 0F H như vậy ta có:

Với IRQ0: ICW2 có giá trị tương ứng với 08 H.

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	1	0	0	0

Với IRQ1: ICW2 có giá trị tương ứng với 09 H

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	1	0	0	1

Và với IRQ7: ICW2 có giá trị tương ứng với 0F H

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	1	1	1	1

Sau khi ICW1, ICW2 và ICW4 lần lượt phát ra và khởi tạo 8259, BVXL 80x86 cũng sẵn sàng nhận nhận các ngắt cứng từ các chân IRQ0 – IRQ7 của 8259. Có thể lập trình để che hoặc thay đổi mức ưu tiên của bất cứ một chân IRQ nào.

8259 có ba từ lệnh là OCW1, OCW2, và OCW3. Bình thường IRQ0 được thiết lập mức ưu tiên cao nhất và IRQ7 có mức ưu tiên thấp nhất.

Máy tính IBM PC XT sử dụng 1 vi mạch để điều khiển được 8 ngắt cứng, từ máy tính AT sử dụng thêm 1 vi mạch nữa để mở rộng ngắt cứng lên 15.

Máy IBM PC AT sử dụng 2 vi mạch 8259, vi mạch thứ nhất làm việc ở chế độ chủ và vi mạch thứ 2 được cấy thêm làm việc ở chế độ tớ và được ghép với vi mạch chủ qua chân IRQ2. Vi mạch chủ và tớ trao đổi thông tin qua chân IRQ2, INT, CAS0, CAS1 và CAS2.

Trên máy XT tín hiệu IRQ2 được đưa ra khe cắm rãnh ISA, trên máy AT, IRQ2 được dùng cho 8259 thứ hai. Lúc này khe cắm trên rãnh ISA được thay bằng IRQ9 (trên 8259 thứ hai).

Trên máy AT mức ưu tiên có thứ tự như sau: IRQ0, IRQ1, IRQ8 – IRQ15, IRQ3 – IRQ7.

Mức kích phát ngắt theo mức được sử dụng trên các Bus EISA và MCA trên máy AT để tránh phải ngắt nhầm do nhiễu tác động vào các đường IRQ khi máy tính làm việc ở tần số cao.

Truy cập trực tiếp bộ nhớ (DMA)

Khi máy tính cần phải chuyển một số lượng lớn số liệu giữa thiết bị ngoại vi và bộ nhớ. Nếu dùng CPU để thực hiện thì đầu tiên CPU nhận thông tin rồi sau đó mới chuyển đến nơi nhận. Quá trình nhận và giải mã lệnh còn cần thêm các thông tin bổ xung do đó công việc này sẽ rất chậm. Vì vậy Intel thiết kế bộ điều khiển và truy cập trực tiếp 8237 với chức năng bỏ qua CPU và truyền số liệu trực tiếp giữa bộ nhớ và ngoại vi, nhờ vậy mà làm cho quá trình sẽ nhanh lên nhiều.

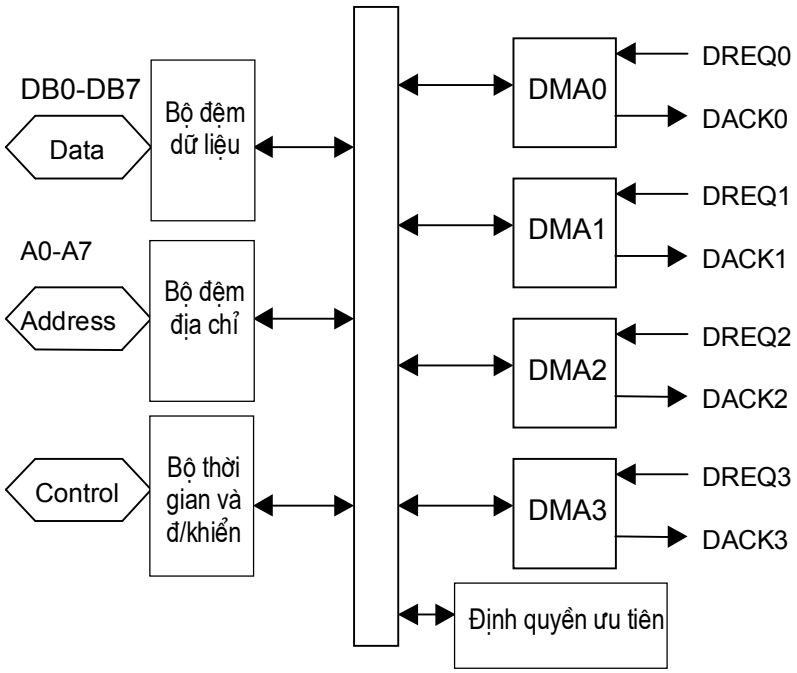
Ví dụ: Để truyền byte số liệu từ ngoại vi và bộ nhớ, nếu thông qua CPU thì phải mất 39 chu kỳ đồng hồ, nhưng dùng 8237 thì chỉ mất 4 chu kỳ đồng hồ

			Số chu kỳ đồng hồ
LAP:	MOV	AL, [SI]	10
	OUT	PORT,AL	10
	INC	SI	2
	LOOP	LAP	17
		; Số chu kỳ :	**39**

Như vậy khi sử dụng DMA thì lúc đó BUS hệ thống không còn do CPU làm chủ nữa mà do bộ điều khiển thâm nhập trực tiếp 8237 làm chủ.

Khác với quá trình truyền số liệu qua CPU, DMA chỉ truyền số liệu mà không có quá trình giải mã và thực hiện lệnh và khi CPU nhận tín hiệu HOLD nó sẽ kết thúc chu kỳ BUS hiện hành để trao quyền điều hành BUS cho 8237 khác với ngắt cứng CPU cần kết thúc lệnh hiện hành trước khi phát tín hiệu INTA.

Vi mạch Intel 8237 bao gồm 4 kênh DMA, hoạt động độc lập, mỗi kênh có một cặp tín hiệu là DREQ và DACK. Tín hiệu HRQ được nối đến chân HOLD của CPU để truyền tín hiệu yêu cầu được điều khiển BUS và chân HLDA được nối với HLDA của CPU để nhận tín hiệu cho phép DMA từ CPU.



SƠ ĐỒ KHỐI ĐƠN GIẢN CỦA 8237

Bốn kênh DMA được định các mức ưu tiên khác nhau bằng lập trình. Mỗi kênh DMA trước khi sử dụng phải được khởi tạo để xác định địa chỉ và kích trước của khối số liệu. Quá trình khởi tạo là ghi vào từng kênh các nội dung:

Địa chỉ đầu của khối số liệu cần truyền đi (Địa chỉ cơ sở)

Số lượng byte cần truyền (Số từ)

Để đặt chế độ làm việc cho các kênh của 8237 cần phải lập trình bằng cách ghi giá trị vào các thanh ghi của 8237 qua bốn chân địa chỉ A0 - A3. Mỗi kênh DMA có địa chỉ cơ sở riêng và có số từ riêng nên 8237 cần 8 cổng. Sau đây là địa chỉ nội bộ của 8237 trong quá trình ghi địa chỉ truyền và đếm.

CH		R/W	$\overline{CS}$	$\overline{IOR}$	$\overline{IOW}$	A3	A2	A1	A0
0	Địa chỉ hiện hành và cơ sở	W	0	1	0	0	0	0	0
	Địa chỉ hiện hành	R	0	0	1	0	0	0	0
	Đếm từ hiện hành và cơ sở	W	0	1	0	0	0	0	1

	Đếm từ hiện hành	R	0	0	1	0	0	0	1
1	Địa chỉ hiện hành và cơ sở	W	0	1	0	0	0	1	0
	Địa chỉ hiện hành	R	0	0	1	0	0	1	0
	Đếm từ hiện hành và cơ sở	W	0	1	0	0	0	1	1
	Đếm từ hiện hành	R	0	0	1	0	0	1	1
2	Địa chỉ hiện hành và cơ sở	W	0	1	0	0	1	0	0
	Địa chỉ hiện hành	R	0	0	1	0	1	0	0
	Đếm từ hiện hành và cơ sở	W	0	1	0	0	1	0	1
	Đếm từ hiện hành	R	0	0	1	0	1	0	1
3	Địa chỉ hiện hành và cơ sở	W	0	1	0	0	1	1	0
	Địa chỉ hiện hành	R	0	0	1	0	1	1	0
	Đếm từ hiện hành và cơ sở	W	0	1	0	0	1	1	1
	Đếm từ hiện hành	R	0	0	1	0	1	1	1

Để lập trình cho 8237 truyền số liệu cần có:

Địa chỉ byte đầu tiên của khối số liệu cần truyền

Số lượng byte số liệu cần truyền.

Để xác định địa chỉ byte đầu tiên của khối số liệu cần truyền đây là một số 16 bits trong khi bus số liệu là 8 bits cho nên cần phải gửi 2 byte số liệu ở cùng một địa chỉ. Để xác định số lượng thông tin cần truyền, cần phải ghi thông tin vào thanh ghi đếm. Số đếm cực đại của thanh ghi đếm là FFFF h do đó cũng cần phải 2 lần ghi liên tiếp ra cùng một địa chỉ.

Ghép nối vi mạch 8237 trên máy tính IBM PC/XT

Các chân địa chỉ của 8237 từ A0 – A7, bốn đường từ A0 – A3 là các đường hai chiều cùng với $\overline{CS}$ cho phép 8237 chọn một trong số 16 thanh ghi. Trong đó các địa chỉ cổng từ x0H – x7H được gán cho 4 kênh DMA và các địa chỉ từ x8H – xFH được gán cho các thanh ghi điều khiển được tất cả các kênh sử dụng.

Khi thực hiện DMA bộ điều khiển 8237 sẽ nắm quyền điều hành Bus do đó nó cũng cần phải có đầy đủ Bus điều khiển, Bus số liệu và Bus địa chỉ.

8237 có Bus số liệu hai chiều 8 bits từ D0 – D7 được nối vào Bus hệ thống.

8237 cũng cung cấp đủ 4 đường điều khiển $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$ và $\overline{MEMW}$.

Với Bus địa chỉ chỉ có 8 bits từ A0 – A7 nên khi cần truyền khối số liệu có kích thước lớn phải cần thêm các đường địa chỉ từ A8 – A15, 8237 sẽ dùng 8 bits số liệu D0 – D7 để tạo phần cao của địa chỉ và dùng một mạch chốt 74LS373. Khi đó tín hiệu ADSTB (Address strobe) có vai trò như tín hiệu ALE ở mạch chốt địa chỉ khi nối ghép CPU với Bus địa chỉ..

CPU chỉ sử dụng BUS khi 8237 không hoạt động. Tín hiệu AEN đảm bảo điều kiện này:

AEN = 0: CPU 80x86 điều khiển bus hệ thống.

AEN = 1: Vi mạch 8237 điều khiển bus hệ thống.

Các chân còn lại của 8237:

RESET: Là đầu vào được nối tới tín hiệu RESET của 8284.

READY là chân vào từ RĐYMA của mạch phát trạng thái đợi với mục đích kéo dài chu kỳ bộ nhớ của DMA

CLK được nối đến chân CLK của 8284 và có tần số làm việc 4,7 Mhz.

HOLD và HLDA được nối với các chân HOLD và HLDA của 80x86.

EOP được đảo pha thành TC. Tín hiệu này được kích hoạt mỗi khi thanh ghi đếm của một trong 4 kênh DMA bằng 0.

DREQ0 và DACK0 là các tín hiệu của kênh 0 được dùng để làm tươi DRAM.

DREQ1 – DREQ3 và DACK0 – DACK3 là các tín hiệu của kênh 1 đến 3 được bố trí trên rãnh ISA.

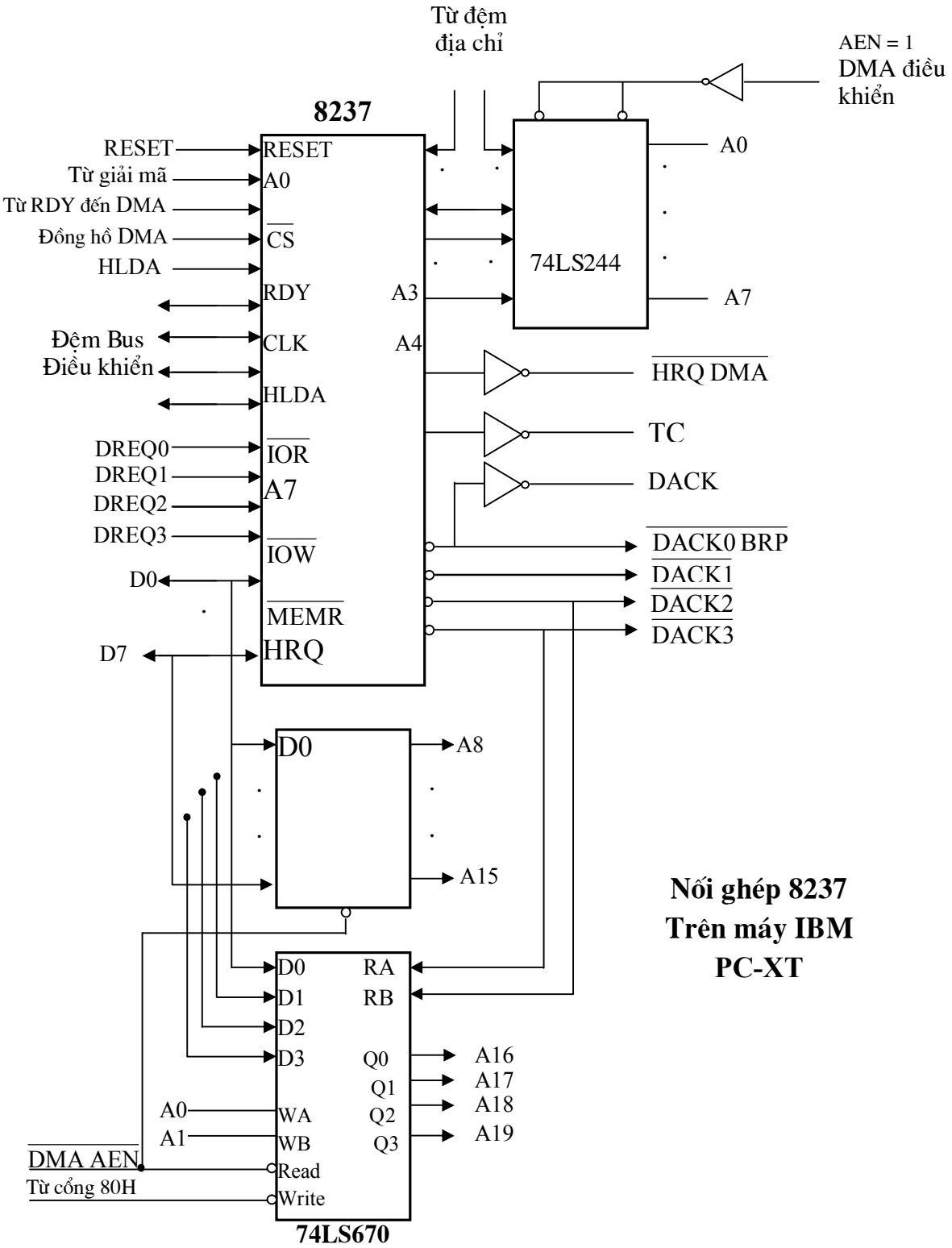
Trên máy tính IBM PC – XT các kênh DMA được sử dụng như sau:

Kênh 0 được dùng làm tươi DRAM. Trong các máy PC/AT không còn chức năng này.

Kênh 1 không được sử dụng, người ta dùng kênh này cho mạng.

Kênh 2 dùng cho bộ điều khiển đĩa mềm.

Kênh 3 dùng cho bộ điều khiển đĩa cứng.



Nối ghép 8237
Trên máy IBM
PC-XT

DMA ở máy tính IBM PC AT 80x86

Trên máy IBM PC XT có 3 kênh DMA được đưa ra trên rãnh ISA. Tất cả các kênh này là DMA 8 bits. Từ máy IBM PC AT dùng BVXL 08286 trở đi có bổ xung thêm vi mạch 8237 thứ hai để thực hiện DAM truyền số liệu 16 bits.

Vi mạch 8237 #1.

DREQ1, DREQ2, DREQ3 của 8237 thứ nhất thực hiện chức năng như trên máy XT (Thực hiện DMA 8 bits cho mạch vào/ra và với bộ nhớ 16 MB của máy AT). Máy tính AT có mạch làm tươi DRAM riêng mà không sử dụng kênh 0 của 8237#1. và DREQ0 và DACK0 được cung cấp trên phần 36 chân mở rộng của ISA Bus .

Các kênh 0 ,1, 2, 3 được dùng để truyền số liệu 8 bits giữa I/O và bộ nhớ.

Thanh ghi đếm có độ dài 16 bits do đó các kênh 0 ,1, 2, 3 có thể truyền được khối số liệu có độ dài đến 64KB.

Vi mạch 8237#2 được dùng làm mạch chủ và kênh 0 được sử dụng để nối tầng với 8237#1. Ba kênh còn lại DREQ5 - DREQ7 và DACK 5 – DACK7 được bố trí ở phần 36 chân của Bus ISA. Ba kênh này được dùng để truyền số liệu 16 bits.

Các kênh DMA 5, 6, 7 của 8237#2 được dùng để truyền số liệu 16 bits giữa bộ nhớ 16 MB và I/O.

Với thanh ghi đếm là thanh ghi 16 bits nên với số liệu 16 bits mỗi kênh DMA có thể truyền được 65536 từ hay 128 KByte giữa I/O và bộ nhớ.

Địa chỉ bộ nhớ của khối số liệu truyền DMA là địa chỉ chẵn.

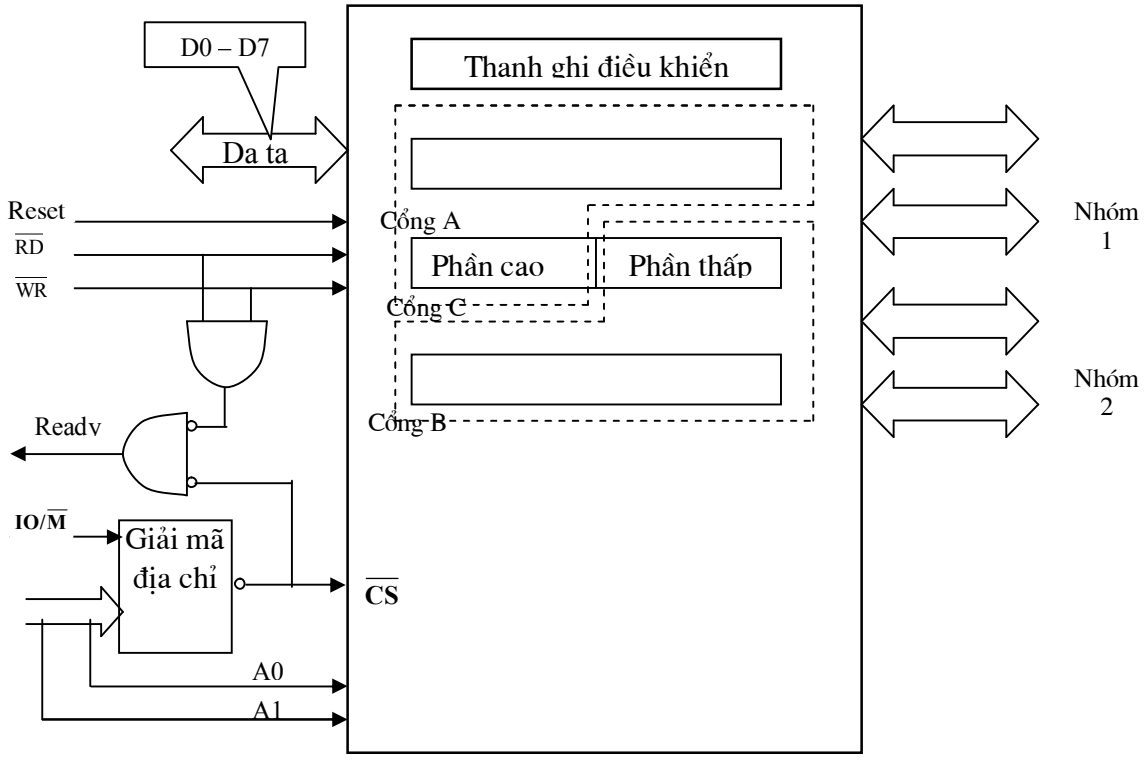
Các kênh 5, 6, 7 truyền số liệu sang bộ nhớ hệ thống 16 Mb theo khối, các khói này có kích thước tối đa là 128 KByte

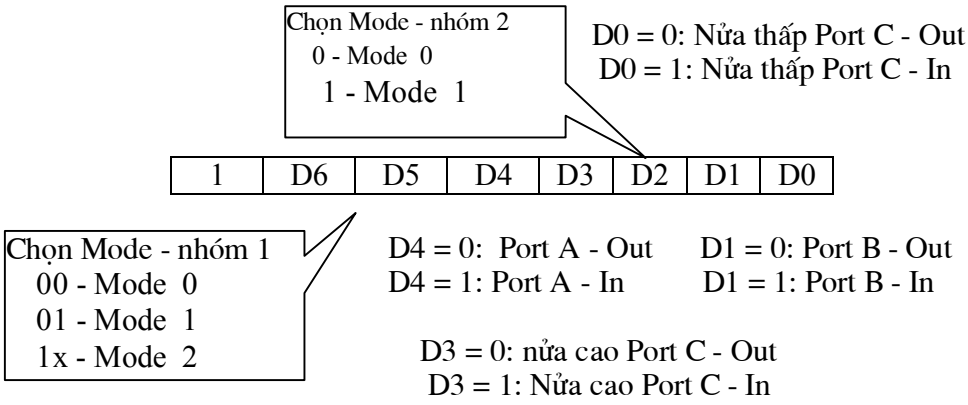
Mạch Vào/Ra song song.

Vào ra song song là với mỗi lần thực hiện lệnh IN hoặc lệnh OUT thì đồng thời cả 8 bits hay 16 bits số liệu được chuyển từ bus đến ngoại vi hay ngược lại. Ưu điểm của truyền song song là đảm bảo tốc độ truyền cao khi số bits được truyền trong 1 lệnh càng lớn (dải thông của bus phụ thuộc vào độ rộng của bus số liệu).

Nhược điểm của phương pháp truyền này là khi truyền với khoảng cách xa sẽ gặp khó khăn vì số lượng dây dẫn lớn và nhiễu tác động lên đường truyền gây ảnh hưởng tới chất lượng truyền số liệu.

Máy tính IBM sử dụng 2 vi mạch điều khiển Vào/Ra song song 8255-A để thực hiện: Ghép nối máy tính với máy in (nằm trên Card Vào/Ra). Một (nằm trên Mother Board) thực hiện nối ghép bàn phím, mạch điều khiển thời gian và kiểm chẵn lẻ RAM. Sơ đồ cấu trúc của vi mạch 8255 như sau:





Định nghĩa thanh ghi điều khiển của 8255A

Vi mạch 8255 trên máy tính IBM PC.

Trên máy tính IBM PC sử dụng 2 vi mạch 8255, một nằm trên MotherBoard với địa chỉ giải mã là 60H, cả 2 nhóm đều làm việc ở Mode 0 có chức năng như sau:

PA0 PA1 PA2 PA3 PA4 PA5 PA6 PA7	Đọc mã quét bàn phím	PB0 PB1 PB2 PB3 PB4 PB5 PB6 PB7	Mở loa Time 2 Mở dữ liệu cho loa Không sử dụng Không sử dụng Kiểm tra chẵn lẻ DRAM Kiểm tra kênh I/O Không sử dụng Không sử dụng	PC0 PC1 PC2 PC3 PC4 PC5 PC6 PC7	Không sử dụng 1: có đồng xử lý 0: có đĩa Không sử dụng Không sử dụng Output kênh time2 Kiểm tra kênh I/O Kiểm tra chẵn lẻ DRAM
Cổng A (vào) địa chỉ 60H		Cổng B (ra) địa chỉ 61H		Cổng C (vào) địa chỉ 62H	

Vi mạch 8255 thứ 2 nằm trên Card I/O cho phép nối ra cổng LPT với địa chỉ giải mã là 0378 h. Do cổng LPT có thể nối được với rất nhiều ngoại vi khác nhau như máy in, scanner, disk box.v.v. do đó với mỗi loại ngoại vi khác nhau, chương trình điều khiển sẽ thiết lập mode cho 8255 để phù hợp với ngoại vi đó. Chẳng hạn như khi kết nối với máy in, hai nhóm cổng của 8255 được thiết lập chế độ 1, với cổng A là cổng ra cho phép truyền số liệu cần in từ máy tính sang, 4 bits cao của cổng C sẽ là tín hiệu bắt tay. Cổng B là cổng vào cho phép truyền trạng thái của máy in sang máy tính và 4 bits thấp của cổng C là tìn hiệu bắt tay.

Trên máy tính IBM PC cho phép có được 4 công LPT là các cổng LPT1, LPT2, LPT3 và LPT4. Khi khởi động máy tính Hệ điều hành sẽ kiểm tra trạng thái của các cổng LPT và lưu địa chỉ các cổng này vào các ô nhớ có địa chỉ 00408 – 004f.

Vào ra nối tiếp

Vào ra nối tiếp là với một lệnh IN hay OUT thì chỉ 1 bits số liệu được truyền đi mà thôi. Truyền nối tiếp có tốc độ chậm hơn so với truyền song song nhưng khi truyền đi với khoảng cách xa thì số lượng dây dẫn ít hơn nhiều do đó dễ triển khai, giá thành hạ nên trên máy tính vẫn sử dụng đồng thời cả hai phương pháp truyền song song và truyền nối tiếp cho các ứng dụng khác nhau của truyền số liệu.

Trong truyền số liệu thường có hai giao thức là truyền đồng bộ và truyền không đồng bộ.

Ở truyền đồng bộ, máy thu và máy phát được làm việc với cùng 1 dao động chủ và mỗi lần truyền người ta thực hiện truyền 1 khối số liệu có dung lượng đến hàng chục KB.

Ở truyền không đồng bộ, máy thu và máy phát được làm việc với 2 nguồn dao động chủ khác nhau và mỗi lần truyền số liệu người ta chỉ truyền 1 byte.

Thủ tục của mỗi lần truyền bao gồm: Tín hiệu khởi động (start), khối số liệu (data), tín hiệu kiểm tra chẵn, lẻ (parity), tín hiệu kết thúc (stop). Thủ tục này bao giờ cũng được thiết lập giống nhau trên cả máy thu và máy phát gọi là định khung truyền số liệu.

Theo sự phát triển của công nghệ truyền số liệu, người ta phân ra các hình thức truyền: đơn công, bán song công và song công. ***(Tham khảo thêm trong tài liệu)***

Các thiết bị truyền thông trong truyền số liệu cũng được phân biệt thành:

Thiết bị đầu cuối (Data Tecminal Equipmet - DTE) là nguồn hoặc đích của số liệu, ví dụ như máy tính, máy điện thoại, máy fax.v.v.

Thiết bị truyền số liệu (Data Comunication Equipment – DCE) để chỉ các thiết bị truyền thông, ví dụ như modem - chịu trách nhiệm truyền dữ liệu.

Truyền số liệu nối tiếp trong máy tính theo tiêu chuẩn RS232

Chuẩn RS 232 cho truyền thông nối tiếp được đưa ra tại Mỹ từ năm 1961, khi cho ra đời PC, IBM đã sử dụng chuẩn này cho truyền thông nối tiếp của IBM PC, RS 232 gồm các tín hiệu sau:

TxD *(Transmitted Data)*-dữ liệu phát.

RxD *(Received Data)*- dữ liệu thu.

DTR *(data terminal ready)* - thiết bị đầu cuối sẵn sàng.

DSR *(data set ready)* - dữ liệu sẵn sàng.

RTS *(request to send)* - yêu cầu gửi dữ liệu.

CTS *(clear to send)* - tín hiệu thông.

DCD *(data carrier detect)* - dò sóng mang dữ liệu.

RI *(ring indicator)* - chỉ thị chuông.

Các tín hiệu trên đảm bảo đảm bảo cho RS 232 truyền số liệu nối tiếp giữa:

DTE với DTE – Nối trực tiếp 2 máy tính với nhau,

TDE với DCE – Nối máy tính với modem,

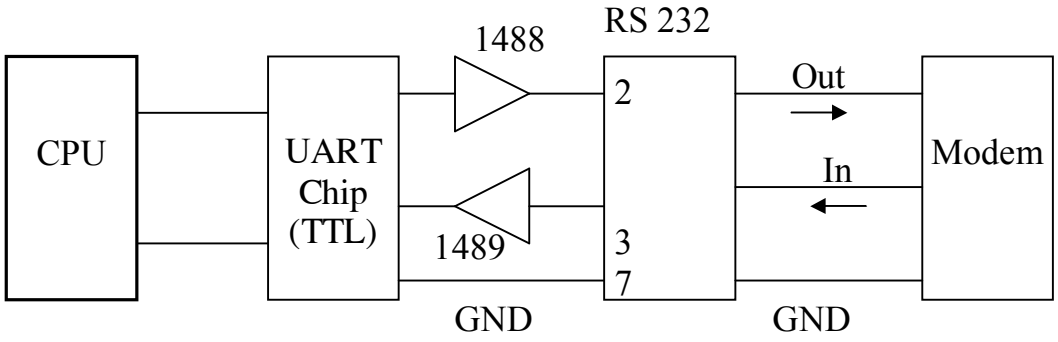
DCE với DCE – Nối hai modem với nhau.

Vì là chuẩn ra đời trước máy tính nên RS 232 quy định các mức tín hiệu "0" và "1" như sau:

Mức 1 tương ứng từ -3V đến -25V, còn mức 0 tương ứng từ +3V đến +25V, từ -3V đến +3V không xác định.

Trong khi đó, máy tính sử dụng điện áp tương thích với mức TTL quy định mức 1 tương ứng với + 5V, mức 0 tương ứng với GND.

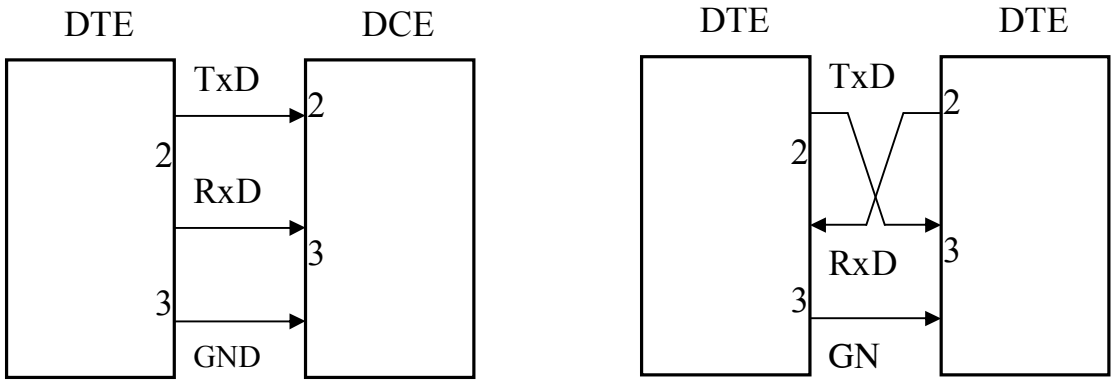
Chính vì vậy khi nối ghép máy tính với RS 232 cần phải có chuyển đổi mức điện áp từ TTL sang RS 232 và ngược lại. Người ta sử dụng các vi mạch chuyển đổi như sau:



Để chuyển đổi từ mức TTL sang RS 232 (ở mạch phát) người ta sử dụng vi mạch 1488. Để chuyển đổi từ mức RS 232 sáng TTL (ở mạch thu) người ta sử dụng vi mạch 1489.

Trên máy tính IBM PC, giao diện RS232 còn được gọi là cổng COM, thường máy tính có 4 cổng COM được đánh số thứ tự là COM1, COM2, COM3, COM4. Khi khởi động máy tính Hệ điều hành sẽ kiểm tra sự có mặt của các cổng COM và lưu địa chỉ các cổng này vào vùng nhớ có địa chỉ 00400 - 00407

Nối ghép giữa DTE với DTE và DTE với DCE đươc biểu diễn như hình vẽ dưới đây:



Vi mạch lập trình vào/ra nối tiếp 8250.

Vi mạch 8250 là vi mạch lập trình có chức năng truyền và nhận số liệu nối tiếp tại các cổng COM theo chuẩn RS 232 trên các máy tính IBM PC.

Vi mạch 8250 có các chân số liệu D0 – D7 được nối đến các chân D0 - D7 tương ứng của bus số liệu cho phép có thể lập trình cho vi mạch này.

Có 3 chân địa chỉ A2A1A0 được nối trực tiếp từ bus địa chỉ đến vi mạch cho phép lựa chọn 8 thanh ghi bên trong vi mạch là.

TT	A2	A1	A0	Tên thanh ghi
	0	0	0	Thanh ghi đệm số liệu thu và lưu số liệu phát
	0	0	1	Thangh ghi cho phép ngắt
	0	1	0	Thanh ghi nhận biết ngắt (chỉ đọc)
	0	1	1	Thanh ghi điều khiển đường truyền
	1	0	0	Thanh ghi điều khiển modem
	1	0	1	Thanh ghi trạng thái
	1	1	0	Thanh ghi trạng thái modem
	1	1	1	Thanh ghi xóa (reset)

Trên máy tính IBM PC cổng COM1 có địa chỉ giải mã từ 03f8h đến 03ffh.
Khi xét truyền nối tiếp giữa DTE và DTE trên các cổng COM ta sẽ xét các thanh ghi đệm số liệu thu và phát (03f8h), thanh ghi điều khiển đường truyền (03fbh) và thanh ghi trạng thái đường truyền (03fdh).
Xét việc truyền số liệu giữa DTE và DTE.
Khi đó cáp kết nối theo quy định như hình vẽ đã trình bày ở trên. Ta cần phải thiết lập cả ở máy thu và phát những tham số sau:
- Định dạng số liệu truyền,
- Tốc độ truyền.
Để thiết lập định dạng số liệu truyền ta lập trình vào thanh ghi điều khiển đường truyền (03fbh) có các bits được định nghĩa như sau:

<table>
<tr><th>D7</th><th>D6</th><th>D5</th><th>D4</th><th>D3</th><th>D2</th><th>D1</th><th>D0</th><th></th></tr>
<tr><td>DLAB</td><td>Break</td><td colspan="3" rowspan="6">Parity</td><td rowspan="4">Stop bit</td><td>0</td><td>0</td><td>5 bits data</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>6 bits data</td></tr>
<tr><td></td><td></td><td>1</td><td>0</td><td>7 bits data</td></tr>
<tr><td></td><td></td><td>1</td><td>1</td><td>8 bits data</td></tr>
<tr><td></td><td></td><td></td><td colspan="3">0 1 bit stop</td></tr>
<tr><td></td><td></td><td></td><td colspan="3">1 2 bit stop</td></tr>
<tr><td></td><td></td><td>0</td><td>0</td><td>0</td><td colspan="4">Không kiểm tra chẵn lẻ</td></tr>
<tr><td></td><td></td><td>0</td><td>0</td><td>1</td><td colspan="4">Kiểm tra lẻ</td></tr>
<tr><td></td><td></td><td>0</td><td>1</td><td>1</td><td colspan="4">Kiểm tra chẵn</td></tr>
</table>

Ví dụ: để thiết lập chế độ truyền giữa hai máy tính với nhau theo định dạng như sau: truyền mã ASCII (8 bits số liệu), 2 bits stop, kiểm tra chẵn, ta phải ghi vào thanh ghi điều khiển đường truyền giá trị là 9Fh bằng các lệnh sau:

```
MOV     DX,03fbh    ; Nạp địa chỉ thanh ghi điều khiển đường truyền
MOV     AL,09Fh     ; Định dạng truyền số liệu
OUT     DX, AL      ; Nạp thanh ghi điều khiển đường truyền
```

Hay để thiết lập chế độ truyền giữa 2 máy tính với nhau theo định dạng, 8 bits số liệu, không kiểm tra chẵn lẻ, 1 bit stop. Khi đó ta phải ghi vào thanh ghi điều khiển đường truyền giá trị là 83h.

```
MOV     DX,03fbh    ; Nạp địa chỉ thanh ghi điều khiển đường truyền
MOV     AL,83h      ; Định dạng truyền số liệu
OUT     DX, AL      ; Nạp thanh ghi điều khiển đường truyền
```

Để thiết lập tốc độ truyền số liệu ta cần phải đặt bit D7 của thanh ghi điều khiển đường truyền lên 1 và sẽ nạp vào 2 thanh ghi có địa chỉ 03f8h và 03f9h một số được tính như sau:

$$\text{Số chia} = \frac{X_{in}}{\text{Tốc độ}*16}$$

Ví dụ: Xác định số chia ghi vào các và MSB để có tốc độ truyền 300 bps với tần số đưa vào chân X_{in} = 1,8432 Mhz:

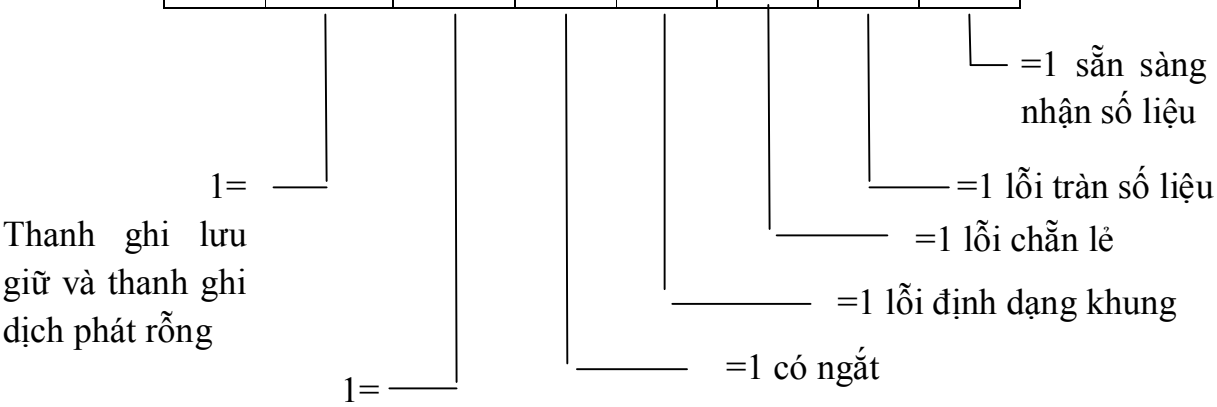
$$384 = \frac{1{,}8432*10^6}{300*16} == 180h$$

Lập trình :

```
        mov     al,80h          ; đặt DLAB = 1
        mov     dx, 3fbh        ; địa chỉ thanh ghi điều khiển đườngtruyền
        out     dx, al          ; đặt DLAB = 1
; gửi số chia
        mov     ax, 180         ; tốc độ truyền la 300 bps
        mov     dx, 3f8h        ; địa chỉ phần thấp bộ chia
        out     dx,al           ; nạp phần thấp số chia
        mov     al, ah          ; chuyển phần cao số chia sang al
        inc     dx              ; địa chỉ phần cao số chia
        out     dx, al          ; nạp phần cao số chia
```

Để truyền và nhận số liệu cần phải hỏi trạng thái của thanh ghi trạng thái đường truyền, nếu được phép lúc đó ta mới được nhận và truyền số liệu. Thanh ghi trạng thái đường truyền có địa chỉ 03fdh có các bits được định nghĩa như sau:

D7	D6	D5	D4	D3	D2	D1	D0
0	TEMT	THRE	BI	PE	PE	OE	DR



Như vậy ta chỉ có thể nhận số liệu khi mà bit Do của thanh ghi trạng thái đường truyền có mức logic 1, đoạn chương trình kiểm tra trạng thái đường truyền cho phép nhận số liệu như sau:

```
STATUS:
MOV     DX, 03fdh   ; Thanh ghi trạng thái đường truyền
IN      AL, DX      : đọc thanh ghi trạng thái đường truyền
AND     AL, 01
CMP     AL,01       ; kiểm tra bit D0 có mức logic "1"
JNE     STATUS      ; nếu chưa thì kiểm tra tiếp
RECEIVER:
MOV     DX, 03F8h   ; nhận nếu bit D0 có mức logic "1"
IN      AL, DX
```

Ta cũng chỉ có thể thực hiên truyền số liệu khi mà D6 D5 của thanh ghi trang thái đường truyền cùng có mức logic "1".

```
STATUS:
MOV     DX, 03fdh   ; Thanh ghi trạng thái đường truyền
IN      AL, DX      : đọc thanh ghi trạng thái đường truyền
AND     AL, 60h
CMP     AL,60       ; kiểm tra bit D6D5 có mức logic "1"
JNE     STATUS      ; nếu chưa thì kiểm tra tiếp
TRANSMISSION:
MOV     DX, 03F8h   ; truyền nếu bit D6D5 có mức logic "1"
OUT     DX, AL
```

Màn hình và Card điều khiển màn hình

Màn hình là một thiết bị ra cho phép hiển thị kết quả của các chương trình và giúp cho giao tiếp người – máy là trực tiếp.

Card điều khiển màn hình thực tế là một hệ vi xử lý được nối ghép với máy tính theo một số chuẩn: CGA, EGA, VGA và SVGA. Khi cần đưa thông tin lên màn hình BVXL của máy tính sẽ đưa các thông tin đó vào vùng nhớ video RAM (A0000h – BFFFFh) và BVXL của card điều khiển màn hình sẽ lấy thông tin đó, biến đổi và đưa ra màn hình.

Một số khái niệm cần phải nắm được:

- Điểm ảnh (Picxel),

- Độ phân giải (resolution),

- Khoảng cách giữa 2 điểm ảnh (dot pich)

Màn hình và card điều khiển màn hình có 2 chế độ làm việc.

Chế độ Text (văn bản).

Chế độ graphic (đồ họa).

Trong chế độ văn bản, mục đích là hiển thị các chữ cái Latinh lên màn hình, một ma trận điểm ảnh sẽ được dùng để hiển thị một chữ cái latinh được mã hóa và gọi là mã ASCII, ma trận điểm ảnh đó có thể là 9*14 hoặc 8*14 tùy theo độ phân giải của màn hình. Người ta tính số chữ cái la tinh (ký tự) có thể hiện ra trên màn hình gọi là độ phân giải. Ở chế độ văn bản có 2 độ phân giải thường là 80*25 (2000 ký tự trên màn hình) và 40*25 (1000 ký tự trên màn hình). Số ký tự trên màn hình có số thứ tự bắt đầu từ 1 được tính từ trái qua phải và từ trên xuống dưới. 1 ký tự trên màn hình cần có các thông tin sau:

Vị trí của nó trên màn hình (số thứ tự),

Ký tự mà nó cần hiển thị,

Màu sắc, độ sáng của ký tự được hiển thị.

Để xác định vị trí của ký tự người ta biểu diễn thông qua vị trí theo hàng và cột của nó trên màn hình. Ví dụ ký tự được biểu diễn tại hàng 12 cột 30 sẽ có số thứ tự là (12*80) + 30. Máy tính sẽ dùng thanh ghi dl để biểu diễn vị trí cột và thanh ghi dh để biểu diễn vị trí hàng của ký tự.

Nội dung của ký tự cần hiển thị chính là mã ASCII của nó, đây là 1 số 8 bits, do đó máy tính sẽ sử dụng 1 byte trong vùng nhớ video RAM để lưu giá trị này.

Màu sắc, độ sáng của ký tự cũng được biểu diễn bằng 1 byte được gọi là byte thuộc tính được định nghĩa như sau:

D7	D6	D5	D4	D3	D2	D1	D0
B	R	G	B	I	R	G	B
B	Nền			Màu chữ			

Byte thuộc tính trong chế độ TEXT

B = 0: Không nhấp nháy; I = 0: Độ sáng thường

Xem nội dung của byte thuộc tính trong tài liệu.

Như vậy, với 1 ký tự cần được hiển thị ra màn hình cần 2 byte thông tin về nó: Byte chứa mã ASCII của ký tự và byte thuộc tính. Với màn hình có độ phân giải 2000 ký tự sẽ cần 4000 byte để biểu diễn cho màn hình này. Do vậy ở chế độ văn bản người ta thường dùng bộ nhớ video RAM bắt đầu từ địa chỉ B8000h. Khi đó nội dung các ô nhớ của bộ nhớ video RAM sẽ có nội dung như sau:

B800:0000	B8000	Ký tự của hàng 1 cột 1
B800:0001	B8001	Thuộc tính của ký tự ở hàng 1 cột 1
B800:0002	B8002	Ký tự của hàng 1 cột 2
B800:0003	B8003	Thuộc tính của ký tự ở hàng 1 cột 2
B800: 07CE	B87CE	Ký tự hàng 25 cột 80
B800: 07CF	B87CF	Thuộc tính của ký tự ở hàng 25 cột 80

Căn cứ quy luật trên, người ta rút ra quy luật: byte chẵn chứa mã ASCII của ký tự, byte lẻ chứa thuộc tính của ký tự.

Màn hình CGA có 8 chế độ làm việc (từ 0 – 7)

Chế độ	**Điểm ảnh**	**Ký tự**	**Hộp ký tự**	**Chế độ**	**Số màu**	**Trang đệm**	**Đ/c đầu**
00H	320 * 350	40 * 25	8 * 14	Văn bản	16	8	B8000H
01H	320 * 350	40 * 25	8 * 14	Văn bản	16	8	B8000H
02H	640 * 350	80 * 25	8 * 14	Văn bản	16	8	B8000H
03H	640 * 350	40 * 25	8 * 14	Văn bản	16	8	B8000H
04H	320 * 200	40 * 25	8 * 8	Đồ họa	4	1	B8000H
05H	320 * 200	40 * 25	8 * 8	Đồ họa	4	1	B8000H
06H	640 * 200	80 * 25	8 * 8	Đồ họa	2	1	B8000H
07H	720 * 200	80 * 25	9 * 14	Văn bản	1	4	B0000H

Trong đó chế độ 4, 5,6 là chế độ đồ họa các chế đọ còn lại là chế độ văn bản.

Màn hình EGA có bổ sung thêm các chế độ đồ họa là 0D, 0E, 0F và 10h:

0DH	320 * 200	40 * 25	8 * 8	Đồ họa	16	2/4	A0000H
0EH	640 * 200	80 * 25	8 * 8	Đồ họa	16	1/2	A0000H
0FH	640 * 350	80 * 25	9 * 14	Đồ họa	1	1	A0000H
10H	640 * 350	80 * 25	8 * 14	Đồ họa	16	2	A0000H

Màn hình VGA có bổ sung thêm các chế độ đồ họa là 11h, 12h, và 13h:

Chế độ	Điểm ảnh	Ký tự	Hộp ký tự	Chế độ	Số màu	Trang đệm	Đ/c đầu
11H	640 * 480	80 * 30	8 * 16	Đồ họa	2	1	A0000H
12H	640 * 480	80 * 30	8 * 16	Đồ họa	16	1	A0000H
13H	320 * 200	40 * 25	8 * 8	Đồ họa	256	1	A0000H

Lập trình ở chế độ văn bản:

Để lập trình cho màn hình ở chế độ văn bản, dùng ngắt int 10h với các giá trị khác nhau của AH để có các chức năng khác nhau. Chi tiết xem trong tài liệu

Int 10h với ah =00h; thiết lập chế độ - Giá trị của AL là chế độ màn hình

Int 10h với ah=02h; dịch chuyển con trỏ đến vị trí được ghi trong dh và dl

Int 10h với ah =0eh; hiển thị ra màn hình tại ví trí của con trỏ ký tự có mã ASCII nằm trong AL.

Int 10h với ah=09h; hiển thị ra màn hình tại ví trí của con trỏ ký tự có mã ASCII nằm trong AL, thuộc tính nằm trong BL, CX số lần hiển thị.

Int 10h với ah=13h; hiển thị ra màn hình dãy ký tự .v.v.

Chế độ đồ họa của màn hình và card màn hình.

Ở chế độ đồ họa, Card điều khiển màn hình sẽ điều khiển đến từng điểm sáng trên màn hình. Thông tin về 1 điểm sáng trên màn hình bao gồm:

Vị trí của điểm sáng trên màn hình.

Màu sắc và độ sáng của điểm sáng đó trên màn hình.

Do độ phân giải lớn nên lúc này không thể dùng thanh ghi 8 bits để biểu diễn vị trí của ký tự, người ta dùng thanh ghi DX để biểu diễn vị trí hàng và thanh ghi CX để biểu diễn vị trí cột của ký tự.

Màu sắc của điểm ảnh phụ thuộc vào số bit mã hóa màu, nếu số bit mã hóa mãu càng lớn thì màu sắc càng rõ ràng và chất lượng của hình ảnh càng cao. Nếu dùng *n* bits mã hóa mà ta có 2^n màu.

Như vậy ở chế độ đồ họa, với độ phân giải lớn số bit mã hóa màu lớn nên bộ nhớ video RAM cùng cần rất lớn, không gian 128 KB video RAM có địa chỉ từ

A0000 h – BFFFFh không đủ để chứa thông tin về số điểm ảnh và màu sắc của nó, vì vậy mà bộ nhớ video RAM trên Card màn hình được bổ sung rất lớn từ 256 KB, 512KB, 1MB, 512 MB thậm chí đến hàng GB.

Lập trình màn hình ở chế độ đồ họa:

Người ta dùng ngắt int 10h với ah=0ch,

Vị trí của điểm ảnh được biểu diễn bằng nội dung của DX (hàng) và CX (cột).

Màu sắc của điểm ảnh được biểu diễn bằng giá trị của AL

Xem thêm ví dụ vẽ đường thẳng, đường ngang, đường chéo trên màn hình.

Card tăng tốc đồ họa AGP.

Ở chế độ đồ họa, để biểu diễn một trang màn hình có độ phân giải cao, màu sắc rõ nét sẽ cần một dung lượng nhớ rất lớn. Khi thực hiện sử lý hình ảnh chuyển động thì trong 1 giây cần phải đưa đên 30 trang lên màn hình, lượng thông tin trao đổi giữa máy tính và Card điều khiển màn hình sẽ rất lớn (dải thông đòi hỏi cao). Thực tế bus PCI của máy tính không đủ dải thông để truyền lượng thông tin này , do vậy người ta thiết kế 1 giao diện truyền số liệu giữa Card điều khiển màn hình và CPU của máy tính trên cơ sở tốc độ truyền số liệu giữa CPU và RAM Card màn hình trên giao diện đó gọi là card tăng tốc đồ họa AGP.

Xem thêm trong tài liệu.

Trên cơ sở dải thông của bus PCI là 266MBps, người ta thiết kế Card AGP với các tốc độ truyền:

2X = 2 * 266MBps

4X = 4* 266MBps

8X = 8* 266MBps

Ngoài việc tăng dải thông, để tăng tốc độ truyền AGP còn cải tiến cả giao thức truyền số liệu.

Hiện nay khi mà PCI dược nâng cấp PCI Exp có dải thông đến 16X thì người ta không sử dụng AGP nữa mà sử dụng chuẩn PCI Exp cho các máy tính.

Giao diện tuần tự đa năng – USB.

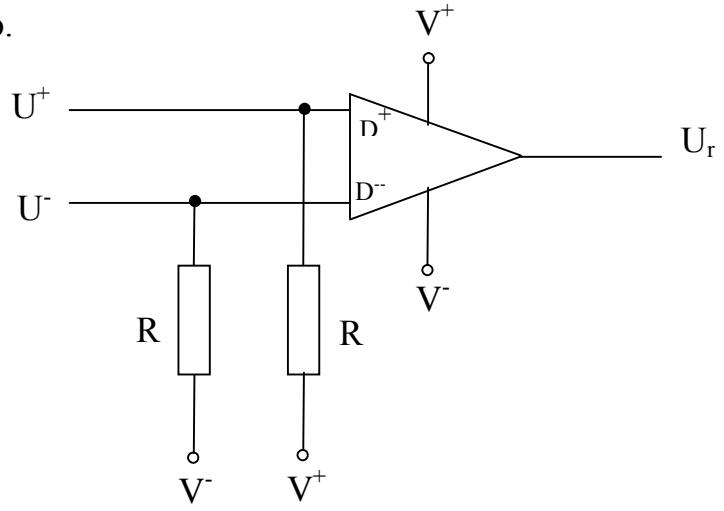
Trên máy tính cho phép có tối đa 4 cổng COM và 4 cổng LPT như vậy chỉ có thể nối tối đa 8 ngoại vi đến máy tính.

Mặt khác tốc độ truyền của các cổng COM và LPT rất hạn chế, không đảm bảo được khi máy tính kết nối với các thiết bị xử lý ảnh như camera, scanner tốcđộ cao.v.v.

Xuất phát từ thực tế trên, để theo kịp được công nghệ của Machintosh (giao diện 1394) Intel đã đề xuất 1 giao diện tuần tự đa năng – USB cho phép nối được nhiều ngoại vi và có tốc độ truyền lớn.

Giao diện USB sử dụng giao thức IP (Internet Protocol) nên có thể quản lý đến 127 ngoại vi nhờ sử dụng hub kết nối.

Về mặt truyền số liệu, USB sử dụng phương pháp truyền vi sai (dây truyền tín hiệu có 2 dây D^+ và D^- kết hợp với bộ khuyếch đại vi sai đảm bảo truyền số liệu với tốc độ cao.



Tín hiệu sẽ được truyền trên hai dây tín hiêu là U+ và U- được đưa vào hai chân D+ và D- tương ứng của bộ khuyếch đại vi sai. Với sơ đồ trên tín hiệu trên đầu ra Ura sẽ là:

$$\mathbf{U}_{ra} = \begin{cases} V^+ & if \quad |D^+| \geq |D^-| \\ V^- & if \quad |D^+| < |D^-| \end{cases}$$

Việc sử dụng bộ khuyếch đại vi sai có ưu điểm với hai tín hiệu U+ và U- được xoắn với nhau cho phép triệt tiêu được nhiễu tác động lên đường truyền. Nếu có một nhiễu loạn U_{nh} tác động lên đường truyền thì tín hiệu đó sẽ tác động đồng

thời lên cả hai dây tín hiệu U^+ và U^-, khi đó:

Tín hiệu tác động lên đầu vào D^+ và D^-: $-\begin{cases} D^+ = U^+ \pm U_{nh} \\ D^- = U^+ \mp U_{nh} \end{cases}$

Khi truyền qua khuyếch đại vi sai: $D^+ - D^- = \left(U^+ \pm U_{nh}\right) - \left(U^- \mp U_{nh}\right) = U^+ - U^-$

Rõ ràng tín hiệu nhiễu tác động lên đường truyền đã bị triệt tiêu khi truyền qua bộ khuyếch đại vi sai.

USB cho phép máy tính nối được rất nhiều thiết bị, vì vậy với mỗi loại thiết bị khác nhau cần có những phương thức truyền số liệu phù hợp, vì vậy có rất nhiều phương thức truyền số liệu ở giao thức USB

Truyền điều khiển

Truyền ngắt

Truyền đồng bộ cách biệt

Truyền khối

Xem thêm trong tài liệu

Khi sử dụng cụ thể ngoại vi nào, chương trình điều khiển sẽ sử dụng giao thức phù hợp với ngoại vi đó.

Bàn phím máy tính.

Bàn phím máy tính là một thiết bị ngoại vi cho phép giao tiếp người máy là giao tiếp trực tiếp.

Bộ điều khiển bàn phím là một hệ vi xử lý sẽ thực hiện:

Nhận biết phím ấn (quét bàn phím).

Gán cho phím được ấn một mã gọi là mã bàn phím.

Truyền mã bàn phím sang máy tính để xử lý.

Xem thêm chi tiết phần vừa nêu ra trong tài liệu.

Máy tính sẽ đọc mã quét bàn phím từ cổng A của vi mạch 8255 trên MotherBoard với địa chỉ là 60h. Ta thấy một phím được ấn, chỉ cung cấp 1 mã bàn phím, thực tế 1 phím có đến 2 mã ASCII (Ví dụ "A" có mã ASCII là 41h, "a" có mã ASCII là 61h). Để làm được điều này chương trình phục vụ ngắt bàn phím sẽ phải có thêm một thông tin nữa là byte trạng thái bàn phím.

Địa chỉ 0040:0017 lưu trữ byte trạng thái bàn phím thứ nhất và địa chỉ 0040:0018 lưu trữ byte trạng thái bàn phím thứ hai, *Xem ý nghĩa của từng bit trong tài liệu.*

Khi một phím được ấn chương trình con phục vụ ngắt INT 09 nhận mã quét và lưu vào vùng nhớ gọi là bộ đệm bàn phím thứ nhất.

Giao tiếp bàn phím sử dụng 2 ngắt để thực hiện,

Ngắt cứng int 09h để nhận mã quét, gán mã ASCII cho mã quét và cất cả mã phím và mã ASCII vào bộ đệm bàn phím thứ nhất.

Ngắt mềm int 16h thực hiện đọc mã phím và mã ASCII từ bộ đệm bàn phím thứ nhất vào bộ đệm bàn phím thứ 2.

Khi int 09h và int 16h không kịp xử lý thông tin trong các bộ đệm sẽ dẫn đến hiện tượng tràn bộ đệm bàn phím thứ nhất và bộ đệm bàn phím thứ 2.

Xem chi tiết cách nhận biết tràn bộ đệm bàn phím trong tài liệu.

Lập trình bàn phím

Để lập trình bàn phím, người ta thường dùng ngắt mềm int 16h, với bàn phím mở rộng.

Int 16h với ah =10h,

Int 16h với ah =11h,

Int 16h với ah = 12h.

Xem chi tiết trong tài liệu.

Ổ đĩa.

Ổ đĩa là một thiết bị nhớ ngoài dùng để lưu trữ thông tin về Hệ điều hành, các chương trình ứng dụng và các kết quả của các chương trình ứng dụng.

Có thế phân chia đĩa thành:

Đĩa mềm, - FDD

Đĩa cứng – HDD

Đĩa quang – CD ROM, DVD ROM.

Đĩa là tấm nhựa hình tròn có đường kính 3,5" hay 2,5" với FDD và HDD thì trên đó được phủ 1 lớp ô xít sắt, quá trình ghi và đọc thông tin lên đĩa là từ tính và đọc từ tính trên bề mặt đĩa.

Cả hai bề mặt đĩa đều được sử dụng, mặt trên gọi là mặt 0, mặt dưới gọi là mặt 1.

Trên mối mặt đĩa được chia thành nhiều vòng tròn đồng tâm được đánh số từ ngoài vào trong gọi là các rãnh hay track, số lượng track trên một mặt đĩa phụ thuộc vào kiểu đĩa và dung lượng của đĩa.

Trên mỗi track lại được chia thành nhiều cung nhỏ, mỗi cung này là 1 đơn vị ghi thông tin nhỏ nhất của đĩa gọi là sector, mỗi sector có khả năng ghi 1 lượng thông tin nhất định phụ thuộc vào loại đĩa. Số lượng sector có trên 1 track cũng phụ thuộc vào loại đĩa và dung lượng đĩa.

Khi thực hiện ghi thông tin lên đĩa, Hệ điều hành ghi thông tin và cất giữ dưới dạng tập tin (file) lúc đó nó sử dụng 1 đơn vị cấp phát dung lượng đĩa để ghi cho file gọi là 1 liên cung (cluster). 1 cluster có thể là 1 hay 1 vài sector, số lượng sector trong 1 cluster phụ thuộc vào dung lượng của đĩa và Hệ điều hành.

Để ghi được thông tin lên đĩa cần phải có 1 thủ tục gọi là định dạng (format) đĩa, quá trình format đĩa chia không gian lưu trữ thông tin trên đĩa thành 4 vùng:

Bản ghi khởi động - Boot record,

Bảng tìm kiếm file – FAT,

Danh sách các file có trên đĩa – Directory,

Số liệu được cất giữ trên đĩa – Data.

Cơ chế tìm kiếm file được cất giữ trên đĩa thông qua bảng FAT

Xem chi tiết trong tài liệu

Bài tập lập trình

Phần bài tập sẽ sử dụng chủ yếu những kiến thức sau:

Ngắt int 21h

Ngắt int 10h

Ngắt int 16h.

Lập trình cho cổng truyền thông nối tiếp qua cổng COM

Các ví dụ mà giáo viên đã đưa ra trên lớp học và các bài tập được giao về nhà.