

ĐẠI HỌC KINH DOANH VÀ CÔNG NGHỆ HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN



GIÁO TRÌNH

KIẾN TRÚC MÁY TÍNH

Chủ biên : TS. Hoàng Xuân Thảo
Biên soạn: GS. Trần Anh Bảo
ThS. Nguyễn Thị Minh Ngọc

(Dùng cho chương trình đào tạo hệ đại học)
Lưu hành nội bộ

HÀ NỘI - 2014

MỤC LỤC

CHƯƠNG 1. TỔNG QUAN	4
1.1. Khái niệm chung về kiến trúc máy tính	4
1.2. Máy tính và phân loại máy tính	4
1.3. Lịch sử phát triển của máy tính.	8
1.4. Hệ thống máy tính	13
1.4.1. Các thành phần chính của máy tính	13
1.4.2. Hoạt động của máy tính	17
CHƯƠNG 2. CÁC HỆ ĐẾM VÀ BIỂU DIỄN THÔNG TIN TRONG MÁY TÍNH	
.	18
2.1. Hệ nhị phân (Binary)	18
2.1.1. Khái niệm:	18
2.1.2. Biến đổi từ nhị phân sang thập phân	18
2.1.3. Biến đổi thập phân thành nhị phân	19
2.2. Hệ thập lục phân (Hexadecima).	19
2.2.1 Khái niệm:	19
2.2.2. Biến đổi thập lục phân thành thập phân	20
2.2.3. Biến đổi thập phân thành thập lục phân	20
2.2.4. Biến đổi thập lục phân thành nhị phân	20
2.2.5. Biến đổi nhị phân thành thập lục phân	21
2.3. Biểu diễn ký tự trong máy tính	21
2.3.1. Bảng mã ASCII.(American Standard Code for Information Interchange)	
.	21
2.4. Biểu diễn giá trị số trong máy tính.	27
2.4.1 Biểu diễn số nguyên.	27
2.4.2. Biểu diễn số thực	28
CHƯƠNG 3. BỘ XỬ LÝ TRUNG TÂM (CPU)	30
3.1. Khái niệm	30
3.2. Giới thiệu cấu trúc bên trong của bộ vi xử lý 8088.	30
3.2.1. Đơn vị giao diện bus (BIU).	30
3.2.2. Đơn vị thực hiện lệnh (EU-Execution Unit).	31
3.2.3. Các thanh ghi.	31
3.3. Chức năng và thông số của BUS	34
3.3.1. BUS trong máy vi tính.	35
3.3.2. Trọng tài bus (bus arbitration).	41
3.4. Xử lý ngắt	43
3.5. Một số bus thông dụng	44
3.5.1. Bus IBM PC	44
3.5.2. Bus IBM PC/AT	47
3.5.3. Bus PCI.	47

3.5.4. Bus nối tiếp chung USB.....	48
CHƯƠNG 4. HỆ THỐNG NHỚ MÁY VI TÍNH	49
4.1. Tổng quan	49
4.1.1. Đặc trưng của hệ thống nhớ	50
4.1.2. Quản lý bộ nhớ (MMU, Memory Management Unit)	50
4.1.3. Tổ chức bộ nhớ của vi xử lý.....	51
4.2. Bộ nhớ trong.	51
4.2.1. Khái niệm và phân loại	51
4.2.2. ROM-BIOS.	53
4.2.3. RAM.....	53
4.2.4 Tổ chức bộ nhớ RAM của máy tính.....	57
4.3. Hoạt động của Cache	58
Làm tươi bộ nhớ DRAM.....	60
Chuyển một mảng số liệu bằng DMA.....	62
4.4. Bộ nhớ ngoài	64
CHƯƠNG 5. HỆ THỐNG VÀO RA VÀ THIẾT BỊ NGOẠI VI	66
5.1. Tổng quan	66
5.1.1.Thiết bị ngoại vi	66
5.1.2. Module vào ra.....	66
5.2.Vai trò của bộ phối ghép	66
5.3. Cấu trúc chung của khối ghép nối	68
5.3.1. Nhiệm vụ của các khối trong khối ghép nối (KGN).	68
5.3.2. Sơ đồ khối.	68
5.4. Giải mã địa chỉ cho bộ ghép nối.	70
TÀI LIỆU THAM KHẢO	72

CHƯƠNG 1. TỔNG QUAN

1.1. Khái niệm chung về kiến trúc máy tính

Kiến trúc máy tính (Computer architecture) là một khái niệm trừu tượng của một hệ thống tính toán dưới quan điểm của người lập trình hoặc người viết chương trình dịch.

Nói cách khác, kiến trúc máy tính được xem xét theo khía cạnh mà người lập trình có thể can thiệp vào mọi mức đặc quyền, bao gồm các thanh ghi, ô nhớ các ngắt ... có thể được thâm nhập thông qua các lệnh.

1.2. Máy tính và phân loại máy tính

a) Máy tính

Máy tính hay máy điện toán là những thiết bị hay hệ thống thực hiện tự động các phép toán số học dưới dạng số hoặc phép toán logic. Các máy tính cỡ nhỏ thường gọi là máy vi tính, trong số đó máy dùng cho cá nhân thường gọi là máy tính cá nhân.

Máy tính được cấu tạo bởi các thành phần có thể thực hiện các chức năng đơn giản đã định nghĩa trước. Quá trình tác động tương hỗ phức tạp của các thành phần này tạo cho máy tính một khả năng xử lý thông tin. Nếu được thiết lập chính xác (thông thường bởi các chương trình máy tính) máy tính có thể mô phỏng lại một số khía cạnh của một vấn đề hay của một hệ thống. Trong trường hợp này, khi được cung cấp một bộ dữ liệu thích hợp nó có thể tự động giải quyết vấn đề hay dự đoán trước sự thay đổi của hệ thống.

Khoa học nghiên cứu về lý thuyết, thiết kế và ứng dụng của máy tính được gọi là khoa học máy tính, hay khoa học điện toán.

Từ "máy tính" (*computer*), đầu tiên, được dùng cho những người tính toán số học, có hoặc không có sự trợ giúp của máy móc, nhưng hiện nay nó hoàn toàn có nghĩa là một loại máy móc. Đầu tiên máy tính chỉ giải các bài toán số học, nhưng máy tính hiện đại làm được nhiều hơn thế. Máy tính có thể mua ở Anh đầu tiên là máy Ferranti Mark 1 Star sản xuất năm 1951 ^[1]

Đến những năm 1990, khái niệm máy tính đã thực sự tách rời khỏi khái niệm điện toán và trở thành một ngành khoa học riêng biệt với nhiều lĩnh vực đa dạng và khái niệm hơn hẳn ngành điện toán thông thường và được gọi là công nghệ thông tin. Tuy vậy đến ngày nay, một số người vẫn còn nhầm lẫn giữa hai khái niệm điện toán và công nghệ thông tin.

b) Phân loại máy tính

Theo mục đích sử dụng chúng ta có:

Máy chủ thực hiện nhiều chức năng hoặc một chức năng duy nhất không bao giờ nghỉ:

- Siêu máy tính

- Siêu máy tính cỡ nhỏ
- Mainframe
- Máy chủ doanh nghiệp
- Máy tính mini
- Máy trạm (*workstation*)

Máy tính phục vụ dân dụng:

- Máy tính cá nhân (PC - Personal Computer)
- Máy tính để bàn (*Desktop*)
- Máy tính xách tay (*Laptop gồm nhiều dòng laptop phổ thông, notebook, untrabook, laptop quân đội và workstation mobile*)
- Máy tính bảng (như Ipad)
- Thiết bị hỗ trợ kỹ thuật số cá nhân (PDA)
- Máy tính tháo lắp

Điểm yếu của xu hướng phân loại này là tính chất mơ hồ của nó. Cách phân loại này thường được sử dụng khi cần phân loại tại một thời điểm nào đó trong quá trình phát triển của ngành công nghiệp máy tính. Sự phát triển nhanh chóng của công nghiệp máy tính đã làm cho định nghĩa trên nhanh chóng trở nên lạc hậu. Rất nhiều loại máy tính hiện nay không được còn sử dụng nữa, như máy phân tích vi phân (*differential analyzer*), không được đưa vào danh sách này. Những sơ đồ phân loại khác cần được đề ra để định nghĩa thuật ngữ máy tính một cách ít (hoặc không) mơ hồ hơn.

Theo mức cải tiến công nghệ

Một cách phân loại máy tính ít mơ hồ hơn là theo mức độ hoàn thiện của công nghệ. Những chiếc máy tính có mặt sớm nhất thuần túy là máy cơ khí. Trong thập niên 1930, các thành phần rơ le cơ-điện đã được giới thiệu vào máy tính từ ngành công nghiệp liên lạc viễn thông. Trong thập niên 1940, những chiếc máy tính thuần túy điện tử đã được chế tạo từ những đèn điện tử chân không. Trong hai thập niên 1950 và thập niên 1960, bóng điện tử dần dà được thay thế bởi transistor, và từ cuối thập niên 1960 đầu thập niên 1970 là bởi mạch tích hợp bán dẫn (chíp bán dẫn, hay IC) cho đến hiện nay.

Một hướng nghiên cứu phát triển gần đây là máy tính quang (*optical computer*) trong đó máy tính hoạt động theo nguyên lý của ánh sáng hơn là theo nguyên lý của các dòng điện; đồng thời, khả năng sử dụng DNA trong công nghệ máy tính cũng đang được thử nghiệm. Một nhánh khác của việc nghiên cứu có thể dẫn công nghiệp máy tính tới những khả năng mới như tính toán lượng tử, tuy rằng nó vẫn còn ở giai đoạn đầu của việc nghiên cứu.

Theo đặc trưng thiết kế

Các máy tính hiện đại đã liên kết các đặc trưng thiết kế chính được phát triển bởi nhiều người đóng góp trong nhiều năm. Các đặc trưng này phần lớn không phụ thuộc vào mức độ hoàn thiện của công nghệ. Các máy tính hiện đại nhận được khả năng tổng thể của chúng theo cách mà các đặc trưng này tác động qua lại với nhau. Một số đặc trưng quan trọng được liệt kê dưới đây:

- Kỹ thuật số và kỹ thuật tương tự

Một quyết định nền tảng trong việc thiết kế máy tính là hoặc sử dụng kỹ thuật số (*digital*) hoặc sử dụng kỹ thuật tương tự (*analog*). Các máy tính kỹ thuật số (*digital computer*) tính toán trên các giá trị số rời rạc (*discrete value*) hoặc giá trị tượng trưng (*symbolic value*), trong khi đó máy tính tương tự (*analog computer*) tính toán trên các tín hiệu dữ liệu liên tục (*continuous data signal*). Bắt đầu từ thập niên 1940, máy tính kỹ thuật số đã trở nên phổ biến hơn mặc dù máy tính tương tự vẫn được sử dụng cho một số mục đích đặc biệt như trong kỹ thuật robot và việc kiểm soát các lò xyclôtron. Các thiết kế khác dùng tính toán xung lượng và tính toán lượng tử cũng hiện hữu nhưng chúng hoặc được sử dụng cho các mục đích đặc biệt hoặc vẫn đang trong vòng thử nghiệm.

- Nhị phân và Thập phân

Một phát triển quan trọng trong thiết kế tính toán kỹ thuật số là việc sử dụng hệ nhị phân như là hệ thống số đếm nội tại. Điều này đã bãi bỏ những yêu cầu cần thiết trong các cơ cấu kỹ thuật phức tạp của các máy tính sử dụng hệ số đếm khác, chẳng hạn như hệ thập phân. Việc áp dụng hệ nhị phân đã làm cho việc thiết kế trở nên đơn giản hơn để thực hiện các phép tính số học và các phép tính logic.

- Khả năng lập trình

Khả năng lập trình của máy tính (*programmability*), nghĩa là cung cấp cho nó một tập hợp các chỉ thị để thực hiện mà không có sự điều khiển vật lý đối với nó, là một đặc trưng thiết kế nền tảng của phần lớn các máy tính. Đặc trưng này là một sự mở rộng đáng kể khi các máy tính đã được phát triển đến mức nó có thể kiểm soát dòng thực hiện của chương trình. Điều này cho phép máy tính kiểm soát được thứ tự trong sự thực thi các chỉ lệnh trong chương trình dựa trên các dữ liệu đã được tính ra.

Điểm nổi bật chính trong thiết kế này đó là nó đã được đơn giản hóa một cách đáng kể với việc áp dụng các phép tính số học theo hệ đếm nhị phân để có thể mô tả hàng loạt các phép tính logic.

- Lưu trữ

Trong quá trình tính toán, máy tính thông thường cần phải lưu trữ các giá trị trung gian để có thể sử dụng trong các tính toán sau đó. Khả năng thực hiện của máy tính

phần lớn phụ thuộc vào tốc độ đọc các giá trị từ bộ nhớ và tốc độ ghi vào bộ nhớ, cũng như dung lượng bộ nhớ. Ban đầu bộ nhớ chỉ được sử dụng cho các giá trị trung gian, nhưng từ thập niên 1940 thì chính bản thân chương trình cũng có thể được lưu trữ theo cách này. Điểm nổi trội này đã dẫn đến việc ra đời của những chiếc máy tính có sẵn chương trình đầu tiên của thế hệ máy tính ngày nay.

Theo năng lực sử dụng

Có lẽ cách tốt nhất để phân loại các thiết bị máy tính là theo năng lực nội tại của nó, hơn là theo việc sử dụng, sự hoàn thiện công nghệ hay các đặc trưng thiết kế. Máy tính có thể chia làm ba dạng chính dựa theo năng lực sử dụng:

- **Các thiết bị có một mục đích** chỉ có thể thực hiện duy nhất một chức năng (ví dụ cỗ máy Antikythera năm 87 trước công lịch, và máy dự báo thủy triều của Lord Kelvin năm 1876)
- **Các thiết bị có mục đích đặc biệt** có thể thực hiện một số chức năng hữu hạn (ví dụ động cơ vi phân số 1. - *Difference Engine No 1* - của Charles Babbage năm 1832 và máy phân tích vi phân của Vannevar Bush năm 1932)
- **Các thiết bị có mục đích không nhất định** là các dạng máy tính sử dụng ngày nay.

- Các máy tính có sẵn chương trình

Trong cuối thập niên 1940 thiết kế đầu tiên cho máy tính có sẵn chương trình (*stored-program computer*) đã được phát triển và biên khảo (Xem thêm Bản thảo đầu tiên) tại trường công nghệ điện Moore của Đại học Pennsylvania. Phương pháp giải quyết, miêu tả trong tài liệu, được biết đến như là kiến trúc Von Neumann, mang tên của nhà toán học Jon von Neumann mặc dù các thành viên của trường công nghệ điện Moore mới thực sự sáng chế ra thiết kế này. Kiến trúc Von Neumann đã giải quyết vấn đề thuộc về thiết kế của máy ENIAC và sửa đổi bằng cách lưu trữ chương trình của máy trong bộ nhớ của nó. Von Neumann cung cấp thiết kế này cho các nhà nghiên cứu khác ngay sau khi ENIAC được công bố vào năm 1946. Nhiều kế hoạch đã được phát triển để hoàn thiện thiết kế này tại trường Moore trong chiếc máy có tên gọi là EDVAC. EDVAC đã không hoạt động được cho đến tận năm 1953 vì những khó khăn kỹ thuật trong việc hoàn thiện độ tin cậy của bộ nhớ. Từ bản sao của thiết kế này, các viện nghiên cứu khác đã giải quyết được vấn đề đó trước trường Moore và hoàn thiện các máy tính có sẵn chương trình của họ. Theo thứ tự của việc hoạt động thành công thì 5 chiếc máy tính có sẵn chương trình đầu tiên dựa trên cơ sở của kiến trúc Von Neumann là:

Thiết kế "chương trình có sẵn", được định nghĩa bởi kiến trúc Von Neumann, cuối cùng đã cho phép máy tính khai thác tiềm năng "mục đích không nhất định" của chúng. Bằng cách lưu trữ chương trình trong bộ nhớ, chúng có thể nhanh chóng "nhảy" từ chỉ thị này tới chỉ thị khác dựa trên kết quả của một điều kiện như đã

được định nghĩa sẵn trong chương trình. Các điều kiện này thông thường lượng giá các dữ liệu đã được tính toán bởi chương trình và cho phép chương trình trở thành động hơn. Thiết kế này cũng hỗ trợ vào khả năng tự động viết lại chương trình ngay trong khi nó đang thực thi - một đặc trưng rất mạnh nhưng cần sử dụng một cách cẩn thận. Các đặc trưng này là nền tảng cho các máy tính hiện đại.

Nói một cách chính xác, phần lớn các máy tính hiện đại là thiết bị tính toán theo phép nhị phân, bằng điện tử, có sẵn chương trình và có mục đích không nhất định.

- Các máy tính có mục đích đặc biệt

Các máy tính có mục đích đặc biệt (*special-purpose computer*) đã được phổ biến trong thập niên 1930 và đầu thập niên 1940 nhưng vẫn chưa bị thay thế hoàn toàn bởi các máy tính có mục đích không nhất định. Sự giảm xuống về kích thước và giá cả cũng như sự tăng năng lực của chúng đã khiến việc sử dụng máy tính có mục đích đặc biệt trong các ứng dụng đặc biệt trở thành một hiệu quả tốt về mặt chi phí. Rất nhiều các thiết bị dùng tại nhà và trong công nghiệp như điện thoại di động, máy thu video, hệ thống đánh lửa tự động v.v có chứa loại máy tính có mục đích đặc biệt này. Trong một số trường hợp các máy tính này là loại Turing hoàn tất (như máy chơi trò chơi điện tử, PDA) nhưng rất nhiều trong số chúng được lập trình một lần tại nhà máy sản xuất và rất ít khi phải lập trình lại. Chương trình mà các thiết bị này thực thi thông thường được lưu trữ trong bộ nhớ chỉ đọc (ROM) mà khi cần thiết có thể thay thế để thay đổi hoạt động của máy. Các máy tính được nhúng bên trong các thiết bị khác thông thường được gọi là vi điều khiển (*microcontroller*) hay máy tính nhúng (*embedded computer*).

- Các máy tính có một mục đích

Các máy tính có một mục đích (*single-purpose computer*) là loại xuất hiện sớm nhất của thiết bị máy tính. Khi được cung cấp dữ liệu, nó có thể tính kết quả của một hàm đơn giản đã được thiết lập trong cơ chế của nó. Các máy tính có mục đích không nhất định gần như đã thay thế hoàn toàn các máy tính có một mục đích và, do đó, đã phát sinh một lĩnh vực hoạt động mới của loài người: phát triển phần mềm. Các máy tính có mục đích không nhất định cần phải được lập trình với một bộ chỉ thị liên quan đến phần mềm máy tính. Việc thiết kế các thiết bị tính toán có một mục đích hay có mục đích đặc biệt hiện nay là những bài tập khái niệm thuần túy bao gồm các phần mềm thiết kế.

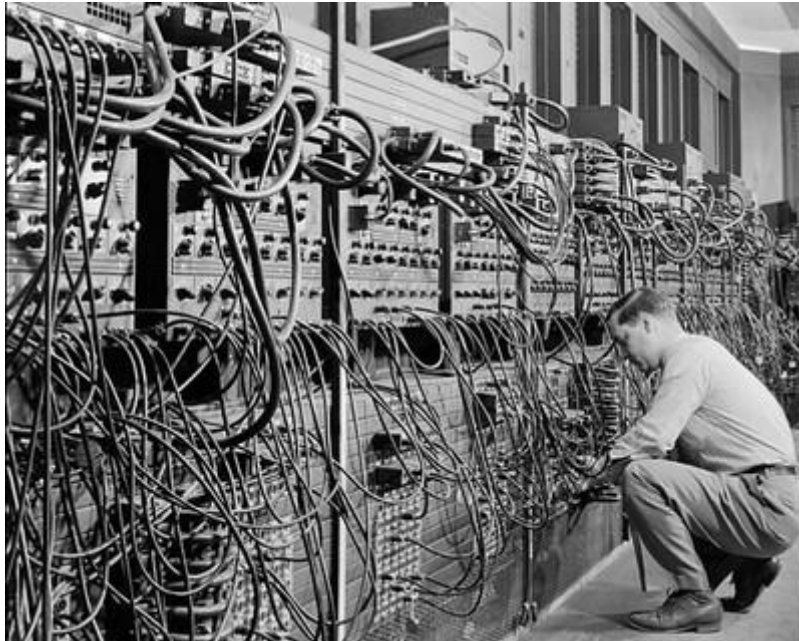
Theo hình thức hoạt động

Máy tính có thể được phân loại tùy theo cách thức người dùng vận hành. Có hai loại chính: kiểu xử lý tuần tự (*batch processing*) và kiểu xử lý tương tác (*interactive processing*).

1.3. Lịch sử phát triển của máy tính.

Lịch sử phát triển của máy tính điện tử có thể chia làm bốn thế hệ như sau:

- **Thế hệ 1:** (1945-1955). Máy tính được xây dựng trên cơ sở đèn điện tử mà mỗi đèn tượng trưng cho 1 bit nhị phân. Do đó máy có khối lượng rất lớn, tốc độ chậm và tiêu thụ điện năng lớn. Chiếc máy tính điện tử đầu tiên là ENIAC (Electronic Numerical Integrator and Computer) được ra đời năm 1946, được chế tạo từ những đèn điện tử, role điện tử và các chuyển mạch cơ khí, có khối lượng 30 tấn, tiêu thụ công suất 140KW.



Hình 1-1: ENIAC

- **Thế hệ thứ 2:** (1955-1965). Máy tính được xây dựng trên cơ sở là các đèn bán dẫn (transistor), máy tính đầu tiên thế hệ này có tên là TX-0 (transistorized experimental computer 0).



Hình 1- 2 : TX-0 (transistorized experimental computer 0)

- **Thế hệ thứ ba:** (1965-1980). Máy tính được xây dựng trên các vi mạch cỡ nhỏ (SSI) và cỡ vừa (MSI), điển hình là thế hệ máy System/360 của IBM. Thế hệ máy tính này có những bước đột phá mới như sau:

- Tính tương thích cao: Các máy tính trong cùng một họ có khả năng chạy các chương trình, phần mềm của nhau.

- Đặc tính đa chương trình: Tại một thời điểm có thể có vài chương trình nằm trong bộ nhớ và một trong số đó được cho chạy trong khi các chương trình khác chờ hoàn thành các thao tác vào/ra.

- Không gian địa chỉ rất lớn.



Hình 1-3: IBM System/360 – Năm 1964

- **Thế hệ thứ tư:** (1980-). Máy tính được xây dựng trên các vi mạch cỡ lớn (LSI) và cực lớn (VLSI).

Đây là thế hệ máy tính số ngày nay, nhờ công nghệ bán dẫn phát triển vượt bậc, mà người ta có thể chế tạo các mạch tổ hợp ở mức độ cực lớn. Nhờ đó máy tính ngày càng nhỏ hơn, nhẹ hơn, mạnh hơn và giá thành rẻ hơn. Máy tính cá nhân bắt đầu xuất hiện và phát triển trong thời kỳ này.



Hình 1- 4 : IBM – Năm 1981

Dựa vào kích thước vật lý, hiệu suất và lĩnh vực sử dụng, hiện nay người ta thường chia máy tính số thế hệ thứ tư thành 5 loại chính, các loại có thể trùm lên nhau một phần:

- **Microcomputer:** Còn gọi là PC (personal computer), là những máy tính nhỏ, có 1 chip vi xử lý và một số thiết bị ngoại vi. Thường dùng cho một người, có thể dùng độc lập hoặc dùng trong mạng máy tính.

- **Minicomputer:** Là những máy tính cỡ trung bình, kích thước thường lớn hơn PC. Nó có thể thực hiện được các ứng dụng mà máy tính cỡ lớn thực hiện. Nó có khả năng hỗ trợ hàng chục đến hàng trăm người làm việc. Minicomputer được sử dụng rộng rãi trong các ứng dụng thời gian thực, ví dụ trong điều khiển hàng không, trong tự động hoá sản xuất.

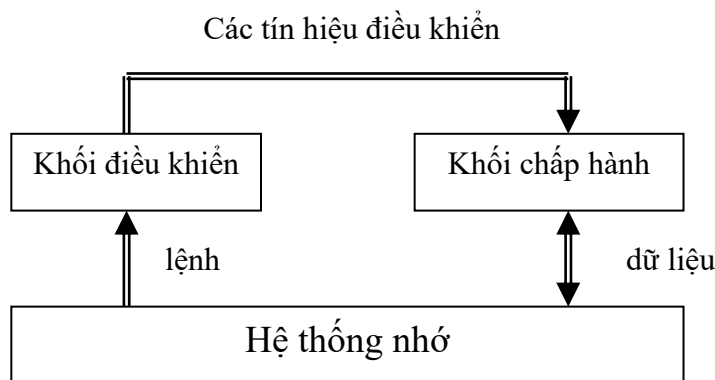
- **Supermini:** Là những máy Minicomputer có tốc độ xử lý nhanh nhất trong họ Mini ở những thời điểm nhất định. Supermini thường được dùng trong các hệ thống phân chia thời gian, ví dụ các máy quản gia của mạng.

- **Mainframe:** Là những máy tính cỡ lớn, có khả năng hỗ trợ cho hàng trăm đến hàng ngàn người sử dụng. Thường được sử dụng trong chế độ các công việc sắp xếp theo lô lớn (Large-Batch-Job) hoặc xử lý các giao dịch (Transaction Processing), ví dụ trong ngân hàng.

- **Supercomputer:** Đây là những siêu máy tính, được thiết kế đặc biệt để đạt tốc độ thực hiện các phép tính dấu phẩy động cao nhất có thể được. Chúng thường có kiến trúc song song, chỉ hoạt động hiệu quả cao trong một số lĩnh vực.

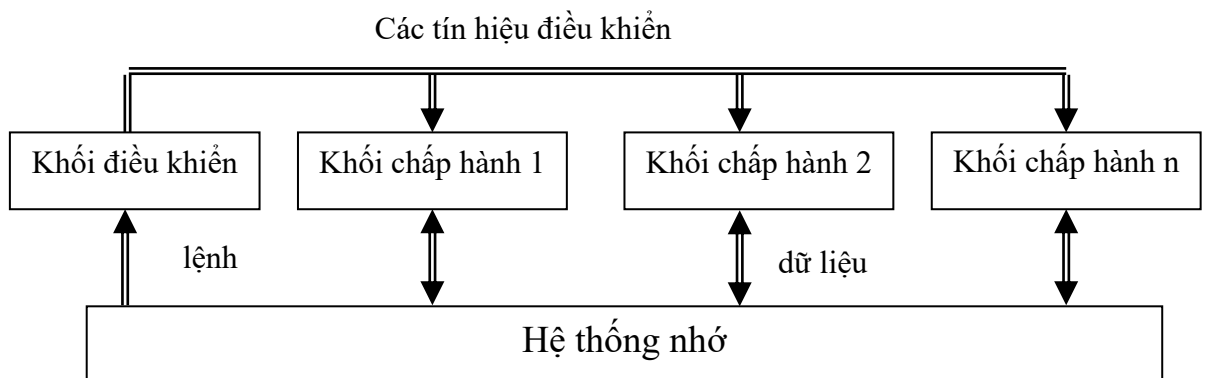
Dựa vào kiến trúc của máy tính người ta cũng phân máy tính ra các loại khác nhau như sau:

- **Kiến trúc SISD** (single instruction - single data, đơn dòng lệnh - đơn dòng dữ liệu), sơ đồ như hình 1-1.



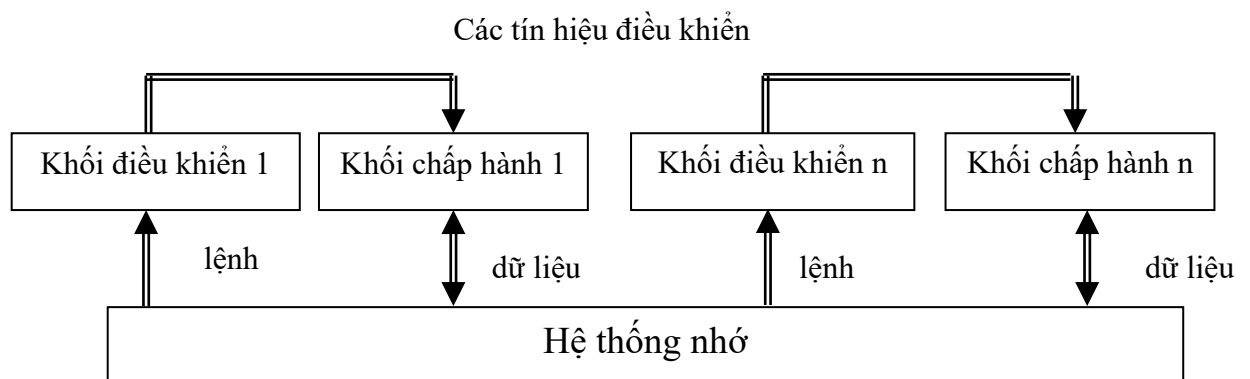
Hình 1-5: Kiến trúc máy tính SISD.

- Kiến trúc SIMD (Single Instruction Multiple Data, đơn dòng lệnh- đa dữ liệu), sơ đồ như hình 1-2.



Hình 1-6: Kiến trúc SIMD.

- Kiến trúc MIMD (Multiple Instruction Multiple Data, đa dòng lệnh- đa dữ liệu), sơ đồ như hình 1-3.



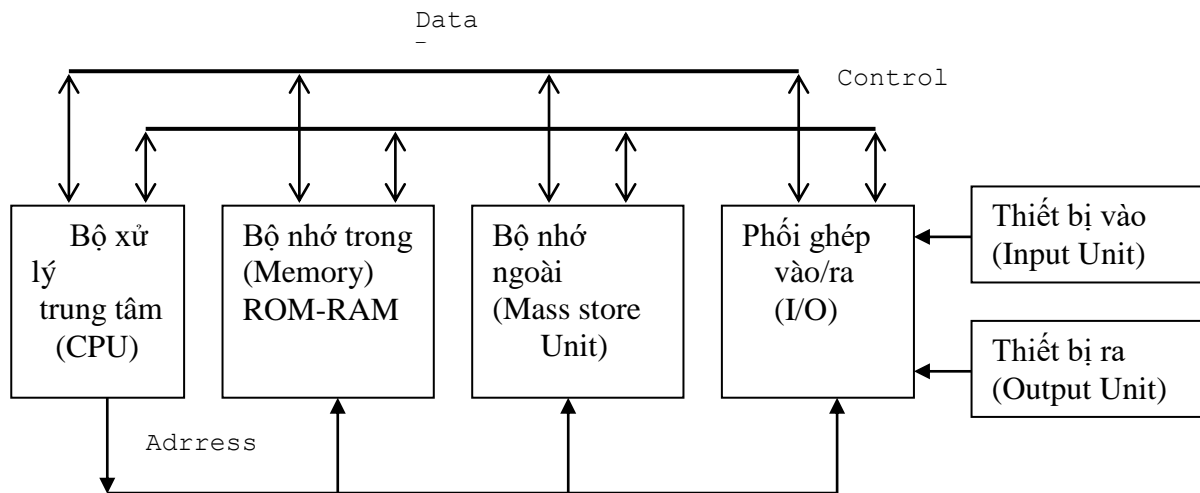
Hình 1-7: Kiến trúc MIMD.

1.4. Hệ thống máy tính

1.4.1. Các thành phần chính của máy tính

Giới thiệu sơ lược cấu trúc của máy vi tính.

Sở dĩ từ khi ra đời, cấu trúc cơ sở của các máy vi tính ngày nay không thay đổi mấy. Mọi máy tính số đều có thể coi như được hình thành từ sáu phần chính (như hình 1-8):



Hình 1-8: Giới thiệu sơ đồ khối tổng quát của máy tính số

Trong sơ đồ này, các khối chức năng chính của máy tính số gồm:

- Khối xử lý trung tâm (central processing unit, CPU),
- Bộ nhớ trong (memory), như RAM, ROM
- Bộ nhớ ngoài, như các loại ổ đĩa, băng từ
- Khối phối ghép với các thiết bị ngoại vi (vào/ra)
- Các bộ phận đầu vào, như bàn phím, chuột, máy quét ...
- Các bộ phận đầu ra, như màn hình, máy in ...

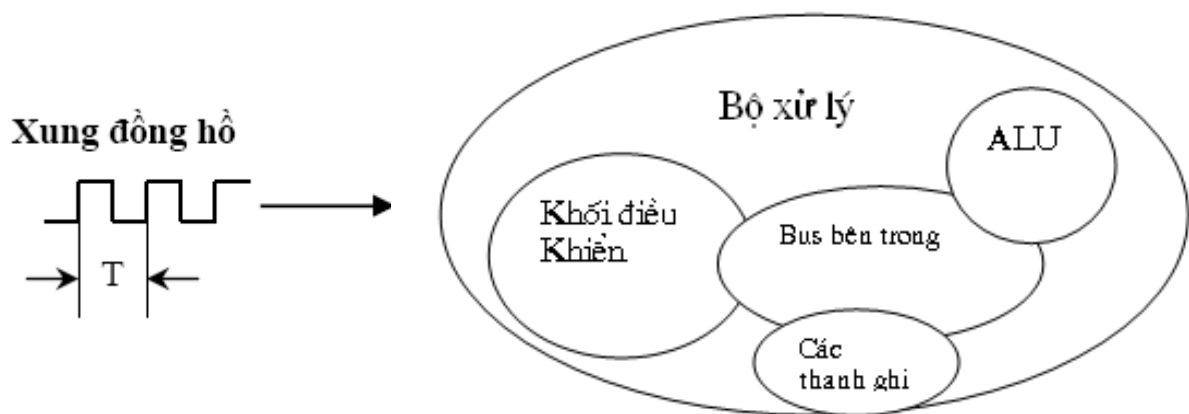
Bốn khối chức năng đầu liên hệ với nhau thông qua tập các đường dây để truyền tín hiệu, gọi chung là *bus hệ thống*. Bus hệ thống bao gồm 3 bus thành phần; ứng với các tín hiệu xác lập địa chỉ từ CPU đến các đơn vị thành phần ta có bus địa chỉ; với các dữ liệu được liên hệ giữa các khối qua bus dữ liệu (data bus); các tín hiệu điều khiển bao gồm các lệnh, các đáp ứng, các trạng thái của các khối được xác lập qua bus điều khiển.

Sự khác biệt quan trọng nhất của các hệ máy tính là kích thước và tốc độ, các máy tính nhỏ hơn và nhanh, mạnh hơn theo từng năm. Sự phát triển không ngừng của các thế hệ máy tính nhờ vào hai yếu tố quan trọng, đó là sự phát triển của công nghệ chế tạo IC và công nghệ chế tạo bộ nhớ.

a. CPU

Bộ xử lý trung tâm (CPU) có trách nhiệm xử lý hầu hết dữ liệu/tác vụ của máy tính, điều khiển thiết bị đầu vào (như chuột, bàn phím) cũng như thiết bị đầu ra (như màn hình, máy in).

Đúng như bạn tưởng tượng, tốc độ và hiệu suất của CPU là một trong những yếu tố quan trọng nhất giúp xác định một máy tính hoạt động tốt như thế nào. Về cơ bản, CPU là một tấm mạch rất nhỏ, bên trong chứa một tấm wafer silicon được bọc trong một con chip bằng gốm và gắn vào bảng mạch.



Hình 1-9: Bộ xử lý (CPU)

Tốc độ của CPU: Là số lệnh thực hiện được trong 1s, MIPS (millions of instruction per second). Trên thực tế tốc độ thường được đánh giá gián tiếp thông qua **Clock Frequency** - tần số xung đồng hồ nhịp cung cấp cho CPU làm việc

Tốc độ CPU được đo bằng đơn vị hertz (Hz) hay gigahertz (GHz), giá trị của con số này càng lớn thì CPU hoạt động càng nhanh.

Một hertz (Hz) được hiểu là một dao động trong mỗi giây, còn một gigahertz là 1 tỷ dao động trong mỗi giây. Tuy nhiên tốc độ CPU không chỉ được đo lường bằng giá trị Hz hay GHz bởi CPU của mỗi hãng sẽ có những công nghệ cải thiện hiệu năng khác nhau nhằm làm tăng thông lượng dữ liệu theo cách riêng.

Một sự so sánh công bằng hơn giữa các CPU khác nhau chính là số lệnh mà chúng có thể thực hiện mỗi giây.

Đến đây chắc bạn cũng đã phần nào hiểu được thuật ngữ CPU bị dùng sai ở Việt Nam khá nhiều. Mọi người thường dùng từ CPU để chỉ cái thùng máy (case) của chiếc máy tính để bàn truyền thống, nhưng thực chất CPU chỉ là một con chip rất

nhỏ bên trong, còn thùng máy thì chứa cả CPU, bo mạch chủ, RAM, ổ cứng, ổ quang và card đồ họa (nếu có).

b. RAM

RAM là một loại bộ nhớ, gọi là bộ nhớ truy cập ngẫu nhiên (RAM), tạo thành một không gian nhớ tạm để máy tính hoạt động. Tuy cũng gọi là bộ nhớ, nhưng bạn đừng lầm tưởng chúng chứa dữ liệu của mình bởi khi tắt máy tính thì RAM chẳng còn nhớ gì dữ liệu từng được máy tính lưu trên đó.

Hay nói cụ thể hơn, RAM chỉ là nơi tạm nhớ những gì cần làm để CPU có thể xử lý nhanh hơn do tốc độ truy xuất trên RAM nhanh hơn rất nhiều lần so với ổ cứng (nơi thật sự lưu dữ liệu của bạn) hay các thiết bị lưu trữ khác như thẻ nhớ, đĩa quang.

Bộ nhớ RAM càng nhiều thì máy tính của bạn có thể mở cùng lúc nhiều ứng dụng mà không bị chậm. Nhìn chung thì thêm RAM cũng có thể làm cho một số ứng dụng chạy tốt hơn.

Dung lượng bộ nhớ RAM hiện tại được đo bằng gigabyte (GB), 1GB tương đương 1 tỷ byte. Hầu hết máy tính thông thường ngày nay đều có ít nhất 2-4GB RAM, với các máy cao cấp thì dung lượng RAM có thể lên đến 16GB hoặc cao hơn.

Giống như CPU, bộ nhớ RAM bao gồm những tấm wafer silicon mỏng, bọc trong chip gốm và gắn trên bảng mạch. Các bảng mạch giữ các chip nhớ RAM hiện tại được gọi là DIMM (Dual In-Line Memory Module) do chúng tiếp xúc với bo mạch chủ bằng hai đường riêng biệt.

c. Ổ cứng

Ổ cứng là nơi lưu trữ hệ điều hành, phần mềm và mọi dữ liệu của bạn. Khi tắt nguồn, mọi thứ vẫn còn đó nên bạn không phải cài lại phần mềm hay mất dữ liệu khi tắt mở máy tính. Khi bật máy tính, hệ điều hành và ứng dụng sẽ được chuyển từ ổ cứng lên bộ nhớ RAM để chạy.

Dung lượng lưu trữ ổ cứng cũng được đo bằng gigabyte (GB) như bộ nhớ. Một ổ đĩa cứng thông thường hiện tại có thể chứa 500GB hoặc thậm chí 1 terabyte (1.000GB) hoặc hơn. Hầu hết ổ cứng được bán ngày nay là loại cơ khí truyền thống sử dụng đĩa kim loại để lưu trữ dữ liệu bằng từ tính. Bạn chắc cũng đã nghe nói đến hoặc đang sử dụng một loại mới hơn là SSD (hay gọi là ổ cứng rắn), sử dụng một loại bộ nhớ, dùng các chip nhớ chứ không có phần quay cơ học, cho tốc độ đọc ghi nhanh hơn nhiều, hoạt động yên tĩnh và độ tin cậy cao hơn nhưng giá của loại sản phẩm này còn khá đắt.

d. Thiết bị đầu vào

Một máy tính có thể đi kèm với một hoặc nhiều thiết bị đầu vào như chuột, touchpad, trackball (máy tính xách tay), bàn phím hay bảng vẽ.

e. Màn hình

Tùy thuộc vào loại máy tính, màn hình (monitor) hiển thị có thể được gắn liền (laptop, máy để bàn All-In-One), hoặc có thể là một đơn vị riêng biệt được gọi là một màn hình với dây nguồn riêng. Một số màn hình có tích hợp cảm ứng, vì vậy bạn có thể sử dụng ngón tay chạm trên màn hình để điều khiển tương tự như dùng điện thoại hay máy tính bảng. Với các máy tính để bàn truyền thống, màn hình nằm riêng biệt chỉ có nhiệm vụ hiển thị nên nếu có hỏng hóc thì bạn có thể yên tâm thay thế mà không lo mất dữ liệu hay phần mềm như một số người dùng vẫn lầm tưởng.

Chất lượng hiển thị được đo bằng độ phân giải, là số lượng điểm ảnh khi hiển thị ở độ phân giải cao nhất có thể. Ví dụ một màn hình máy tính xách tay có độ phân giải 1.920x1.080 pixel; số đầu tiên đại diện cho độ phân giải ngang và số thứ hai là độ phân giải dọc. Bạn có thể nhân hai số này để ra số lượng điểm ảnh và sau đó chia kích thước đường chéo (inch) màn hình để ra chỉ số mật độ điểm ảnh (dpi) mà bạn vẫn thường thấy trên các bài báo công nghệ hay chi tiết kỹ thuật quảng cáo các sản phẩm liên quan đến hiển thị.

Một yếu tố khác bạn cần quan tâm là tỷ lệ khung hình. Hiện tại có hai tiêu chuẩn là 4:3 (hay gọi là màn hình vuông – thực chất không phải hình vuông) và 16:9 (màn hình rộng hay màn hình wide, cũng là tiêu chuẩn của hầu hết nội dung video hiện nay).

Với thông số độ phân giải, bạn cũng có thể biết ngay một màn hình sở hữu khung hình dạng nào bằng cách rút gọn tỷ lệ độ phân giải ngang/độ phân giải dọc. Ví dụ, một màn hình có độ phân giải tối đa là 800x600, thì bạn lấy 800 chia cho 600, được giá trị 4/3 tức là tỷ lệ 4:3.

f. Ổ đĩa quang

Hầu hết máy tính để bàn và máy tính xách tay (ngoại trừ các máy dòng siêu mỏng hay quá nhỏ gọn) đều đi kèm với một ổ đĩa quang, nơi đọc/ghi đĩa CD, DVD, và Blu-ray (tùy thuộc máy).

Gọi bằng tên ổ đĩa quang là do cách chúng đọc ghi dữ liệu trên đĩa. Cụ thể, một đèn laser sẽ chiếu ánh sáng vào bề mặt, và một cảm biến sẽ đo lượng ánh sáng bật ngược trở lại từ một điểm nào đó trên đĩa và giải mã ra dữ liệu.

Ngày nay, với sự phát triển của tốc độ truy cập Internet thì hầu hết dữ liệu, phim ảnh đều có thể lưu trữ hoặc cài đặt từ các dịch vụ điện toán đám mây (hay nói cho dễ hiểu là một nơi lưu trữ trên Internet) nên vai trò của ổ đĩa quang cũng dần mờ nhạt.

g. Card mạng

Khi sở hữu máy tính, ắt hẳn bạn sẽ muốn dùng nó để kết nối Internet và điều đó có nghĩa là bạn muốn máy tính của mình sở hữu một card mạng.

Hầu hết máy tính ngày nay đều được tích hợp ít nhất một card mạng LAN (có dây hoặc không dây) trên bo mạch chủ để bạn có thể kết nối chúng với bộ định tuyến Internet (bộ định tuyến thường đi kèm dịch vụ Internet của các nhà mạng VNPT, Viettel, FPT).

1.4.2. Hoạt động của máy tính

Muốn máy tính làm một việc gì đó con người phải đưa vào máy tính một hay nhiều lệnh thông qua các thiết bị nhập.

Khi một lệnh được nhận từ các thiết bị nhập, lệnh này lập tức được truyền vào CPU (bộ xử lý trung tâm - Central Processing Unit). CPU phải phân tích lệnh và tìm kiếm trong bộ nhớ, trong những thiết bị lưu trữ, những công cụ và phương thức để thực hiện các lệnh đó. Nếu tìm thấy thì thực hiện lệnh và trả kết quả về thông qua bộ phận xuất. Nếu không tìm thấy thì sẽ trả về thông tin báo lỗi.

CHƯƠNG 2. CÁC HỆ ĐẾM VÀ BIỂU DIỄN THÔNG TIN TRONG MÁY TÍNH

2.1. Hệ nhị phân (Binary)

2.1.1. Khái niệm:

Hệ nhị phân (hay **hệ đếm cơ số hai**) là một hệ đếm dùng hai ký tự để biểu đạt một giá trị số, bằng tổng số các lũy thừa của 2. Hai ký tự đó thường là 0 và 1. Mỗi một con số nhị phân được gọi là một Bit (BInary digiT). Bit ngoài cùng bên trái là bit có trọng số lớn nhất (MSB, Most Significant Bit) và bit ngoài cùng bên phải là bit có trọng số nhỏ nhất (LSB, Least Significant Bit) như dưới đây:

	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	
MSB	1	1	1	0	.	1	1
							LSB

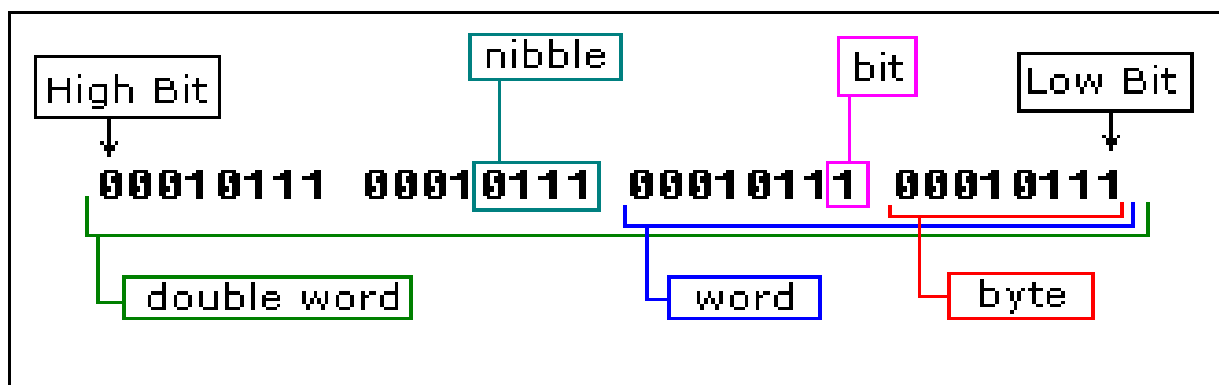
Chấm nhị phân

Số nhị phân $(1110.11)_2$ có thể biểu diễn thành:
 $(1110.11)_2 = 1*2^3 + 1*2^2 + 1*2^1 + 0*2^0 + 1*2^{-1} + 1*2^{-2} = (14.75)_{10}$.

Chú ý: dùng dấu ngoặc đơn và chỉ số dưới để ký hiệu cơ số của hệ đếm.

Mỗi ký số (digit) trong hệ binary number được gọi là BIT.

- 4 bits nhóm thành 1 NIBBLE
- 8 bits tạo thành 1 BYTE
- 2 bytes tạo thành 1 WORD
- 2 Words tạo thành 1 DOUBLE WORD (ít dùng):
-



2.1.2. Biến đổi từ nhị phân sang thập phân

Ví dụ : Biến đổi số nhị phân $(11101)_2$ thành số thập phân:

Trọng số vị trí:	2^4	2^3	2^2	2^1	2^0
Giá trị vị trí:	16	8	4	2	1
Số nhị phân:	1	1	1	0	1
Số thập phân:	$1*2^4 + 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = (29)_{10}$				

2.1.3. Biến đổi thập phân thành nhị phân

Để thực hiện việc đổi từ thập phân sang nhị phân, ta áp dụng phương pháp chia lặp như sau: lấy số thập phân chia cho cơ số để thu được một thương số và số dư. Số dư được ghi lại để làm một thành tố của số nhị phân. Sau đó, số thương lại được chia cho cơ số một lần nữa để có thương số thứ 2 và số dư thứ 2. Số dư thứ hai là con số nhị phân thứ hai. Quá trình tiếp diễn cho đến khi số thương bằng 0.

Ví dụ 1: Biến đổi số thập phân $(29)_{10}$ thành nhị phân:

$$35/2 = 17 \text{ dư } 1(\text{LSB})$$

$$17/2 = 8 \text{ dư } 1$$

$$8/2 = 4 \text{ dư } 0$$

$$4/2 = 2 \text{ dư } 0$$

$$2/2 = 1 \text{ dư } 0$$

$$1/2 = 0 \text{ dư } 1(\text{MSB})$$

Vậy $(35)_{10} = (100011)_2$.

Đối với phần lẻ của các số thập phân, số lẻ được nhân với cơ số và số nhớ được ghi lại làm một số nhị phân. Trong quá trình biến đổi, số nhớ đầu chính là **bit** MSB và số nhớ cuối là **bit** LSB.

Ví dụ 2: Biến đổi số thập phân $(0.625)_{10}$ thành nhị phân:

$$0.625 \times 2 = 1.250. \text{ Số nhớ là } 1, \text{ là } \mathbf{\text{bit}} \text{ MSB.}$$

$$0.250 \times 2 = 0.500. \text{ Số nhớ là } 0$$

$$0.500 \times 2 = 1.000. \text{ Số nhớ là } 1, \text{ là } \mathbf{\text{bit}} \text{ LSB.}$$

$$\text{Vậy : } (0.625)_{10} = (0.101)_2.$$

2.2. Hệ thập lục phân (Hexadecima).

2.1.1 Khái niệm:

Các hệ máy tính hiện đại thường dùng một hệ đếm khác là hệ thập lục phân. Hệ thập lục phân là hệ đếm dựa vào vị trí với cơ số là 16. Hệ này dùng các con số từ 0 đến 9 và các ký tự từ A đến F như trong bảng sau:

Bảng 2.1 Hệ thập lục phân:

Thập lục phân	Thập phân	Nhị phân
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010

B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

2.2.2. Biến đổi thập lục phân thành thập phân.

Các số thập lục phân có thể được biến đổi thành thập phân bằng cách tính tổng của các con số nhân với giá trị vị trí của nó.

Ví dụ :

a. Biến đổi các số $(7C)_{16}$ thành thập phân

Số thập lục phân: 7 C

Trọng số vị trí: 16^1 16^0

Giá trị vị trí : 16 1

Số thập phân: $7 \cdot 16 + C \cdot 1 = (124)_{10}$.

b. Biến đổi số $(A2B)_{16}$ thành thập phân.

Số thập lục phân: A 2 B

Trọng số vị trí: 16^2 16^1 16^0

Giá trị vị trí : 256 16 1

Số thập phân: $A \cdot 256 + 2 \cdot 16 + B \cdot 1 = (2603)_{10}$.

2.2.3. Biến đổi thập phân thành thập lục phân.

Để biến đổi các số thập phân thành thập lục phân, ta sử dụng phương pháp chia lặp, với cơ số 16.

Ví dụ : Biến đổi $(670)_{10}$ thành thập lục phân.

$670/16 = 41$ dư 14 hoặc E(LSB).

$41/16 = 2$ dư 9.

$2/16 = 0$ dư 2(MSB).

Số thập lục phân: $(29E)_{16}$.

2.2.4. Biến đổi thập lục phân thành nhị phân.

Các số thập lục phân rất dễ đổi thành nhị phân. Thực ra các số thập lục phân cũng chỉ là một cách biểu diễn các số nhị phân thuận lợi hơn mà thôi (bảng 2-1). Để đổi các số thập lục phân thành nhị phân, chỉ cần thay thế một cách đơn giản từng con số thập lục phân bằng bốn bit nhị phân tương đương của nó.

Ví dụ: Đổi số thập lục $(AF4)_{16}$ thành nhị phân:

A	F	4
↓	↓	↓
1010	1111	0100

$(AF4)_{16} = (101011110100)_2$.

2.2.5. Biến đổi nhị phân thành thập lục phân.

Để biến đổi một số nhị phân thành số thập lục phân tương đương thì chỉ cần gộp lại thành từng nhóm gồm 4 bit nhị phân, bắt đầu từ dấu chấm nhị phân.

Ví dụ: Biến đổi số nhị phân $(110111100001)_2$ thành thập lục phân.

1101	1110	0001	
↓	↓	↓	
D	E	1	Số thập lục phân: $(DE1)_{16}$.

2.3. Biểu diễn ký tự trong máy tính

2.3.1. Bảng mã ASCII.(American Standard Code for Information Interchange)

Viện Chuẩn Hóa Hoa Kỳ (American National Standards Institute) đã đưa ra bộ mã chuẩn trong giao tiếp thông tin trên máy tính gọi là bộ mã ASCII. Các mã đó gọi là **bộ mã ký tự và số**.

Bảng mã ASCII là mã 7 bit được dùng phổ biến trong các hệ máy tính hiện nay. Với mã 7 bit nên có $2^7 = 128$ tổ hợp mã. Mỗi ký tự (chữ hoa và chữ thường) cũng như các con số thập phân từ 0..9 và các ký hiệu đặc biệt khác đều được biểu diễn bằng một mã số như bảng 2-2.

Việc biến đổi thành ASCII và các mã ký tự số khác, tốt nhất là sử dụng mã tương đương trong bảng.

Ví dụ: dãy bit sau là biểu diễn của chuỗi ký tự "Hi Sue " :

1001000 1101001 0100000 1010011 1110101 1100101 0100000						
H	I	space	S	u	e	space

Bảng 2-2: Mã ASCII.

ASCII value	Character	Control character	ASCII value	Character	ASCII value	Character	ASCII value	Character
000	(null)	NUL	032	(space)	064	@	096	
001	☺	SOH	033	!	065	A	097	a
002	☼	STX	034	"	066	B	098	b
003	♥	ETX	035	#	067	C	099	c
004	♦	EOT	036	\$	068	D	100	d
005	♣	ENQ	037	%	069	E	101	e
006	♠	ACK	038	&	070	F	102	f
007	(beep)	BEL	039	'	071	G	103	g
008	■	BS	040	(072	H	104	h
009	(tab)	HT	041)	073	I	105	i
010	(line feed)	LF	042	*	074	J	106	j
011	(home)	VT	043	+	075	K	107	k
012	(form feed)	FF	044	,	076	L	108	l
013	(carriage return)	CR	045	-	077	M	109	m
014	♪	SO	046	.	078	N	110	n
015	☼	SI	047	/	079	O	111	o
016	▲	DLE	048	0	080	P	112	p
017	▴	DC1	049	1	081	Q	113	q
018	↕	DC2	050	2	082	R	114	r
019	!!!	DC3	051	3	083	S	115	s
020	π	DC4	052	4	084	T	116	t
021	\$	NAK	053	5	085	U	117	u
022	▬	SYN	054	6	086	V	118	v
023	↕	ETB	055	7	087	W	119	w
024	↕	CAN	056	8	088	X	120	x
025	↕	EM	057	9	089	Y	121	y
026	→	SUB	058	:	090	Z	122	z
027	←	ESC	059	;	091	[123	{
028	(cursor right)	FS	060	<	092	\	124	
029	(cursor left)	GS	061	=	093]	125	}
030	(cursor up)	RS	062	>	094	^	126	~
031	(cursor down)	US	063	?	095	_	127	☐

Copyright 1998, JimPrice.Com Copyright 1982, Loading Edge Computer Products, Inc.

NUL	Null	FF	Form feed	CAN	Cancel
SOH	Start of heading	CR	Carriage return	EM	End of medium
STX	Start of text	SO	Shift out	SUB	Substitute
ETX	End of text	SI	Shift in	ESC	Escape
EOT	End of transmission	DLE	Data link escape	FS	File separator
ENQ	Enquiry	DC1	Device control 1	GS	Group separator
ACK	Acknowledge	DC2	Device control 2	RS	Record separator
BEL	Bell	DC3	Device control 3	US	Unit separator
BS	Backspace	DC4	Device control 4	SP	Space
HT	Horizontal tab	NAK	Negative acknowledge	DEL	Delete
LF	Line feed	SYN	Synchronous idle		
VT	Vertical tab	ETB	End of transmission block		

Bảng mã ASCII

2.3.2. Bảng mã Unicode

Bảng mã Unicode (hay còn gọi là bảng mã thống nhất, mã đơn nhất) : là bộ mã chuẩn quốc tế được thiết kế để dùng làm bộ mã duy nhất cho tất cả các ngôn ngữ khác nhau trên thế giới, kể cả các ngôn ngữ sử dụng ký tự tượng hình phức tạp như tiếng Trung Quốc, tiếng Việt Nam, tiếng Thái Lan ... Vì những điểm ưu việt đó, Unicode đã và đang từng bước thay thế các bộ mã truyền thống, kể cả bộ mã tiêu chuẩn ISO 8859 và hiện đang được hỗ trợ trên rất nhiều phần mềm cũng như các trình ứng dụng, chẳng hạn Windows.

Mã ASCII dùng các giá trị (mã) từ 0 - 127 để miêu tả các ký tự :

- mã từ 0 - 31 là các mã điều khiển như CR=13 (Carriage Return), LF=10 (Line Feed), ESC=27 (Escape)...
- mã 32 miêu tả ký tự trống, 33 miêu tả ký tự !,... theo bảng sau :

	!	"	#	\$	%	&	'	()	*	+	,	-	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{		}	~	

Sự khác biệt giữa các bảng mã là :

UTF 8 : UTF-8 là một cách mã hóa để có tác dụng giống như UCS-4 (cũng là UTF-16), chứ không phải có code point nào khác. UTF-8 được thiết kế để tương thích với chuẩn ASCII. UTF-8 có thể sử dụng từ một (cho những ký tự trong ASCII) cho đến 6 byte để biểu diễn một ký tự.

Chính vì tương thích với ASCII, UTF-8 cực kỳ có lợi thế khi được sử dụng để bổ sung hỗ trợ Unicode cho các phần mềm có sẵn. Thêm vào đó, các nhà phát triển phần mềm vẫn có thể sử dụng các hàm thư viện có sẵn của ngôn ngữ lập trình C để so sánh (comparisons) và xếp thứ tự. (Ngược lại, để hỗ trợ các cách mã hóa 16 bit hay 32 bit như ở trên, một số lớn phần mềm buộc phải viết lại do đó tốn rất nhiều công sức. Một điểm mạnh nữa của UTF-8 là với các văn bản chỉ có một số ít các ký tự ngoài ASCII, hay thậm chí cho các ngôn ngữ dùng bảng chữ cái Latinh như tiếng Việt, tiếng Anh, tiếng Đức ...; cách mã hóa kiểu này cực kỳ tiết kiệm không gian lưu trữ.

UTF-8 được thiết kế đảm bảo không có chuỗi byte của ký tự nào lại nằm trong một chuỗi của ký tự khác dài hơn. Điều này khiến cho việc tìm kiếm ký tự theo byte trong một văn bản là rất dễ dàng. Một số dạng mã hóa khác (như Shift-JIS) không có tính chất này khiến cho việc xử lý chuỗi ký tự trở nên phức tạp hơn nhiều. Mặc dù để thực hiện điều này đòi hỏi phải có độ dư (văn bản sẽ dài thêm) nhưng những ưu điểm mà nó mang lại vẫn nhiều hơn. Việc nén dữ liệu không phải là mục đích hướng tới của Unicode và việc này cần được tiến hành một cách độc lập.

Các quy định chính xác của UTF-8 như sau (các số bắt đầu bằng 0x là các số biểu diễn trong hệ thập lục phân)

- Các ký tự có giá trị nhỏ hơn 0x80, sử dụng 1 byte có cùng giá trị.
- Các ký tự có giá trị nhỏ hơn 0x800, sử dụng 2 byte: byte thứ nhất có giá trị 0xC0 cộng với 5 bit từ thứ 7 tới 11 (7th-11th least significant bits); byte thứ hai có giá trị 0x80 cộng với các bit từ thứ 1 tới thứ 6 (1st-6th least significant bits).

- Các ký tự có giá trị nhỏ hơn 0x10000, sử dụng 3 byte: byte thứ nhất có giá trị 0xE0 cộng với 4 bit từ thứ 13 tới 16; byte thứ hai có giá trị 0x80 cộng với 6 bit từ thứ 7 tới 12; byte thứ ba có giá trị 0x80 cộng với 6 bit từ thứ 1 tới thứ 6.

- Các ký tự có giá trị nhỏ hơn 0x200000, sử dụng 4 byte: byte thứ nhất có giá trị 0xF0 cộng với 3 bit từ thứ 19 tới 21; byte thứ hai có giá trị 0x80 cộng với 6 bit từ thứ 13 tới 18; byte thứ ba có giá trị 0x80 cộng với 6 bit từ thứ 7 tới thứ 12; byte thứ tư có giá trị 0x80 cộng với 6 bit từ thứ 1 tới thứ 6.

Hiện nay, các giá trị khác ngoài các giá trị trên đều chưa được sử dụng. Tuy nhiên, các chuỗi ký tự dài tới 6 byte có thể được dùng trong tương lai.

- Chuỗi 5 byte sẽ lưu trữ được mã ký tự chứa đến 26 bit: byte thứ nhất có giá trị 0xF8 cộng với 2 bit thứ 25 và 26, các byte tiếp theo lưu giá trị 0x80 cộng với 6 bit có ý nghĩa tiếp theo.

- Chuỗi 6 byte sẽ lưu trữ được mã ký tự chứa đến 31 bit: byte thứ nhất có giá trị 0xFC cộng với bit thứ 31, các byte tiếp theo lưu giá trị 0x80 cộng với 6 bit có ý nghĩa tiếp theo.

Mã ISO8859-1 dùng các giá trị (mã) từ 0 - 255 để miêu tả các ký tự (128 mã ký tự đầu qui định giống như mã ASCII) :

- mã từ 0 - 31 là các mã điều khiển như CR=13 (Carriage Return), LF=10 (Line Feed), ESC=27 (Escape)...
- mã 32 miêu tả ký tự trống, 33 miêu tả ký tự !,... theo bảng sau :


	!	"	#	\$	%	&	'	()	*	+	,	-	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{		}	~	
€	□	,	f	„	...	†	‡	^	%	Š	<	Œ	□	Ž	□	□	'	'	"	"	•	-	-	~	™	š	>	œ	□	ž	Ÿ
	ı	ç	£	¤	¥	!	§	"	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	,	¹	º	»	¼	½	¾	¿
À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

UTF 32 :

Cách đơn giản nhất để lưu trữ tất cả các 220+216 Unicode code points là sử dụng 32 bit cho mỗi ký tự, nghĩa là, 4 byte – do đó, cách mã hóa này được Unicode gọi là UTF 32 và ISO/IEC 10646 gọi là UCS-4 . Vấn đề chính của cách này là nó hao chỗ hơn 4 lần so với trước kia, do đó nó ít được dùng trong các vật nhớ ngoài (như đĩa, băng). Tuy nhiên, nó rất đơn giản, nên một số chương trình sẽ sử dụng mã hóa 32 bit bên trong khi xử lý Unicode.

Mã ĐHBK 1 byte có được bằng cách hiệu chỉnh bảng mã ISO8859-1 :

- mã từ 0 - 31 là các mã điều khiển như CR=13 (Carriage Return), LF=10 (Line Feed), ESC=27 (Escape)...
- mã 32 miêu tả ký tự trống, 33 miêu tả ký tự !,... theo bảng sau :



	!	"	#	\$	%	&	'	()	*	+	,	-	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	Ÿ	_
Ÿ	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	Ÿ	Ÿ	Đ	Ấ	
Á	À	Â	Ã	Ä	Å	Æ	È	É	Ê	Ë	Ē	Ē	Ē	Ē	Ē	İ	Í	Î	Ï	Ĭ	Ó	Ò	Ô	Õ	Ö	Ů	Ú	Û	Ü	Ý	À
Á	À	Â	Ã	Ä	Å	Æ	È	É	Ê	Ë	Ē	Ē	Ē	Ē	Ē	İ	Í	Î	Ï	Ĭ	Ó	Ò	Ô	Õ	Ö	Ů	Ú	Û	Ü	Ý	À
á	à	â	ã	ä	å	æ	è	é	ê	ë	ē	ē	ē	ē	ē	ı	í	î	ï	ĭ	ó	ò	ô	õ	ö	ů	ú	û	ü	ý	à
á	à	â	ã	ä	å	æ	è	é	ê	ë	ē	ē	ē	ē	ē	ı	í	î	ï	ĭ	ó	ò	ô	õ	ö	ů	ú	û	ü	ý	à

UTF 16 :

UTF-16 là một cách mã hóa dùng Unicode 20 bit. Các ký tự trong BMP được diễn tả bằng cách dùng giá trị 16-bit của code point trong Unicode CCS. Có hai cách để viết giá trị 16 bit trong một dòng (stream) 8-bit. Có lẽ bạn đã nghe qua chữ endian. Big Endian có nghĩa là cho Most Significant Byte đi trước, tức là nằm bên trái – do đó ta có UTF-16BE. Còn Little Endian thì ngược lại, tức là Least Significant Byte đi trước – do đó ta có UTF-16LE. Thí dụ, giá trị 16-bit của con số Hex1234 được viết là Hex12 Hex34 trong Big Endian và Hex34 Hex12 trong Little Endian.

Những ký hiệu không nằm trong BMP được biểu diễn bằng cách dùng surrogate pair (cặp thay thế). Code points có giá trị từ U+D800 đến U+DFFF được dành riêng ra để dùng cho mục đích này. Trước hết, một code point có 20 bit được phân ra làm hai nhóm 10 bit. Nhóm Most Significant 10 bit được map vào một giá trị 10 bit nằm trong khoảng từ u+D800 đến u+DBFF. Nhóm Least Significant 10 bit được map vào một giá trị 10 bit nằm trong khoảng từ U+DC00 đến U+DFFF. Theo cách đó UTF-16 có thể biểu diễn được những ký hiệu Unicode có 20 bit.

Mã Unicode Windows dùng 2 byte để miêu tả 1 ký tự :

- 256 mã đầu từ 0 - 255 giống y như mã ISO8859-1.
- mã từ 256 trở đi chứa các ký tự của hầu hết các ngôn ngữ trên thế giới (quá khứ, hiện tại và tương lai).
- thí dụ sau là 1 phần mã tiếng Việt trong mã Unicode :

mã 1ea0_H biểu diễn ký tự A

mã 1ef9_H biểu diễn ký tự ý

À	á	Â	à	Ã	ä	Ä	å	Å	â	Ă	ă	Ą	ą	Ǻ	ǻ	Ǽ	ǽ	Ǿ	ǿ	Ȧ	ȧ	Ė	ė	Ë	ë	Ẻ	ẻ
Ê	ê	Ë	ë	Ẹ	ẹ	Ĭ	ĩ	Į	į	ᄀ	ᄁ	ᄂ	ᄃ	ᄄ	ᄅ	ᄆ	ᄇ	ᄈ	ᄉ	ᄊ	ᄋ	ᄌ	ᄍ	ᄎ	ᄏ		
Œ	œ	Ɔ	ɔ	Ư	ư	Ủ	ủ	Ừ	ừ	Ứ	ứ	Ự	ự	Ỡ	ỡ	Ỗ	ỗ	Ỗ	ỗ	Ỗ	ỗ	Ỗ	ỗ	Ỗ	ỗ	Ỗ	ỗ

UTF 7 :

Chuẩn hóa được ít dùng nhất có lẽ là UTF-7. Chuẩn MIME yêu cầu mọi thư điện tử phải được gửi dưới dạng ASCII cho nên các thư điện tử nào sử dụng mã hóa Unicode được coi là không hợp lệ. Tuy nhiên hạn chế này thường bị hầu hết mọi người bỏ qua. UTF-8 cho phép thư điện tử sử dụng Unicode và đồng thời cũng phù hợp với tiêu chuẩn. Các ký hiệu ASCII sẽ được giữ nguyên, tuy nhiên các ký tự khác ngoài 128 ký hiệu ASCII chuẩn sẽ được mã hóa bằng một sequence hay một dấu ‘+’ theo sau một ký tự Unicode được mã hóa bằng Base64, và kết thúc bằng một dấu ‘-’. Ký tự ‘+’ nổi tiếng sẽ được mã hóa thành ‘+-’.

Bảng 2-3: Mã Unicode

0000	NUL	0020	SP	0040	@	0060	`	0080	Ctrl	00A0	NBS	00C0	À	00E0	à
0001	SOH	0021	!	0041	A	0061	a	0081	Ctrl	00A1	¡	00C1	Á	00E1	á
0002	STX	0022	"	0042	B	0062	b	0082	Ctrl	00A2	¢	00C2	Â	00E2	â
0003	ETX	0023	#	0043	C	0063	c	0083	Ctrl	00A3	£	00C3	Ã	00E3	ã
0004	EOT	0024	\$	0044	D	0064	d	0084	Ctrl	00A4	¤	00C4	Ä	00E4	ä
0005	ENQ	0025	%	0045	E	0065	e	0085	Ctrl	00A5	¥	00C5	Å	00E5	å
0006	ACK	0026	&	0046	F	0066	f	0086	Ctrl	00A6	¦	00C6	Æ	00E6	æ
0007	BEL	0027	'	0047	G	0067	g	0087	Ctrl	00A7	§	00C7	Ç	00E7	ç
0008	BS	0028	(0048	H	0068	h	0088	Ctrl	00A8	¨	00C8	È	00E8	è
0009	HT	0029)	0049	I	0069	i	0089	Ctrl	00A9	©	00C9	É	00E9	é
000A	LF	002A	*	004A	J	006A	j	008A	Ctrl	00AA	ª	00CA	Ê	00EA	ê
000B	VT	002B	+	004B	K	006B	k	008B	Ctrl	00AB	«	00CB	Ë	00EB	ë
000C	FF	002C	,	004C	L	006C	l	008C	Ctrl	00AC	¬	00CC	Ì	00EC	ì
000D	CR	002D	-	004D	M	006D	m	008D	Ctrl	00AD		00CD	Í	00ED	í
000E	SO	002E	.	004E	N	006E	n	008E	Ctrl	00AE	®	00CE	Î	00EE	î
000F	SI	002F	/	004F	O	006F	o	008F	Ctrl	00AF	¯	00CF	Ï	00EF	ï
0010	DLE	0030	0	0050	P	0070	p	0090	Ctrl	00B0	°	00D0	Ð	00F0	ð
0011	DC1	0031	1	0051	Q	0071	q	0091	Ctrl	00B1	±	00D1	Ñ	00F1	ñ
0012	DC2	0032	2	0052	R	0072	r	0092	Ctrl	00B2	²	00D2	Ò	00F2	ò
0013	DC3	0033	3	0053	S	0073	s	0093	Ctrl	00B3	³	00D3	Ó	00F3	ó
0014	DC4	0034	4	0054	T	0074	t	0094	Ctrl	00B4	´	00D4	Ô	00F4	ô
0015	NAK	0035	5	0055	U	0075	u	0095	Ctrl	00B5	µ	00D5	Õ	00F5	õ
0016	SYN	0036	6	0056	V	0076	v	0096	Ctrl	00B6	¶	00D6	Ö	00F6	ö
0017	ETB	0037	7	0057	W	0077	w	0097	Ctrl	00B7	·	00D7	×	00F7	×
0018	CAN	0038	8	0058	X	0078	x	0098	Ctrl	00B8	,	00D8	Ø	00F8	ø
0019	EM	0039	9	0059	Y	0079	y	0099	Ctrl	00B9	¸	00D9	Ù	00F9	ù
001A	SUB	003A	:	005A	Z	007A	z	009A	Ctrl	00BA		00DA	Ú	00FA	ú
001B	ESC	003B	;	005B	[007B	{	009B	Ctrl	00BB	»	00DB	Û	00FB	û
001C	FS	003C	<	005C	\	007C		009C	Ctrl	00BC	¼	00DC	Ü	00FC	ü
001D	GS	003D	=	005D]	007D	}	009D	Ctrl	00BD	½	00DD	Ý	00FD	ý
001E	RS	003E	>	005E	^	007E	~	009E	Ctrl	00BE	¾	00DE	ÿ	00FE	ÿ
001F	US	003F	?	005F	_	007F	DEL	009F	Ctrl	00BF	¿	00DF	Ş	00FF	ş

NUL	Null	SOH	Start of heading	CAN	Cancel	SP	Space
STX	Start of text	EOT	End of transmission	EM	End of medium	DEL	Delete
ETX	End of text	DC1	Device control 1	SUB	Substitute	Ctrl	Control
ENQ	Enquiry	DC2	Device control 2	ESC	Escape	FF	Form feed
ACK	Acknowledge	DC3	Device control 3	FS	File separator	CR	Carriage return
BEL	Bell	DC4	Device control 4	GS	Group separator	SO	Shift out
BS	Backspace	NAK	Negative acknowledge	RS	Record separator	SI	Shift in
HT	Horizontal tab	NBS	Non-breaking space	US	Unit separator	DLE	Data link escape
LF	Line feed	ETB	End of transmission block	SYN	Synchronous idle	VT	Vertical tab

Bảng mã UNICODE

2.4. Biểu diễn giá trị số trong máy tính.

2.4.1 Biểu diễn số nguyên.

a. Biểu diễn số nguyên không dấu:

Tất cả các số cũng như các mã ... trong máy vi tính đều được biểu diễn bằng các chữ số nhị phân.

Để biểu diễn các số nguyên không dấu, người ta dùng n bit. Tương ứng với độ dài của số bit được sử dụng, ta có các khoảng giá trị xác định như sau:

Số bit	Khoảng giá trị
n bit:	0.. $2^n - 1$
8 bit	0.. 255 Byte
16 bit	0.. 65535 Word

b. Biểu diễn số nguyên có dấu:

Người ta sử dụng bit cao nhất biểu diễn dấu; bit dấu có giá trị 0 tương ứng với số nguyên dương, bit dấu có giá trị 1 biểu diễn số âm. Như vậy khoảng giá trị số được biểu diễn sẽ được tính như sau:

Số bit	Khoảng giá trị:
n bit	$2^{n-1}-1$
8 bit	-128.. 127 Short integer
16 bit	-32768.. 32767 Integer
32 bit	$-2^{31}.. 2^{31}-1$ (-2147483648.. 2147483647) Long integer

2.4.2. Biểu diễn số thực

Có hai cách biểu diễn số thực trong một hệ nhị phân: số có dấu chấm cố định (fixed point number) và số có dấu chấm động (floating point number). Cách thứ nhất được dùng trong những bộ VXL(micro processor) hay những bộ vi điều khiển (micro controller) cũ. Cách thứ 2 hay được dùng hiện nay có độ chính xác cao. Đối với cách biểu diễn số thực dấu chấm động có khả năng hiệu chỉnh theo giá trị của số thực. Cách biểu diễn chung cho mọi hệ đếm như sau:

$$R = m.B^e.$$

Trong đó m là phần định trị, trong hệ thập phân giá trị tuyệt đối của nó phải luôn nhỏ hơn 1. Số e là phần mũ và B là cơ số của hệ đếm.

Có hai chuẩn định dạng dấu chấm động quan trọng là: chuẩn MSBIN của Microsoft và chuẩn IEEE. Cả hai chuẩn này đều dùng hệ đếm nhị phân.

Thường dùng là theo tiêu chuẩn biểu diễn số thực của IEEE 754-1985(Institute of Electric & Electronic Engineers), là chuẩn được mọi hãng chấp nhận và được dùng trong bộ xử lý toán học của Intel. Bit dấu nằm tại vị trí cao nhất; kích thước phần mũ và khuôn dạng phần định trị thay đổi theo từng loại số thực.

Giá trị số thực IEEE được tính như sau:

$$R = (-1)^S * (1 + M_1 * 2^{-1} + \dots + M_n * 2^{-n}) * 2^{E 7 \dots E 0 - 127}.$$

Chú ý: giá trị đầu tiên M_0 luôn mặc định là 1.

- Dùng 32 bit để biểu diễn số thực, được số thực ngắn: $-3,4.10^{38} < R < 3,4.10^{38}$

31	30	23	22	0
S	E7 - E0	Định trị (M1 - M23)		

- Dùng 64 bit để biểu diễn số thực, được số thực dài: $-1,7.10^{308} < R < 1,7.10^{308}$

63	62	52	51	0
S	E10 - E0	Định trị (M1 - M52)		

Ví dụ tính số thực:

0100 0010 1000 1100 1110 1001 1111 1100

Phần định trị: $2^{-4}+2^{-5}+2^{-8}+2^{-9}+2^{-10}+2^{-12}+2^{-15}+2^{-16}+2^{-17}+2^{-18}+2^{-19}+2^{-20}+2^{-21} = 0,1008906$.

Giá trị ngầm định là: 1,1008906.

Phần mũ: $2^8+2^2+2^0=133$

Giá trị thực (bit cao nhất là bit dấu): $133-128=6$.

Dấu: 0 = số dương

Giá trị số thực là: $R = 1,1008906.2^6 = 70,457$.

Phương pháp đổi số thực sang số dấu phẩy động 32 bit:

- Đổi số thập phân thành số nhị phân.
- Biểu diễn số nhị phân dưới dạng $\pm 1, xxxBy$ (B: cơ số 2).
- Bit cao nhất 31: lấy giá trị 0 với số dương, 1 với số âm.
- Phần mũ y đổi sang mã excess -127 của y, được xác định bằng cách: $y + (7F)_{16}$.
- Phần xxx là phần định trị, được đưa vào từ bit 22..0.

Ví dụ: Biểu diễn số thực $(9,75)_{10}$ dưới dạng dấu phẩy động.

Ta đổi sang dạng nhị phân: $(9,75)_{10} = (1001.11)_2 = 1,00111B3$.

Bit dấu: bit 31 = 0.

Mã excess - 127 của 3 là: $7F + 3 = (82)_{16} = 82H = (10000010)_2$. Được đưa vào các bit tiếp theo: từ bit 30 đến bit 23.

Bit 22 luôn mặc định là 0.

Cuối cùng số thực $(9,75)_{10}$ được biểu diễn dưới dạng dấu phẩy động 32 bit như sau:

0100 0001 0001 1100 0000 0000 0000 0000
bit |31|30 23|22 0|

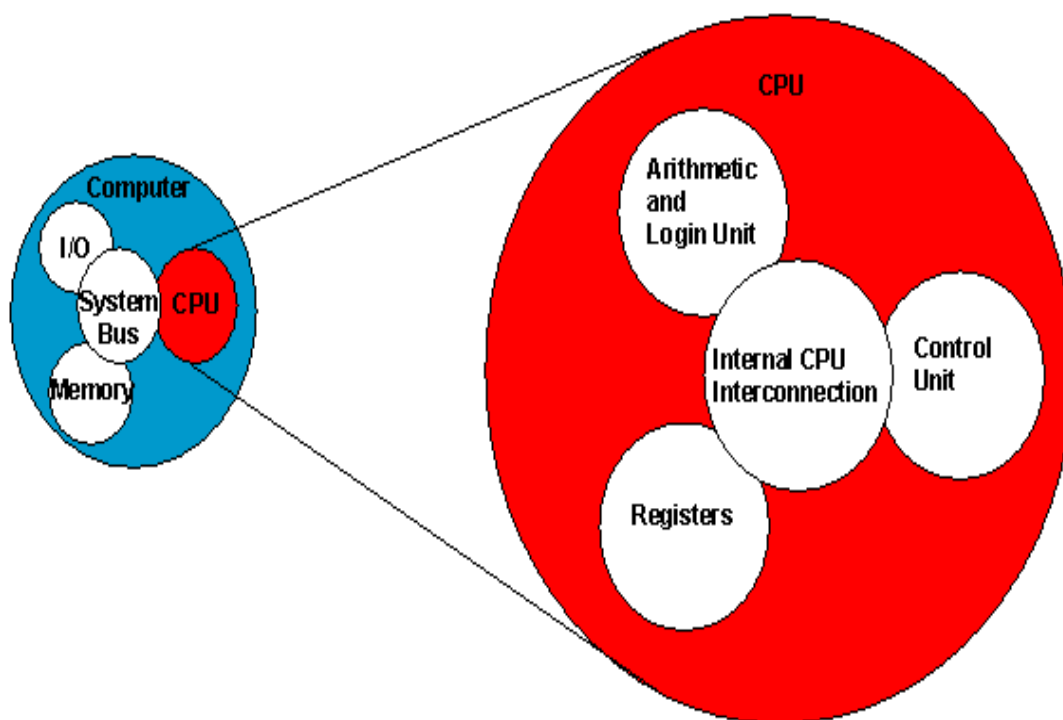
CHƯƠNG 3. BỘ XỬ LÝ TRUNG TÂM (CPU)

3.1. Khái niệm

Bộ xử lý trung tâm CPU là cốt lõi của một máy vi tính. CPU thực hiện mọi tính toán và xử lý của hệ thống - ngoại trừ xử lý tăng cường tính toán đặc biệt trong những hệ thống có một chip đơn vị đồng xử lý toán, mà chip này cũng đã được tích hợp ngay trong các CPU hiện nay. Tất cả những máy tính IBM và tương thích IBM sử dụng những bộ xử lý họ Intel hoặc tương thích với bộ xử lý họ Intel, dù chính những bộ xử lý có thể đã được nhiều công ty khác nhau thiết kế và sản xuất, gồm AMD, IBM, Cyrix...

Một trong những bộ xử lý điển hình thuộc họ 80x86 của Intel là bộ xử lý 8088. Đây là bộ vi xử lý khá đơn giản và vì vậy việc tìm hiểu nó là tương đối dễ đối với những người bắt đầu thâm nhập vào lĩnh vực vi xử lý, mặt khác việc nắm vững các vấn đề kỹ thuật của bộ vi xử lý 8088 sẽ là cơ sở để nắm bắt được các kỹ thuật của các bộ xử lý khác trong họ 80x86 của Intel, của các họ khác và của các bộ xử lý hiện đại ngày nay.

3.2. Giới thiệu cấu trúc bên trong của bộ vi xử lý 8088.



Hình 3-1 là sơ đồ khối cấu trúc bên trong của bộ vi xử lý 8088.

3.2.1. Đơn vị giao diện bus (BIU).

Theo sơ đồ khối trên hình 3-1 ta thấy bên trong CPU 8088 có hai khối chính: *khối phối ghép bus* (bus interface unit, BIU) và *khối thực hiện lệnh* (execution unit, EU). Việc chia CPU thành hai phần đồng thời có liên hệ với nhau qua đệm lệnh làm tăng đáng kể tốc độ xử lý của CPU. Các bus bên trong CPU có nhiệm vụ

chuyên tải tín hiệu của các khối khác. Trong số các bus có bus dữ liệu 16 bit của ALU, bus các tín hiệu điều khiển ở EU và bus trong của hệ thống ở BIU. Trước khi đi ra bus ngoài hoặc đi vào bus trong của bộ vi xử lý, các tín hiệu truyền trên bus thường được cho đi qua các bộ đệm để nâng cao tính tương thích cho nối ghép hoặc nâng cao khả năng phối ghép.

BIU bao gồm các thanh ghi đoạn (segment registers: CS, DS, SS, ES), con trỏ lệnh IP (instruction pointer) và bộ điều khiển logic bus (bus control logic, BCL). Đơn vị giao diện BIU còn có bộ nhớ đệm cho mã lệnh. Bộ nhớ này có chiều dài 4 byte (trong 8088) và 6 byte (trong 8086). Bộ nhớ đệm mã lệnh được nối với khối điều khiển CB (control block) của đơn vị thực hiện lệnh EU. Bộ nhớ này lưu trữ tạm thời mã lệnh trong một dãy gọi là hàng đợi lệnh. Hàng đợi lệnh cho phép bộ vi xử lý có khả năng xử lý xen kẽ liên tục dòng mã lệnh (pipelining). Hoạt động của bộ CPU được chia làm ba giai đoạn: đọc mã lệnh (operation code fetching), giải mã lệnh (decoding) và thực hiện lệnh (execution).

BIU đưa ra địa chỉ, đọc mã lệnh từ bộ nhớ, đọc/ghi dữ liệu từ các cổng vào hoặc bộ nhớ. Nói cách khác BIU chịu trách nhiệm đưa địa chỉ ra bus và trao đổi dữ liệu với bus.

3.2.2. Đơn vị thực hiện lệnh (EU-Execution Unit).

Trong EU có khối điều khiển (control unit, CU). Chính tại bên trong khối điều khiển này có mạch giải mã lệnh. Mã lệnh đọc vào từ bộ nhớ được đưa đến đầu vào của bộ giải mã, các thông tin thu được từ đầu ra của nó sẽ được đưa đến mạch tạo xung điều khiển, kết quả thu được là các dãy xung khác nhau tùy theo mã lệnh, để điều khiển hoạt động của các bộ phận bên trong và bên ngoài CPU.

Trong EU có khối số học và lôgic (arithmetic and logic unit, ALU) chuyên thực hiện các phép tính số học và logic mã toán tử của nó nằm trong các thanh ghi đa năng. Kết quả thường được đặt về thanh ghi AX.

Ngoài ra trong EU còn có các thanh ghi đa năng (registers: AX, BX, CX, DX, SP, BP, SI, DI), thanh ghi cờ FR (flag register) mà công dụng của chúng sẽ được đề cập đến trong phần sau.

Tóm lại, khi CPU hoạt động EU sẽ cung cấp thông tin về địa chỉ cho BIU để khối này đọc lệnh và dữ liệu, còn bản thân nó thì giải mã và thực hiện lệnh.

3.2.3. Các thanh ghi.

Các thanh ghi đa năng (general registers) Có nhiệm vụ ghi tham số cho mã lệnh, đây cũng là nơi lệnh trả kết quả về sau khi được thực hiện. Những thanh ghi đa năng của vi xử lý 16 bit là:

- **AX (accumulator)** rộng 16 bit, được chia làm hai phần: 1 byte cao AH và 1 byte thấp AL. Đây là thanh ghi quan trọng nhất và chuyên được dùng để chứa kết quả các thao tác lệnh. Cả ba cách viết AX, AH, AL đều có thể sử dụng như những thanh ghi riêng biệt.

- **BX (base)** thanh ghi cơ sở, rộng 16 bit, cũng được chia ra làm BH và BL. Đây là thanh ghi thường dùng chứa địa chỉ cơ sở của một bảng dùng trong lệnh XLAT, Cả ba cách viết BX, BH, BL đều có thể sử dụng như những thanh ghi riêng biệt.
- **CX (count)** bộ đếm, rộng 16 bit. Được chia ra làm CH và CL. Thanh ghi CX được dùng để chứa số lần lặp trong trường hợp các lệnh LOOP. Thanh ghi thấp CL được dùng để chứa (nhớ) số lần quay hoặc dịch của các lệnh quay (rotate) và dịch (shift).
- **DX (data)** thanh ghi dữ liệu, rộng 16 bit. Thanh ghi này cùng thanh ghi AX tham gia vào các thao tác của phép nhân hoặc chia các số 16 bit. DX còn dùng để chứa địa chỉ 16 bit của các cổng cứng (dài hơn 8 bit) trong các lệnh truy nhập các cổng ngoại vi (I/O port).

Các thanh ghi đoạn (segment registers) dùng để ghi địa chỉ một đoạn bộ nhớ. Vi mạch 8088/8086 có 20 đường dây trên bus địa chỉ. Do các thanh ghi con trỏ và thanh ghi chỉ số chỉ rộng 16 bit nên không thể định địa chỉ cho toàn bộ nhớ vật lý của máy tính là ($2^{20} = 1.048.576 = 1\text{Mbyte}$). Vì vậy trong chế độ thực (real mode) bộ nhớ được chia làm nhiều đoạn để một thanh ghi con trỏ 16 bit có thể quản lý được. Các thanh ghi đoạn 16 bit sẽ chỉ ra địa chỉ đầu của 4 đoạn trong bộ nhớ, dung lượng lớn nhất của mỗi đoạn nhớ sẽ dài $2^{16} = 64\text{ Kbyte}$ và tại một thời điểm nhất định bộ vi xử lý chỉ làm việc được với 4 đoạn nhớ 64Kbyte này. Việc thay đổi giá trị của các thanh ghi đoạn làm cho các đoạn có thể dịch chuyển linh hoạt trong không gian 1 Mbyte, vì vậy các đoạn có thể nằm cách nhau khi thông tin cần lưu trong chúng đòi hỏi dung lượng đủ 64 Kbyte hoặc cũng có thể nằm trùm nhau do có những đoạn không dùng hết độ dài 64 Kbyte và vì thế các đoạn khác có thể bắt đầu nối tiếp ngay sau đó. Địa chỉ của ô nhớ nằm ở đầu đoạn được ghi trong một thanh ghi đoạn 16 bit, địa chỉ này gọi là *địa chỉ cơ sở*. Mười sáu bit này tương ứng với các đường dây địa chỉ từ A4 đến A20. Như vậy giá trị vật lý của địa chỉ đoạn là giá trị trong thanh ghi đoạn dịch sang trái 4 vị trí. Điều này tương đương với phép nhân với $2^4 = 16$. Địa chỉ của các ô nhớ khác nằm trong đoạn tính được bằng cách cộng thêm vào địa chỉ cơ sở một giá trị gọi là địa chỉ lệch hay độ lệch (offset), gọi như thế vì nó ứng với khoảng lệch của tọa độ một ô nhớ cụ thể nào đó so với ô đầu đoạn. Độ lệch này được xác định bởi các thanh ghi 16 bit khác đóng vai trò thanh ghi lệch (offset register). Nguyên tắc này dẫn đến công thức tính địa chỉ vật lý (physical address) từ địa chỉ đoạn (segment) trong thanh ghi đoạn và địa chỉ lệch (offset) trong thanh ghi con trỏ như sau:

Địa chỉ vật lý = Thanh ghi đoạn x 16 + Thanh ghi lệch
--

Việc dùng hai thanh ghi để nhớ thông tin về địa chỉ thực chất tạo ra một loại địa chỉ gọi là địa chỉ logic và được ký hiệu như sau:

Thanh ghi đoạn : Thanh ghi lệch hay segment:offset.

Địa chỉ kiểu **segment : offset** là logic vì nó tồn tại dưới dạng giá trị của các thanh ghi cụ thể bên trong CPU và khi cần thiết truy nhập ô nhớ nào đó thì nó phải

đổi ra địa chỉ vật lý để rồi đưa lên bus địa chỉ. Việc chuyển đổi này do một bộ tạo địa chỉ thực hiện (phần tử Σ trên hình 3-1).

Vì xử lý 16 bit có 4 thanh ghi đoạn như sau:

- **CS (code segment)** là thanh ghi đoạn mã 16 bit. thanh ghi này phối hợp với con trỏ lệnh IP để ghi địa chỉ mã lệnh trong bộ nhớ. Địa chỉ đầy đủ là CS:IP.
- **DS (data segment)** là thanh ghi đoạn 16 bit cho một đoạn dữ liệu. Thanh ghi này phối hợp với hai thanh ghi chỉ số SI và DI để đánh địa chỉ cho dữ liệu. Địa chỉ đầy đủ cho dữ liệu cần đọc vào là DS:SI, cho dữ liệu cần ghi ra là DS:DI.
- **SS (stack segment)** là thanh ghi đoạn 16 bit cho một ngăn xếp. Địa chỉ đỉnh của ngăn xếp được biểu diễn cùng với con trỏ ngăn xếp SP là SS:SP.
- **ES (extra segment)** là thanh ghi dữ liệu phụ có chiều dài 16 bit. Thường được dùng để đánh địa chỉ một chuỗi. ES:DI là địa chỉ chuỗi cần viết đến (chuỗi đích) và DS:SI là địa chỉ chuỗi đọc vào (chuỗi nguồn).

Các thanh ghi con trỏ và chỉ số có thể được dùng như một thanh ghi đa năng 16 bit. Vì mạch 8088 có tất cả ba thanh ghi con trỏ là (IP, BP, SP) và hai thanh ghi chỉ số (SI, DI). Nhiệm vụ của từng thanh ghi như sau:

- **IP (instruction pointer)** là con trỏ chỉ tới lệnh máy tiếp theo. Lệnh này nằm trong bộ nhớ mà địa chỉ đoạn được ghi trong CS. Như vậy địa chỉ của mã k=lệnh này là CS:IP.
- **BP (base pointer)** là con trỏ cơ sở trỏ về dữ liệu bộ nhớ mà địa chỉ đoạn được ghi trong SS. Địa chỉ đầy đủ sẽ là SS:BP.
- **SP (stack pointer)** là con trỏ ngăn xếp luôn trỏ vào đỉnh ngăn xếp mà địa chỉ đoạn được ghi trong SS. Địa chỉ đầy đủ của dữ liệu là DS:SP.
- **SI (source index)** là chỉ số nguồn, trỏ vào dữ liệu mà địa chỉ đoạn được ghi trong DS. Địa chỉ đầy đủ của dữ liệu là DS:SI.
- **DI (destination index)** là chỉ số đích, cũng trỏ vào đoạn dữ liệu mà địa chỉ đoạn ghi trong DS. Địa chỉ đầy đủ của đoạn dữ liệu là DS:SI.

Thanh ghi cờ FR (flag register) đây là thanh ghi khá đặc biệt trong CPU, dùng để ghi trạng thái kết quả các phép xử lý trong đơn vị số học và logic ALU hoặc một trạng thái hoạt động của EU. Dựa vào các cờ này người lập trình có thể có các lệnh thích hợp tiếp theo cho bộ vi xử lý (các lệnh nhảy có điều kiện). Thanh ghi này là một thanh ghi 16 bit trong 8088/8086. Nhưng chỉ có 9 bit trong thanh ghi được định nghĩa và sử dụng, đó là:

x	x	x	x	O	D	I	T	S	Z	x	A	x	P	x	C
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

x: bit không được định nghĩa.

Hình 3-2. Sơ đồ thanh ghi cờ của bộ vi xử lý 8086/8088.

- **Bit 0: CF** (carry flag) cờ nhớ, CF=1 khi có nhớ hoặc mượn từ MSB.

- **Bit 2: PF** (parity flag) cờ parity, PF phản ánh tính chẵn (parity) của tổng số bit 1 có trong kết quả. Cờ PF =1 khi tổng số bit 1 trong kết quả là chẵn (even parity, parity chẵn).
- **Bit 4: AF** (auxiliary carry flag) cờ nhớ phụ dùng cho các phép tính với mã BCD. AF = 1 khi có nhớ hoặc mượn từ một số BCD thấp (4 bit thấp) sang một số BCD cao (4 bit cao).
- **Bit 6: ZF** (zero flag) cờ rỗng, ZF = 1 khi kết quả bằng 0.
- **Bit 7: SF** (sing flag) cờ dấu, SF = 1 khi kết quả âm.
- **Bit 8: TF** (trap flag) cờ bẫy, TF = 1 khi vi xử lý ở trong chế độ chạy từng lệnh (chế độ này dùng khi cần tìm lỗi trong một chương trình).
- **Bit 9: IF** (interrupt enable flag) cờ cho phép ngắt, IF = 1 cho phép các yêu cầu ngắt che được (maskable interrupt) được tác động.
- **Bit A: DF** (direction flag) cờ hướng. DF = 1 khi CPU làm việc với chuỗi ký tự theo thứ tự từ phải sang trái (lùi).
- **Bit B: OF** (overflow) cờ tràn, OF =1 khi kết quả vượt ra ngoài giới hạn, xảy ra đối với phép tính có dấu.

3.3. Chức năng và thông số của BUS

Một trong những hoạt động và chức năng cơ bản của máy tính là truyền số liệu (data transfer). Sự hoạt động của máy tính do các bộ vi xử lý điều khiển. Bộ vi xử lý và các chip hỗ trợ khác đến lượt mình cũng thường xuyên phải truyền số liệu giữa các khối, bộ phận trong và ngoài chúng với nhau.

Vì có rất nhiều các bộ phận, khối riêng rẽ trong bản thân các Chip và các đường truyền số liệu rất đa dạng, nên một cách hợp lý ta không thể thực hiện các đường nối giữa các bộ phận, khối từng đôi một với nhau mà ta nối chung tất cả các lối vào/ lối ra của các khối riêng rẽ với nhau lên một hệ thống các đường dẫn chung; hệ thống này được gọi là bus

Các bộ phận, khối được nối lên bus phải thoả mãn một yêu cầu là có khả năng được cắt ra hoặc nối trở lại theo lệnh của điều khiển. Lúc một output được cắt ra khỏi bus, nó ở trạng thái trở kháng cao (High impedance, Hi-Z).

Quy tắc nghiêm ngặt của truyền số liệu là trong mỗi thời điểm, tối đa chỉ có một output được cấp số liệu lên bus.

Do trong mỗi thời điểm một output thường cần phải đồng thời cấp số liệu cho nhiều input, cho nên nó cần phải có khả năng phát ra (source) ở mức logic cao hoặc nuốt vào (sink) ở mức logic thấp, một dòng điện lớn tới vài chục mA cấp cho các input đó, đóng vai trò tải của output.

Thông số đặc trưng cho đường bus là trở kháng vào của nó (gồm có điện trở thuần và dung kháng). Thường điện trở thuần khoảng vài $K\Omega$ là thoả mãn yêu cầu của output, chỉ có dung kháng của bus gây khó khăn cho các thiết bị output, (vì nó cản trở tăng tốc độ biến thiên của các mức điện áp trên bus), do đó dung kháng được xem là thông số đặc trưng của bus.

Ví dụ xét trường hợp một bus có điện dung vào 100 pF. Nếu muốn tốc độ biến thiên điện áp trên bus là $du/dt = 2V/10ns$ thì thiết bị output phải nuốt được dòng điện điện dung là

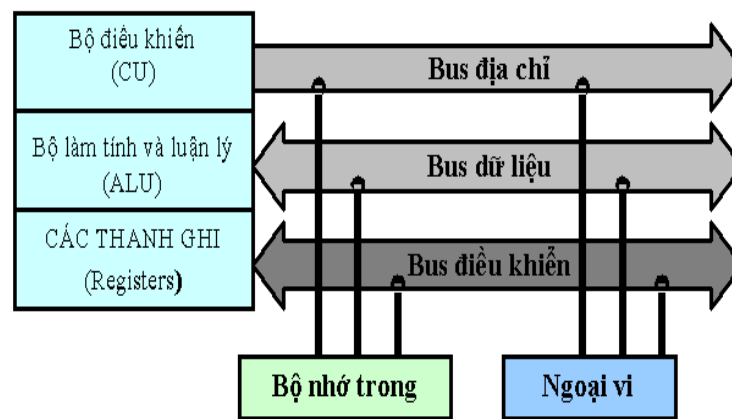
$$i = dq/dt = C(du/dt) = 20 \text{ mA}.$$

Căn cứ theo cấu hình của các thiết bị nối vào bus, người ta phân chúng thành 3 nhóm như sau:

- Output cấp số liệu cho bus.
- Input nhận số liệu từ bus.
- In/ Out khi là input, khi là output.

3.3.1. BUS trong máy vi tính.

Bộ xử lý trung tâm (CPU)



Cấu trúc của một hệ máy tính đơn giản

Hình 3-2: Cấu trúc của một hệ máy tính đơn giản

a. Bus trong vi xử lý và bus bộ xử lý

Trong các bộ vi xử lý có một hệ thống các bus dùng để truyền số liệu, lệnh, các tín hiệu điều khiển ,... , giữa các khối bên trong của nó. Ngoài ra có một hệ thống các bus đưa ra ngoài qua các chân của nó. Các đường bus trong được điều khiển bởi khối điều khiển tùy thuộc hoặc vào nội dung lệnh được giải mã hoặc theo các điều khiển ngắt của bên ngoài đưa vào vi xử lý. Các đường bus này hoạt động theo nhịp của một clock bên trong vi xử lý.

Các bus trong vi xử lý truyền số liệu giữa các khối với nhau, có hai loại đường truyền, một chiều và hai chiều. Hệ các đường bus nối với các bộ phận, khối bên ngoài vi xử lý gồm 20 đường địa chỉ (AD0 - AD 19), 8 đường số liệu (), và các đường thuộc bus điều khiển.

Chính khối điều khiển phát các tín hiệu điều khiển các bus.

Bus bộ vi xử lý là đường truyền dẫn giữa CPU và các chip hỗ trợ trung gian. Những chip hỗ trợ này được gọi là bộ chip (chip set). Bus này dùng để truyền dữ liệu giữa CPU và bus hệ thống chính hoặc giữa CPU và cache ngoài.

Vì mục đích của bus bộ xử lý để gửi hoặc nhận thông tin từ CPU với tốc độ nhanh nhất có thể, nên bus này hoạt động nhanh hơn nhiều so với bất kỳ bus nào khác trong hệ thống và đảm bảo tránh hiện tượng tắc nghẽn ở đây. Bus bộ xử lý bao gồm bus dữ liệu, bus địa chỉ và bus điều khiển. Trong một hệ thống thiết kế cho VXL Pentium, bus bộ xử lý có 64 đường dữ liệu, 32 đường địa chỉ. Pentium Pro và Pentium II có 36 đường địa chỉ.

Bus bộ xử lý hoạt động ở tốc độ đồng hồ cơ sở giống như CPU chạy ngoại trú. Ví dụ Pentium II 333MHz chạy ở tốc độ đồng hồ 333MHz nội trú nhưng chỉ ở 66,6 MHz ngoại trú.

Tốc độ truyền của bus bộ xử lý được xác định bằng cách nhân độ rộng dữ liệu với tốc độ đồng hồ cơ sở rồi chia cho 8.

Khi thiết kế các bộ vi xử lý, có thể tùy ý lựa chọn loại bus bên trong vi xử lý, còn với các bus liên hệ với bên ngoài cần phải xác định rõ các quy tắc làm việc cũng như các đặc điểm kỹ thuật về điện và cơ khí để người thiết kế Main Board có thể ghép nối vi xử lý với các thiết bị khác. Nói cách khác, các bus này phải tuân theo một chuẩn nhất định. Tập các quy tắc của chuẩn còn được gọi là nghi thức bus (bus protocol).

Trong thế giới máy tính có rất nhiều loại bus khác nhau được sử dụng, các bus này nói chung là không tương thích với nhau. Sau đây là một số loại bus được dùng phổ biến:

Tên bus	Lĩnh vực áp dụng
- Camac	Vật lý hạt nhân
- EISA	Một số hệ thống dùng bộ VXL 8036
- IBM PC, PC/AT	Máy tính IBM PC, IBM/PC/AT
- Massbus	Máy PDP - 1 và VAX
- Microchannel	Máy PS/2
- Multibus I	Một số hệ thống có VXL 8088, 8086
- Multibus II	Một số hệ thống có VXL 80386
- Versabus	Một số hệ thống dùng VXL Motorola
- VME	Một số hệ thống dùng VXL 68x0 của Motorola.

Người ta thường phân loại bus theo ba cách sau:

1. Theo tổ chức phân cứng (như các loại bus nêu trên)
2. Theo nghi thức truyền thông (bus đồng bộ và không đồng bộ).
3. Theo loại tín hiệu truyền trên bus (bus địa chỉ, bus dữ liệu ...)

Sự làm việc của các bus

Thường có nhiều thiết bị nối với bus, một số là thiết bị tích cực và có thể đòi hỏi truyền thông tin trên bus, trong khi đó lại có các thiết bị thụ động chờ các yêu cầu từ các thiết bị khác. Các thiết bị tích cực được gọi là chủ bus (master), còn các thiết bị thụ động là tớ (slave).

Khi CPU ra lệnh cho bộ điều khiển đĩa đọc/ ghi một khối dữ liệu thì CPU là master còn bộ điều khiển đĩa là slave. Tuy nhiên khi bộ điều khiển đĩa ra lệnh cho bộ nhớ nhận dữ liệu mà nó đọc từ đĩa thì nó lại giữ vai trò của master.

Bus Driver và Bus Receiver.

Tín hiệu điện mà các thiết bị trong máy tính phát ra thường không đủ mạnh để điều khiển được bus, nhất là khi bus khá dài và có nhiều thiết bị nối với nó. Chính vì vậy mà hầu hết các bus master được nối với bus thông qua một chip được gọi là bus driver, về căn bản đó là bộ khuếch đại tín hiệu số. Tương tự như vậy, hầu hết các slave bus được nối với bus thông qua bus receiver. Đối với các thiết bị có thể khi thì đóng vai trò master, khi thì đóng vai trò slave, người ta sử dụng một chip kết hợp, gọi là transceiver. Các chip này đóng vai trò ghép nối và thường là các thiết bị 3 trạng thái, cho phép có thể ở trạng thái thứ ba: hở mạch (còn gọi là thả nổi).

Giống như MPU, bus có các đường địa chỉ, đường số liệu và đường điều khiển. Tuy nhiên không nhất thiết phải có ánh xạ một - một giữa các tín hiệu ở các chân ra của MPU và các đường dây của bus.

Những vấn đề quan trọng nhất liên quan đến thiết kế bus là: Nhịp đồng hồ bus (sự phân chia thời gian, hay còn gọi là bus clocking), cơ chế trọng tài bus (bus arbitration), xử lý ngắt và xử lý lỗi.

Các bus có thể được chia theo nghi thức truyền thông tin thành hai loại riêng biệt là bus đồng bộ và bus không đồng bộ phụ thuộc vào việc sử dụng nhịp đồng hồ bus.

b. Bus đồng bộ (Synchronous bus)

Bus đồng bộ có một đường dây điều khiển bởi một bộ dao động thạch anh, tín hiệu trên đường dây này có dạng sóng vuông, với tần số thường nằm trong khoảng 5MHz - 50 MHz. Mọi hoạt động bus xảy ra trong một số nguyên lần chu kỳ này và được gọi là chu kỳ bus.

Giãn đồ thời gian của một bus đồng bộ với tần số đồng hồ là 4MHz, như vậy chu kỳ bus là 250ns.

- **T1** bắt đầu bằng sườn lên của tín hiệu đồng hồ Φ , trong một phần thời gian của T1, MPU đặt địa chỉ của byte cần đọc lên bus địa chỉ. Sau khi tín hiệu địa chỉ được thiết lập giá trị mới, MPU đặt các tín hiệu \overline{MREQ} và \overline{RD} tích cực. Tín hiệu \overline{MREQ} (memory request, truy cập bộ nhớ) chứ không phải thiết bị I/O; còn tín hiệu \overline{RD} (Read) chọn Read.

- T2 là thời gian cần thiết để bộ nhớ giải mã địa chỉ và đưa dữ liệu lên bus dữ liệu.

- **T3** tại sườn xung xuống của T3, MPU nhận dữ liệu trên bus dữ liệu, chứa vào thanh ghi bên trong MPU và chốt dữ liệu. Sau đó MPU đảo các tín hiệu \overline{MREQ} và \overline{RD} .

Như vậy đã kết thúc một thao tác đọc, tại chu kỳ máy tiếp theo MPU có thể thực hiện một thao tác khác.

- **T_{AD}** : theo giản đồ thời gian, $T_{AD} \leq 110\text{ns}$, đây là thông số do nhà sản xuất đảm bảo, MPU sẽ đưa ra tín hiệu địa chỉ không chậm hơn 110ns tính từ thời điểm giữa sườn lên của T1.

- **T_{DS}** : Giá trị nhỏ nhất là 50ns, thông số này cho phép dữ liệu được đưa ra ổn định trên bus dữ liệu ít nhất là 50ns trước thời điểm giữa sườn xuống của T3. Yêu cầu về thời gian này đảm bảo cho MPU đọc dữ liệu tin cậy.

Khoảng thời gian bắt buộc đối với T_{AD} và T_{DS} cũng nói lên rằng, trong trường hợp xấu nhất, bộ nhớ chỉ có $250 + 250 + 125 - 110 - 50 = 465\text{ns}$ tính từ thời điểm có tín hiệu địa chỉ cho tới khi nó đưa dữ liệu ra bus địa chỉ. Nếu bộ nhớ không đáp ứng đủ nhanh, nó cần phải phát tín hiệu xin chờ \overline{WAIT} trước sườn xuống của T2. Thao tác này đưa thêm vào một trạng thái chờ (wait state), khi bộ nhớ đã đưa ra dữ liệu ổn định, nó sẽ đảo tín hiệu \overline{WAIT} thành $WAIT$.

- **T_{ML}** : Đảm bảo rằng tín hiệu địa chỉ sẽ được thiết lập trước tín hiệu \overline{MREQ} ít nhất là 60ns. Khoảng thời gian này là quan trọng nếu tín hiệu \overline{MREQ} điều khiển sự tạo ra tín hiệu chọn chip CS, bởi vì một số chip nhớ đòi hỏi phải nhận được tín hiệu địa chỉ trước tín hiệu chọn chip. **Như vậy không thể chọn chip nhớ với thời gian thiết lập là 75ns.**

- **T_M, T_{RL}** : Các giá trị bắt buộc đối với 2 đại lượng này có ý nghĩa là cả hai tín hiệu \overline{MREQ} và \overline{RD} sẽ là tích cực trong khoảng thời gian 85ns tính từ thời điểm xuống của xung đồng hồ T1. Trong trường hợp xấu nhất, chip nhớ chỉ có $250 + 250 - 85 - 50 = 365\text{ns}$ sau khi hai tín hiệu trên là tích cực để đưa dữ liệu ra bus. Sự bắt buộc về thời gian này bổ sung thêm sự bắt buộc thời gian với tín hiệu đồng hồ.

- **T_{MH}, T_{RH}** : Hai đại lượng này cho biết cần có bao nhiêu thời gian để các tín hiệu \overline{MREQ} và \overline{RD} sẽ được đảo sau khi dữ liệu đã được MPU đọc vào.

- **T_{DH}** : Cho biết bộ nhớ cần phải lưu dữ liệu bao lâu trên bus sau khi tín hiệu \overline{RD} đã đảo.

Block Transfer, truyền tải khối dữ liệu.

Ngoài các chu kỳ đọc/ ghi, một số bus đồng bộ còn hỗ trợ truyền dữ liệu theo khối. Khi một thao tác đọc/ ghi bắt đầu, bus master báo cho slave biết có bao nhiêu byte cần truyền đi, sau đó slave sẽ liên tục đưa ra mỗi chu kỳ một byte, cho đến khi đủ số byte được thông báo. Như vậy, khi đọc dữ liệu theo khối, n byte dữ liệu cần n+2 chu kỳ, thay cho 3n chu kỳ như trước.

Cách khác làm cho bus truyền dữ liệu nhanh hơn là làm cho các chu kỳ ngắn lại. Trong ví dụ trên, mỗi byte được truyền đi trong 750ns, vậy bus có dải thông là 1.33MBs. Nếu xung đồng hồ là 8MHz, thời gian một chu kỳ chỉ còn một nửa, dải thông sẽ là 2.67MBs.

Tuy vậy việc giảm chu kỳ bus dẫn đến các khó khăn về mặt kỹ thuật, các bit tín hiệu truyền trên các đường dây khác nhau trong bus không phải luôn có cùng vận tốc, dẫn đến một hiệu ứng, gọi là **bus skew**.

Khi nghiên cứu về bus cần phải quan tâm đến vấn đề tín hiệu tích cực nên là mức thấp hay mức cao. Điều này tùy thuộc vào người thiết kế bus xác định mức nào là thuận lợi hơn.

Bảng 3.1. Giá trị của một số thông số thời gian

Ký hiệu	Tham số	Min	Max
T_{AD}	Thời gian trễ của tín hiệu địa chỉ		110
T_{ML}	Thời gian địa chỉ ổn định trước tín hiệu \overline{MREQ}	60	
T_M	Thời gian trễ của \overline{MREQ} so với sườn xuống của tín hiệu đồng hồ T1		85
T_{RL}	Thời gian trễ của \overline{RD} so với sườn xuống của tín hiệu đồng hồ T1		85
T_{DS}	Thời gian thiết lập dữ liệu trước sườn xuống của tín hiệu đồng hồ T3	50	
T_{MH}	Thời gian trễ của \overline{MREQ} so với sườn xuống của tín hiệu đồng hồ T3		85
T_{RH}	Thời gian trễ của \overline{RD} so với sườn xuống của tín hiệu đồng hồ T3		85
T_{DH}	Thời gian lưu trữ dữ liệu từ lúc đảo tín hiệu \overline{RD}	0	

c. Bus không đồng bộ (asynchronous bus).

Bus không đồng bộ không sử dụng một xung đồng hồ định nhịp. Chu kỳ của nó có thể kéo dài tùy ý và có thể khác nhau đối với các cặp thiết bị trao đổi tin khác nhau.

Làm việc với bus đồng bộ dễ dàng hơn do nó được định thời một cách gián đoạn, tuy vậy chính đặc điểm này cũng dẫn đến nhược điểm. Thứ nhất là: mọi công việc được tiến hành trong những khoảng thời gian là bội số nhịp đồng hồ bus, nếu một thao tác nào đó của CPU hay bộ nhớ có thể hoàn thành trong 3,2 chu kỳ thì nó sẽ phải kéo dài thành 4 chu kỳ. Điều hạn chế lớn nữa là đã chọn chu kỳ bus và đã xây dựng bộ nhớ, I/O Card cho bus này thì khó có thể tận dụng được những tiến bộ của công nghệ. Chẳng hạn sau khi đã xây dựng bus với sự định thời như trên, công nghệ mới đưa ra các chip CPU và chip nhớ có thời gian chu kỳ là 100ns (thay cho 250ns như cũ), chúng vẫn cứ phải chạy với tốc độ thấp như các CPU và chip nhớ loại cũ, bởi vì nghi thức bus đòi hỏi chip nhớ phải đưa ra dữ liệu và ổn định dữ liệu ngay trước thời điểm ứng với sườn xuống của T3. Nếu có nhiều thiết bị khác nhau nối với một bus, trong đó có một số thiết bị có thể hoạt động nhanh hơn các thiết bị khác thì cần phải đặt bus hoạt động phù hợp với thiết bị chậm nhất.

Bus không đồng bộ ra đời nhằm khắc phục các nhược điểm của bus đồng bộ. Hình 3.3 minh họa sự hoạt động của bus không đồng bộ, trong đó master yêu cầu đọc bộ nhớ.

Trước hết master cần phát ra địa chỉ nhớ mà nó muốn truy cập, sau đó phát tín hiệu \overline{MREQ} tích cực để báo rằng nó muốn truy cập bộ nhớ chứ không phải cổng I/O. Tín hiệu này là cần thiết vì bộ nhớ và các cổng I/O đều có thể dùng chung một miền địa chỉ. Tiếp theo master phải phát tín hiệu \overline{RD} tích cực để bên slave biết rằng master sẽ thực hiện thao tác đọc chứ không phải là thao tác ghi.

Các tín hiệu \overline{MREQ} và \overline{RD} được đưa ra sau tín hiệu định địa chỉ bao lâu tùy thuộc vào tốc độ của master. Sau khi hai tín hiệu này đã ổn định, master sẽ phát tín hiệu đặc biệt, là \overline{MSYN} (Master SYNchronization) ở mức tích cực để báo cho slave biết rằng các tín hiệu cần thiết đã sẵn sàng trên bus, slave có thể nhận lấy. Khi slave nhận các tín hiệu này, nó sẽ thực hiện công việc với tốc độ nhanh nhất có thể được (nhanh chóng đưa dữ liệu của ô nhớ yêu cầu lên bus dữ liệu). Khi hoàn thành, slave sẽ phát tín hiệu \overline{SSYN} (Slave SYNchronization) tích cực.

Khi master nhận được tín hiệu \overline{SSYN} tích cực, nó biết rằng dữ liệu của slave đã sẵn sàng và thực hiện việc chốt dữ liệu, sau đó đảo các đường địa chỉ cũng như các tín hiệu \overline{MREQ} và \overline{RD} và \overline{MSYN} .

Khi slave nhận được sự đảo tín hiệu \overline{MSYN} thành không tích cực, nó biết rằng một chu kỳ đã kết thúc và đảo tín hiệu \overline{SSYN} . Bây giờ bus lại trở lại trạng thái ban đầu, mọi tín hiệu đều là không tích cực, tất cả sẵn sàng chờ bus master mới.

Trên giản đồ thời gian của bus không đồng bộ, ta sử dụng mũi tên để thể hiện nguyên nhân và kết quả. Việc đưa \overline{MSYN} lên mức tích cực dẫn đến việc truyền dữ liệu ra bus dữ liệu và đồng thời cũng dẫn đến việc slave phát ra tín hiệu \overline{SSYN} tích cực. Đến lượt mình, tín hiệu \overline{SSYN} lại gây ra sự đảo mức của các đường địa chỉ, \overline{MREQ} và \overline{RD} và \overline{MSYN} . Cuối cùng sự đảo mức của \overline{MSYN} lại gây ra sự đảo mức tín hiệu \overline{SSYN} và kết thúc một chu kỳ đọc.

Full handshake.

Tập các tín hiệu phối hợp với nhau như vậy được gọi là Full handshake, nó chủ yếu gồm có 4 sự kiện sau:

1. \overline{MSYN} được đặt lên mức tích cực.
2. \overline{SSYN} được đặt tích cực để đáp lại tín hiệu \overline{MSYN}
3. \overline{MSYN} được đảo để đáp lại tín hiệu \overline{SSYN}
4. \overline{SSYN} được đảo để đáp lại tín hiệu \overline{MSYN} thành không tích cực.

Ta có thể nhận thấy Full handshake là quan hệ nhân quả, độc lập với thời gian. Nếu một cặp master-slave nào đó hoạt động chậm hoặc thời gian bị kéo dài thì cặp master-slave kế tiếp không hề bị ảnh hưởng.

Tuy ưu điểm của bus không đồng bộ rất rõ ràng, nhưng trong thực tế phần lớn các bus đang được sử dụng là loại bus đồng bộ. Lý do căn bản là các hệ thống sử dụng bus đồng bộ là dễ thiết kế hơn. CPU chỉ cần chuyển các mức tín hiệu cần thiết sang trạng thái tích cực là các chip nhớ đáp ứng ngay, không cần tín hiệu phản hồi. Chỉ cần các chip được chọn phù hợp thì mọi hoạt động đều trôi chảy.

3.3.2. Trọng tài bus (bus arbitration).

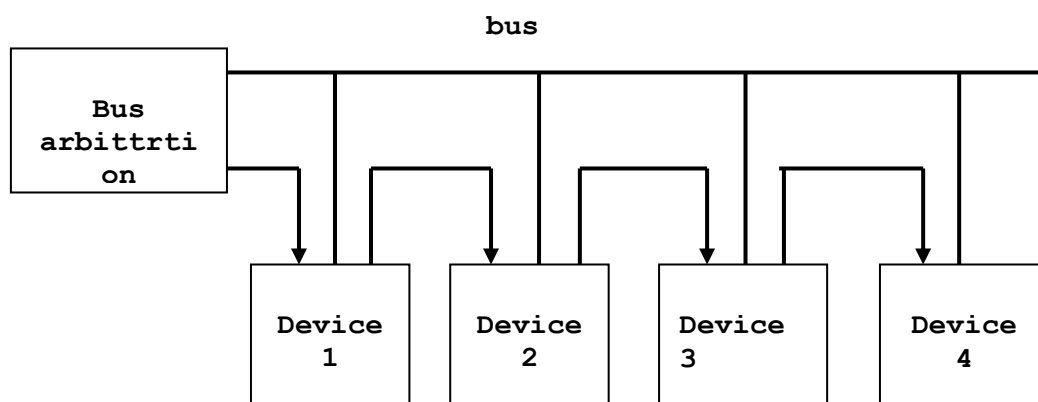
Trong hệ thống máy tính không phải chỉ có CPU làm bus master, thực tế các chip I/O cũng có lúc phải làm chủ bus để có thể đọc hoặc ghi vào bộ nhớ và để gọi ngắt; các bộ đồng xử lý cũng có thể làm chủ bus. Như vậy cần phải giải quyết vấn đề tranh chấp khi có từ hai thiết bị trở lên đồng thời muốn làm chủ bus. Để giải quyết vấn đề này cần có một cơ chế trọng tài để tránh sự xung đột. Cơ chế trọng tài có thể là tập trung hoặc không tập trung.

a. Trọng tài bus tập trung

Hình 3.4 là một ví dụ đơn giản về trọng tài bus tập trung. ở đây, một trọng tài bus duy nhất sẽ quyết định thiết bị nào được là chủ bus tiếp theo. Nhiều bộ VXL có đơn vị trọng tài bus được thiết kế ngay trong chip VXL, trong một số máy tính mini, đơn vị trọng tài bus nằm ngoài CPU.

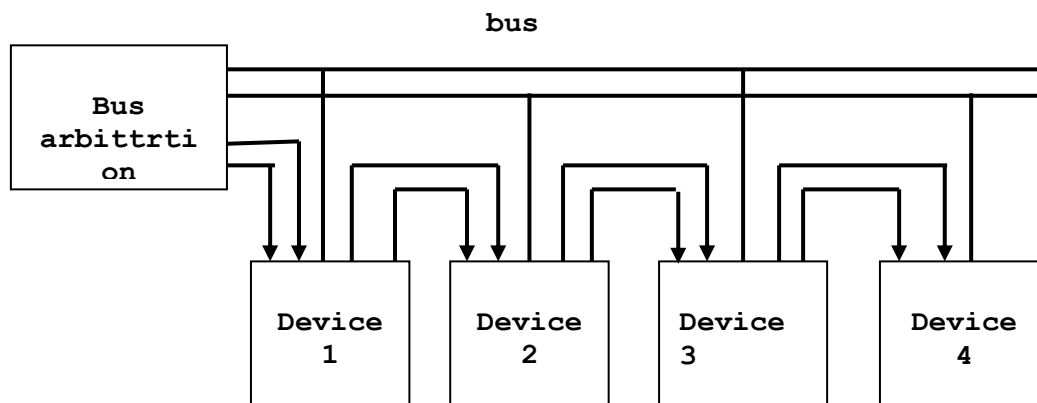
Theo cơ chế này, trọng tài chỉ có thể biết là có yêu cầu chiếm dụng bus hay không, chứ không biết có bao nhiêu đơn vị muốn chiếm bus. Khi trọng tài bus nhận được một yêu cầu, nó sẽ phát ra một tín hiệu cho phép trên đường dây bus grant (cho dùng bus). Đường dây này nối qua tất cả các thiết bị vào/ ra theo kiểu nối tiếp.

Khi thiết bị nằm gần trọng tài nhất nhận được tín hiệu cho phép, nó sẽ kiểm tra xem có phải chính nó đã phát yêu cầu chiếm bus không? Nếu đúng thì nó sẽ chiếm lấy bus và không truyền tiếp tín hiệu cho phép trên đường dây. Nếu nó kiểm tra thấy không phải là yêu cầu của mình thì tiếp tục truyền tín hiệu cho phép tới thiết bị kế tiếp trên đường dây.



Hình 3.4. Trọng tài bus tập trung có một mức, mắc nối tiếp.

Một số loại bus có nhiều mức độ ưu tiên, với mỗi mức ưu tiên có một đường dây yêu cầu bus và một đường dây cho chiếm bus. Hình 5.5 là một ví dụ về bus có hai mức (các bus trong thực tế thường có 4, 8 hay 16 mức). Mỗi thiết bị trong hệ thống máy tính nối với một trong các mức yêu cầu bus, các thiết bị thường được sử dụng hơn được gắn với đường dây có mức ưu tiên cao hơn.



Hình 3.5. Trọng tài bus tập trung có hai mức, mắc nối tiếp.

Nếu có một số thiết bị ở các mức ưu tiên khác nhau cùng yêu cầu, trọng tài bus sẽ chỉ phát tín hiệu cho phép đối với yêu cầu có mức ưu tiên cao nhất. Trong số các thiết bị có cùng mức ưu tiên, thiết bị gần trọng tài bus hơn sẽ có quyền ưu tiên cao hơn.

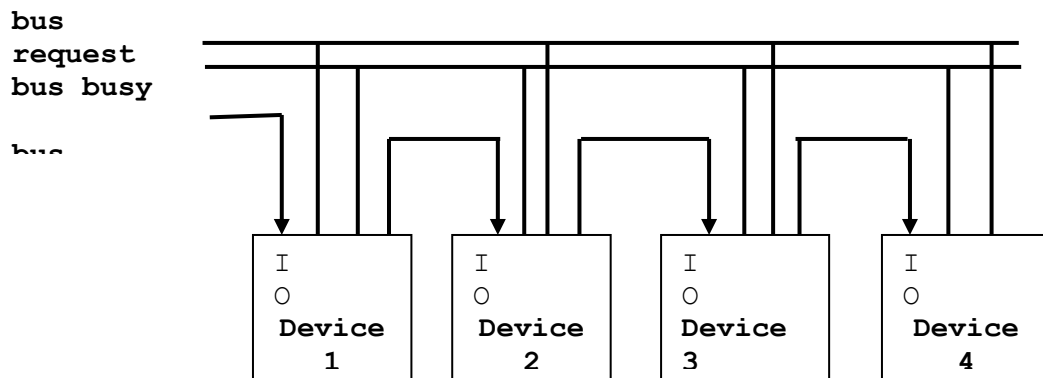
Một số trọng tài bus có đường dây thứ ba nối tới các thiết bị để các thiết bị xác nhận việc nhận được tín hiệu cho phép và chiếm dụng bus, gọi là đường dây biên nhận acknowledgement (ACK). Ngay sau khi một thiết bị phát tín hiệu tích cực trên đường dây ACK, trọng tài bus có thể đảo tín hiệu trên các đường dây trên các đường dây yêu cầu bus và cho phép dùng bus thành mức không tích cực. Kết quả là các thiết bị khác có thể đòi hỏi chiếm dụng bus trong khi thiết bị đầu tiên đang dùng bus. Khi kết thúc phiên làm việc hiện thời, bus master kế tiếp sẽ được lựa chọn. Cách làm việc như vậy làm tăng hiệu quả sử dụng bus, nhưng cần thêm một đường truyền tín hiệu ACK và cấu trúc của các thiết bị cũng phức tạp hơn. Các chip của Motorola sử dụng các bus loại này.

b. Trọng tài bus không tập trung

Trong cơ chế trọng tài bus không tập trung, không cần sử dụng một đơn vị riêng làm trọng tài bus, nhờ vậy giảm được giá thành phần cứng. Trong một số loại máy tính khác nhau, người ta đã sử dụng một vài kiểu trọng tài bus theo cơ chế này.

Trọng tài bus không tập trung trong multibus

Trong Multibus, người ta cho phép có thể lựa chọn cơ chế trọng tài bus không tập trung hoặc không tập trung, cơ chế trọng tài bus không tập trung được thực hiện theo sơ đồ trên hình 5.6



Hình 3.6. Trọng tài bus không tập trung trong Multibus.

Người ta chỉ sử dụng 3 đường dây, không phụ thuộc vào số lượng thiết bị nối với bus. Bao gồm:

- + Yêu cầu chiếm dụng bus (bus request)
- + Trạng thái bus (bus busy), được bus master đặt ở mức tích cực
- + Trọng tài bus, được mắc nối tiếp qua các thiết bị

Khi không có thiết bị nào yêu cầu chiếm bus, đường dây trọng tài bus truyền mức tích cực tới tất cả các thiết bị. Khi một đơn vị nào đó muốn chiếm dụng bus, đầu tiên nó kiểm tra bus có rồi không và kiểm tra đầu vào của đường trọng tài bus, nếu thấy có điện áp $IN = 5V$ thì nó có thể xin bus bằng cách đưa tín hiệu yêu cầu bus (Request) và xóa tín hiệu OUT, tức là đặt $OUT = 0V$. Do đó các thiết bị ưu tiên thấp hơn sẽ không xin được bus. Lúc này nó trở thành bus master.

3.4. Xử lý ngắt

Một chức năng quan trọng của bus là xử lý ngắt. Khi CPU ra lệnh cho một thiết bị trong máy tính thực hiện việc đọc, ghi hay xử lý tin, nó thường chờ đợi tín hiệu ngắt do thiết bị I/O phát ra khi hoàn thành công việc được CPU yêu cầu. Khi nhận được tín hiệu ngắt, CPU đáp ứng ngay, đó có thể là việc nhận dữ liệu do thiết bị I/O chuyển về, cũng có thể là việc tiếp tục gửi dữ liệu tới thiết bị I/O hoặc CPU sử dụng bus cho một thao tác khác Như vậy chính ngắt phát ra tín hiệu yêu cầu bus.

Vì có thể có nhiều thiết bị ngoại vi cùng phát tín hiệu ngắt, cho nên cũng cần có một cơ chế trọng tài giống như đối với các bus thông thường. Giải pháp thường dùng là gán các mức độ ưu tiên cho các thiết bị và sử dụng một trọng tài tập trung

để trao quyền ưu tiên cho các thiết bị và sử dụng một trong tài tập trung để trao quyền ưu tiên cho các thiết bị quan trọng thường xuyên được sử dụng.

3.5. Một số bus thông dụng

3.5.1. Bus IBM PC

Đây là ví dụ điển hình về một loại bus được sử dụng trong các hệ thống thương mại, nó được sử dụng rộng rãi trong các hệ thống vi xử lý dựa trên chip 8088. Hầu hết họ PC, kể cả các máy tương thích đều sử dụng bus này. Chính bus IBM PC tạo nên cơ sở cho bus IBM PC/AT và nhiều loại bus khác. Bus này có 62 đường dây, trong đó có 20 đường địa chỉ, 8 đường số liệu và các đường tín hiệu khác. Được liệt kê trong bảng 3.2

Về mặt vật lý, bus IBM PC được thiết kế ngay trên bo mạch chính và có khoảng gần chục đầu nối dạng khe cắm (slot) để cắm các card mở rộng, trên mỗi khe cắm có 62 chân được chia thành hai hàng.

Tín hiệu	Số dây	In	Out	giải thích
OSC	1		x	Chân dao động (14,31818 MHz)
CLK	1		x	Xung đồng hồ (4,77 MHz)
RESET	1		x	Tín hiệu reset CPU và các thiết bị I/O
A0 - A9	20		x	Các đường dây địa chỉ
D0 - D7	8		x	Các đường truyền dữ liệu
ALE	1		x	Chốt địa chỉ
(MEMR)	1		x	Đọc bộ nhớ
(MEMW)	1		x	Ghi vào bộ nhớ
(IOR)	1		x	Đọc cổng I/O
(IOW)	1		x	Ghi ra cổng I/O
AEN	1	x		Address ENable, yêu cầu bus địa chỉ
(IOCHCHK)	1	x		I/O Chanel Check
IOCHRDY	1	x		I/O Chanel Ready
IRQ2 - IRQ7	6	x		Các đường yêu cầu ngắt
DRQ1 - DRQ3	3	x		DMA Request
DACK0 - DACK3	4		x	DMA Acknowledge
T/C	1		x	Terminal/Count
Power	5			
GND	3			
Reserved	1			

Tín hiệu đồng hồ OSC và CLK

Các máy IBM PC đầu tiên sử dụng các phần tử dao động thạch anh, ở tần số 14,31818 MHz, tần số này được chọn phải thoả mãn việc tạo ra tín hiệu đồng bộ màu hệ NTSC. Tần số này cao hơn so với chuẩn 8088 (tần số cực đại là 5 MHz), do đó nó được chia ba thành 4,77MHz. Tần số 4,77 MHz được dùng làm đồng hồ chính để xác định chu kỳ bus. Tín hiệu tần số 4,77 MHz cũng có trên bus IBM PC và ký hiệu là CLK. Tín hiệu này không cân xứng như tín hiệu đồng hồ 14,31818 MHz mà trong một chu kỳ bao gồm 2/3 thời gian ở mức thấp và 1/3 thời gian ở mức cao.

Tín hiệu RESET

Tín hiệu RESET trên bus do chip 8284A tạo ra. Để RESET CPU, các mạch điện bên ngoài gửi tín hiệu tới 8284A, nó sẽ đặt tín hiệu reset lên mức tchs cực, buộc CPU và các thiết bị I/O khởi tạo lại.

Các đường địa chỉ và dữ liệu

CPU không nối trực tiếp với các đường địa chỉ và đường số liệu của bus, mà thông qua các chip khác. Các đường địa chỉ được chốt bằng cách dùng 3 chip 74LS373, mỗi chip là một bộ 8 thanh ghi chốt, tuy nhiên chỉ sử dụng 20 trong số 24 đường có thể.

Các đường dữ liệu sẽ được lấy mẫu (đọc nhanh giá trị) hoặc cung cấp giá trị trong những thời gian xác định, như trong sườn dương của một tín hiệu đồng hồ nào đó, vì vậy không cần chốt. Các đường dữ liệu của bus được điều khiển bởi bus transceiver (chip 74LS245). Chân DIR xác định hướng của tín hiệu là đi đến hay đi ra từ CPU.

Lý do chính của việc nối các chân của MPU với bên ngoài thông qua các bộ đệm chính là vì nó được chế tạo theo công nghệ MOS, CPU không có khả năng cung cấp đủ dòng để điều khiển tất cả các phần tử nối với bus. Các chip làm bộ đệm dùng công nghệ TTL có khả năng cung cấp đủ dòng cho cả các thiết bị nối với bus.

Ngoài ra còn lý do khác là, khi có một thiết bị nào đó khác CPU muốn trở thành bus master (như DMAC), CPU cần phải thả nổi các bus. Phương pháp đơn giản nhất được áp dụng là thiết bị đó phải phát tín hiệu AEN (Address ENable) để đảo tín hiệu cho phép đưa ra trên các thanh ghi chốt và transceiver, làm cho các bus được thả nổi.

Tín hiệu ALE (Address Latch Enable)

Tín hiệu ALE được đặt mức tích cực khi CPU đang điều khiển các đường tín hiệu địa chỉ, cho phép các chip 74LS373 biết khi nào cần chốt địa chỉ lại. Tín hiệu này củ bus cũng cho bộ nhớ và các chip I/O biết khi nào các tín hiệu trên bus địa chỉ là hợp lệ.

Các đường tín hiệu $\overline{\text{MEMR}}$, $\overline{\text{MEMW}}$, $\overline{\text{IOR}}$, $\overline{\text{IOW}}$

Để điều khiển việc đọc/ ghi vào bộ nhớ hoặc các thiết bị vào/ra. Nhờ các tín hiệu này, bus cung cấp hai thông gian địa chỉ riêng biệt, một cho MEM và một cho I/O. Bộ nhớ sẽ không phản ứng khi thấy tín hiệu \overline{IOR} , \overline{IOW} ở mức tích cực.

CPU sử dụng các tín hiệu $\overline{S0}$ - $\overline{S2}$ đưa vào chip điều khiển bus 8288 để tạo ra các tín hiệu \overline{MEMR} , \overline{MEMW} , \overline{IOR} , \overline{IOW} cùng với tín hiệu ALE. Chip điều khiển bus cũng nhận tín hiệu điều khiển ANE từ bus, tín hiệu này do một thiết bị muốn trở thành bus master đưa ra, khi nhận được tín hiệu ANE, chip điều khiển bus sẽ phát tín hiệu điều khiển các chip chốt địa chỉ và chip bus transceiver thả nổi bus.

Tín hiệu $\overline{IOCHCHK}$ (I/O CHanel CHeck)

Tín hiệu này sẽ tích cực khi có lỗi chẵn /lẻ bị phát hiện trên bus. Tín hiệu này sẽ tác động mmoet ngắt NMI.

Tín hiệu $\overline{IOCHRDY}$ (I/O CHanel ReaDY)

Tín hiệu này do bộ nhớ đưa ra khi tốc độ hoạt động của nó thấp, yêu cầu CPU cho thêm một số chu kỳ để đợi, bằng cách chèn wait states vào các chu kỳ đọc/ghi bộ nhớ.

Các tín hiệu IRQ2-IRQ7.

Là các tín hiệu do các thiết bị ngoại vi đưa ra, đưa đến chip điều khiển ngắt 8259A. Khi có tín hiệu gửi đến chip điều khiển ngắt, nó sẽ kiểm soát các tín hiệu này và đưa ra một tín hiệu yêu cầu ngắt tới CPU và đặt số hiệu vector ngắt lên đường dữ liệu khi CPU cần đến. IRQ0 thường được mạch đồng hồ và IRQ1 được bàn phím sử dụng.

Các tín hiệu liên quan đến DMA

Các tín hiệu còn lại nói chung liên quan đến hoạt động DMA, chẳng hạn khi CPU yêu cầu ổ đĩa đọc một khối dữ liệu, mạch điều khiển ổ đĩa sẽ chờ nhận được byte đầu tiên từ ổ đĩa đưa ra, sau đó phát ra một yêu cầu trở thành bus master để ghi byte đó vào bộ nhớ.

Chip 8237A được INTEL thiết kế nhằm quản lý các nghi thức bus và thực hiện DMA trong đó có việc taung địa chỉ bộ nhớ và giảm con đếm sau khi truyền mỗi byte. Việc này nó thực hiện thay cho các thiết bị I/O, giúp giảm giá thành của chúng.

Về căn bản, chip 8237A là một CPU nhỏ, có các chương trình được ghi sẵn bên trong. Khi 8088 muốn bắt đầu hoạt động DMA đối với một thiết bị ngoại vi nào đó, nó nạp số hiệu vào thiết bị, địa chỉ ô nhớ, số byte, hướng truyền và các thông tin khác vào các thanh ghi bên trong 8237A. Khi chip điều khiển đã sẵn sàng đọc hoặc ghi byte đầu tiên, nó đặt mức tích cực lên một trong các đường DRQ của bus để đưa vào chip 8237A. Khi nhận được tín hiệu, 8237A đòi chiếm bus và sẵn sàng truyền một byte. Chip 8237A phát tín hiệu \overline{DACK} tới chip điều khiển báo cho nó biết hãy ghi hoặc đọc byte của mình (trong thao tác đọc hoặc ghi tương ứng).

Trong khoảng một chu kỳ này, chip 8237A điều khiển hoạt động của bus như một bus master.

Chip 8237A có 4 kênh độc lập và có thể quản lý đồng thời 4 đường truyền.

Tín hiệu T/C (Terminal/Count)

Đường T/C được chip 8237A đặt mức tích cực khi con đếm byte (byte count) bằng 0, báo cho bộ điều khiển I/O biết rằng công việc yêu cầu đã hoàn tất, đã đến lúc báo hiệu cho 8258A gọi ngắt tới CPU.

3.5.2. Bus IBM PC/AT

Bus IBM PC/AT là bước phát triển tiếp theo của thế hệ bus IBM PC nhằm phát huy được những khả năng hơn hẳn của bộ VXL 80286 so với 8088 trước nó. Với bus địa chỉ 24 dây, có khả năng đánh địa chỉ cho $2^{24} = 16\text{MB}$ bộ nhớ và có bus dữ liệu 16 bit.

Với giải pháp mở rộng PC bus, bổ sung thêm vào các khe cắm cũ một đoạn khe cắm ngắn, trên đó có 36 dây tín hiệu, tăng thêm cho bus địa chỉ 4 dây, bus dữ liệu 8 dây, các đường yêu cầu ngắt, kênh DMA, Nhờ vậy các card mở rộng trước đây vẫn dùng cho IBM PC có thể dùng cho IBM PC/AT.

Ngoài việc mở rộng bus, tần số tín hiệu đồng hồ bus cũng được tăng từ 4,77 MHz ở PC bus thành 8MHz, nhờ đó tốc độ truyền thông trên bus cũng tăng lên nhiều.

Năm 1991 tổ chức IEEE (Institute of Electrical and Electronic Engineers) đã đưa ra tiêu chuẩn quốc tế cho bus của máy AT, gọi là bus ISA (Industrial Standard Architecture)

Các nhà sản xuất PC khác đã đưa ra một chuẩn khác, đó là bus EISA (Extended ISA), về căn bản bus này là sự mở rộng bus PC/AT thành 32 bit, giữ nguyên tính tương thích với các máy tính và các card mở rộng đã có.

Ở thế hệ PS/2, thế hệ sau của IBM PC/AT một bus hoàn toàn mới được áp dụng, bus Micro channel.

3.5.3. Bus PCI

Vào đầu năm 1992, Intel đã thành lập nhóm công nghệ mới. Nhằm nghiên cứu cải thiện các đặc tính kỹ thuật và những hạn chế của các bus hiện có như: bus ISA, bus EISA.

PCI (Peripheral Component Interconnect, liên kết các thành phần ngoại vi). Định chuẩn bus PCI đã được đưa ra vào tháng 6 năm 1992 và được cập nhật vào tháng 4 năm 1993, đã thiết kế lại bus PC truyền thống bằng cách bổ sung thêm một bus khác vào giữa CPU và bus I/O.

Bus PCI thường được gọi là bus mezzanine vì nó bổ sung thêm một tầng khác vào cấu hình bus truyền thống. PCI bỏ qua bus I/O tiêu chuẩn, nó sử dụng bus hệ thống để tăng tốc độ đồng hồ bus lên và khai thác hết lợi thế của đường dẫn dữ liệu của CPU.

Thông tin được truyền qua bus PCI ở 33MHz và độ rộng dữ liệu đầy đủ của CPU. Khi bus này được sử dụng để nối với CPU 32 bit, dải thông là 132 MBit/s, được tính theo công thức: $33\text{MHz} \times 32\text{bit}/8 = 132\text{MBit/s}$. Khi bus này được sử dụng với những hệ thống bổ sung 64 bit, dải thông tăng gấp đôi, nghĩa là tốc độ truyền

dữ liệu đạt tới 264MBs. Lý do chính mà bus PCI đã tăng tốc độ nhanh hơn các bus khác là nó có thể hoạt động đồng thời với bus vi xử lý. CPU có thể được xử lý dữ liệu trong các cache ngoại trú, trong khi bus PCI phải truyền thông tin liên tục giữa các thành phần khác của hệ thống, đây là ưu điểm thiết kế chính của bus PCI.

Định chuẩn PCI có ba cấu hình, mỗi cấu hình được thiết kế cho một kiểu hệ thống riêng biệt với những quy định nguồn riêng. Định chuẩn 5V cho những hệ thống máy tính văn phòng, định chuẩn 3,3V cho các hệ thống máy tính xách tay và những định chuẩn chung cho những bo mẹ và các card hoạt động trong hai kiểu ấy.

3.5.4. Bus nối tiếp chung USB

Bus USB (Universal Serial Bus) là một công nghệ bus mới đầy triển vọng, nhanh chóng phổ biến trong những thế máy tính ngày nay. Chủ yếu USB là cấp cho phép nối lên tới 127 thiết bị bằng cách sử dụng chuỗi xích. Tuy nhiên nó truyền dữ liệu không nhanh bằng FireWire, ở tốc độ 12MBs nó có khả năng đáp ứng cho hầu hết các thiết bị ngoại vi. Định chuẩn USB được đưa ra vào năm 1996 do một hội đồng gồm những đại diện của các nhà sản xuất máy tính lớn như Compaq, Digital, IBM, NEC và Northern Telecom.

Một ưu điểm nổi bật của USB là những thiết bị ngoại vi tự nhận dạng, một đặc trưng hết sức thuận lợi cho việc cài đặt, xác lập các thiết bị ngoại vi. Đặc trưng này hoàn toàn tương thích với những công nghệ PnP và cung cấp tiêu chuẩn công nghệ cho kết nối tương lai. Hơn nữa, những thiết bị USB có khả năng cắm nóng.

CHƯƠNG 4. HỆ THỐNG NHỚ MÁY VI TÍNH

4.1. Tổng quan

Một trong các hoạt động cơ bản của máy tính là lưu trữ dữ liệu dạng nhị phân. Các dữ liệu này là các chương trình hoặc số liệu mà Vi xử lý đưa ra hoặc đọc vào tùy theo yêu cầu. Bộ nhớ là các thiết bị để thực hiện nhiệm vụ lưu trữ dữ liệu của máy vi tính.

Mỗi ô nhớ được xác định bởi một địa chỉ. Thông thường mỗi ô nhớ có dung lượng là 1 byte. Các byte được ghép thành từ. Những máy 16 bit số liệu thì tổ chức 2 byte/từ, còn các máy 32 bit số liệu thì độ dài từ gấp đôi (4 byte/từ).

Trật tự các byte trong từ.

Có thể là từ phải sang trái (vi xử lý họ Intel) hoặc ngược lại từ trái sang phải (vi xử lý họ Motorola). Trường hợp dữ liệu lưu giữ là số nguyên thì hai cách sắp xếp trên không có trở ngại gì. Nhưng khi dữ liệu bao gồm cả số nguyên và cả xâu ký tự ... thì có vấn đề.

Ví dụ, xét một bản ghi (h 7.1) gồm có xâu là tên nhân viên BILL GATE và trường là số nguyên: tuổi 42. Xâu kết thúc bằng các byte 0 ở cuối để điền kín chỗ trống của từ, còn số nguyên thì được thêm vào các byte ở phần có trọng số cao hơn. Do vậy nếu dịch cách sắp xếp nọ sang cách kia của xâu giống như của số nguyên thì sẽ bị nhầm.

Mã phát hiện lỗi và sửa sai.

Số các vị trí bit khác nhau trong hai từ gọi là khoảng cách Hamming. Ví dụ, trong hai từ: 10001001 và 10110001 có khoảng cách Hamming bằng 3.

Để sửa sai, bên cạnh m số bit số liệu của từ, người ta thêm vào r bit dư (redundant bits) và chiều dài tổng của từ là n : $n = m + r$

Để phát hiện d bit lỗi đơn, cần dùng mã có khoảng cách d+1. Tương tự, để sửa lỗi d bit đơn, cần dùng mã có khoảng cách 2d+1. Ví dụ, dùng mã bit parity thêm vào byte số liệu, mã này có khoảng cách bằng 2, dùng để phát hiện 1 bit sai, nhưng không sửa được lỗi.

Trong truyền 1 khối ký tự, mỗi ký tự có một bit parity để kiểm tra. ở cuối mỗi khối, ta truyền thêm một ký tự là parity của toàn thể bản tin, gọi là longitudinal check (LRC). Phía thu sẽ tính LRC và so với LRC nhận được để kiểm tra lỗi. Một phương pháp nữa để kiểm tra lỗi khi truyền số liệu là dùng CRC (Cyclic redundancy check), đó là một đa thức nhị phân dư thu được khi chia đa thức các bit của bản tin cho một đa thức quy định.

Ví dụ mã sửa sai là mã có 4 từ dài 10 bit như sau:

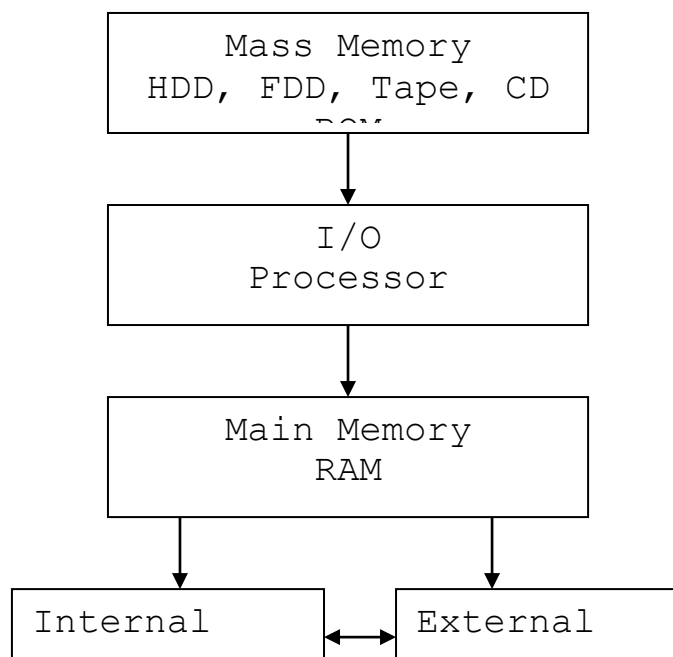
0000000000, 0000011111, 1111100000, 1111111111. Mã này có khoảng cách là 5, tức là nó có thể sửa được các lỗi kép. Ví dụ nếu ta nhận được từ 0000000111, máy thu sẽ biết rằng từ đó phải là 0000011111 (nếu coi như không có

nhiều hơn một lỗi kép). Nhưng nếu một lỗi ba xảy ra, biến 0000000000 thành 0000000111 thì ta không sửa lỗi được.

Để sửa lỗi, người ta dùng thuật toán của Hamming.

4.1.1. Đặc trưng của hệ thống nhớ

Xét một cách tổng thể, bộ nhớ của máy tính có kiến trúc theo cung bậc (hierarchy) trải dài từ bộ nhớ ngoài đến bộ nhớ trong và cuối cùng là đến bộ nhớ đệm (cache) trong và ngoài CPU.



Hình 4.1. Hieratchy của bộ nhớ trong máy vi tính.

4.1.2. Quản lý bộ nhớ (MMU, Memory Management Unit)

Công việc quản lý bộ nhớ của máy vi tính chủ yếu là do bộ vi xử lý đảm nhiệm. Bên cạnh đó còn có DMAC (Direct Memory Access Controller) cũng tham gia quản lý bộ nhớ trong việc truyền số liệu giữa controller ổ đĩa với bộ nhớ và làm tươi bộ nhớ. Ở những máy có Cache Memory thì Cache Memory Controller thực hiện các công việc truyền số liệu giữa Cache Memory và RAM.

Ở khu vực trung tâm của máy vi tính (bộ vi xử lý, ROM, RAM, các bus...), thực chất của việc quản lý bộ nhớ là các thanh ghi của vi xử lý đưa ra các địa chỉ của ô nhớ hoặc của cổng I/O qua bus địa chỉ, cùng các lệnh điều khiển/ trạng thái khác và đọc vào/ viết ra các số liệu của các ô nhớ ấy. Các bộ phận bên ngoài VXL sẽ giải mã các địa chỉ và các tín hiệu điều khiển/ trạng thái đó để trở vào các byte/ từ/ từ kép... của bộ nhớ để thực hiện các thao tác tương ứng.

Còn từ các ổ đĩa trở đi, việc quản lý bộ nhớ là thực hiện các lệnh coả hên điều hành lên các file (có địa chỉ 3 chiều là C-H-S), cụ thể là truyền số liệu nhờ DMAC giữa vùng đệm (buffer) của bộ điều khiển ổ đĩa với bộ nhớ RAM.

Các bộ vi xử lý Intel từ thế hệ 286 trở đi phân biệt hai mode địa chỉ: mode địa chỉ thực (chỉ quản lý 20 bit địa chỉ vật lý của bộ nhớ) và mode địa chỉ bảo vệ (quản lý tới 32 bit địa chỉ ảo nhờ các thanh ghi ẩn trong bộ vi xử lý).

Ở cấp dưới, tức cấp ngoại vi, như bộ điều khiển ổ đĩa, bộ điều khiển màn hình, máy in... cũng có tổ chức bộ nhớ riêng của chúng để tiện cho việc cất giữ và xử lý với các đặc thù riêng.

Các bộ nhớ RAM-ROM và các vùng nhớ của bộ nhớ ngoài (trên các ổ đĩa), khác nhau về cách mã hoá các bit, cách tổ chức, do đó cả cách truy nhập cũng khác nhau.

4.1.3. Tổ chức bộ nhớ của vi xử lý.

Bộ nhớ của vi xử lý có thể xem như bao gồm có bộ nhớ ROM và bộ nhớ RAM. Bộ nhớ RAM của vi xử lý chính là các thanh ghi (thanh ghi chung, thanh ghi chỉ số, thanh ghi đoạn, thanh ghi ngăn xếp, thanh ghi trạng thái, thanh ghi cờ, các bộ đệm số liệu/ địa chỉ/ điều khiển...). Còn bộ nhớ RAM là bộ phận giải mã lệnh để phát ra các vi lệnh.

Nhằm mục đích quản lý được số lượng địa chỉ nhớ (ảo) nhiều hơn số đường địa chỉ của bộ vi xử lý và bảo vệ các vùng nhớ của các nhiệm vụ khác nhau (task) và của hạt nhân (kernal) chống truy nhập không hợp pháp, các vi xử lý có các cách tổ chức đặc biệt các thanh ghi địa chỉ (bộ phận phân trang, điều khiển đoạn của các nhiệm vụ).

Các bộ vi xử lý từ thế hệ 486 trở đi còn có một bộ nhớ Cache Memory với kích thước nhiều Kbyte để chứa mảng các lệnh và số liệu đang thường dùng lấy từ bộ nhớ RAM, nhằm tăng tốc độ truy nhập.

Để tăng tốc độ tính toán các phép toán dấu chấm động, trong các bộ vi xử lý từ 486 trở đi còn có bộ phận dấu chấm động (FPU, Floating Point Unit), bộ phận này cũng có các thanh ghi FPU phục vụ riêng cho nó.

4.2. Bộ nhớ trong.

4.2.1. Khái niệm và phân loại

Các bộ nhớ có thể chia làm hai loại tổng quát, ROM và RAM. ROM là Read-only Memory(bộ nhớ chỉ đọc) và RAM là Random-access Memory (bộ nhớ truy xuất ngẫu nhiên). Nói chung ROM chứa các dữ liệu một cách cố định và không thể thay đổi. Còn RAM có thể đọc ra và có thể ghi vào.

Khái niệm truy xuất ngẫu nhiên có nghĩa là bất kỳ một vị trí nhớ nào cũng có thể được mở ra hoặc được gọi ra ở bất kỳ lúc nào, các thông tin không cần phải đọc ra hay ghi vào một cách tuần tự. Về thực chất, cả RAM và ROM đều là truy xuất ngẫu nhiên. Chỉ có điều khác nhau cơ bản là ROM chỉ cho phép đọc mà không thể ghi vào nó, còn RAM là bộ nhớ có thể đọc và ghi, vì thế RAM được gọi là “bộ nhớ đọc/ghi”.

Cấu trúc bộ nhớ

Hình 2-2 trình bày sơ đồ khối của một mạch nhớ. Mạch nhớ được nối với các bộ phận khác trong máy tính thông qua các đường dây địa chỉ và các đường dây dữ liệu của nó. Kiểm soát mạch nhớ bằng đường dây cho phép (enable), riêng đối với RAM còn có thêm đường dây kiểm soát đọc/ghi (Read/write).

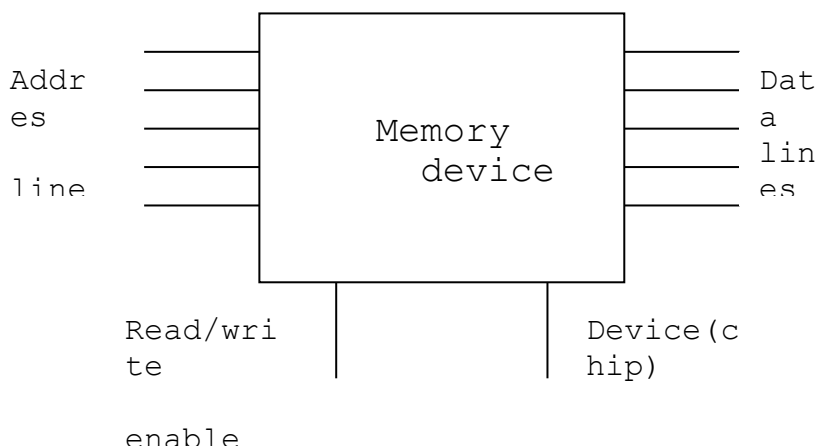
Các mạch nhớ nói chung được tổ chức dưới dạng ma trận, gồm những hàng và những cột để xác định vị trí hay địa chỉ nhớ. Mỗi ô trong ma trận gọi là một *phần tử* (cell) hay *vị trí nhớ* (memory location). Vị trí hay phần tử nhớ được dò tìm bằng cách chọn địa chỉ nhờ mạch giải mã địa chỉ. Mạch này gồm hai phần: **mạch chọn địa chỉ hàng RAS** (row-address selector) và **mạch chọn địa chỉ cột CAS** (Column-address selector). Các đường dây địa chỉ sẽ chọn địa chỉ hàng và cột. Đường dây enable dùng để mở các mạch điện lối ra bộ nhớ theo ba trạng thái. Còn đường dây Read/write quyết định dạng thao tác sẽ thực hiện.

Bộ nhớ hoặc là có tổ chức bit hoặc là loại có tổ chức lời (word organized). Bộ nhớ tổ chức bit có thể lưu giữ một bit đơn trong mỗi vị trí địa chỉ. Bộ nhớ tổ chức lời sẽ được lựa chọn cả một nhóm phần tử nhớ cùng một lúc với mỗi vị trí địa chỉ. Mỗi nhóm phần tử nhớ thường là một byte (8 bit), hoặc một lời (16 bit).

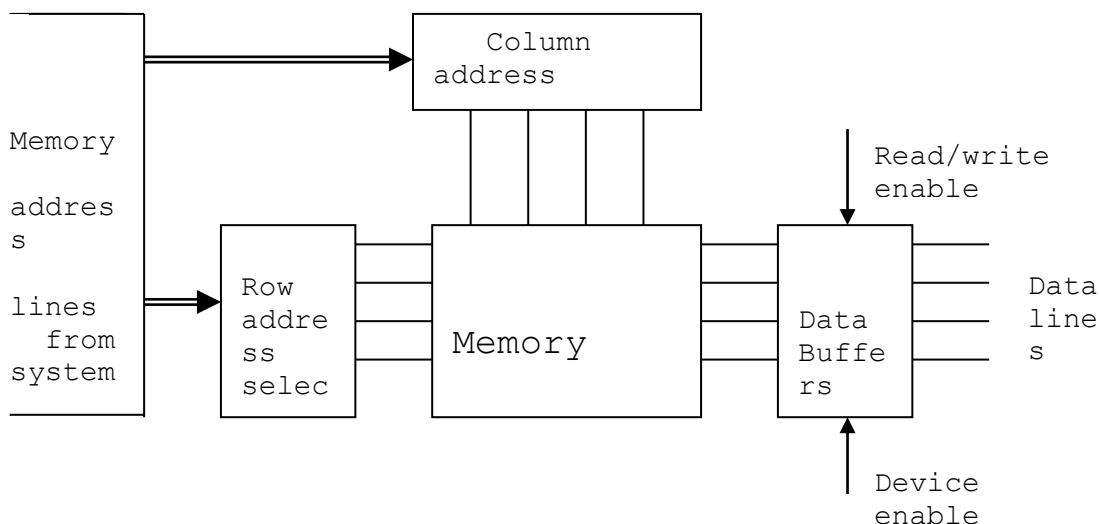
Số đường dây địa chỉ của mạch nhớ sẽ quyết định số vị trí nhớ cực đại tính theo công thức sau:

$$\text{Số vị trí nhớ cực đại} = 2^N.$$

trong đó, N là số lượng các đường địa chỉ.



a. Mạch nhớ cơ bản (basic memory device)



b. Sơ đồ khối (Block diagram)

Hình 4-2 *Mạch nhớ.*

4.2.2. ROM-BIOS.

Bất cứ hệ máy tính nào cũng có một vi mạch ROM. vi mạch này chứa chương trình của hệ điều hành vào ra cơ sở BIOS (basic input/output system). Những chương trình này cần thiết để khởi động máy và cài đặt chế độ làm việc cơ sở cho các thiết bị ngoại vi.

Nói chung, có thể chia ROM thành bốn loại. **ROM mặt nạ** (maskable ROM) là loại ROM do nhà sản xuất đã nạp sẵn dữ liệu, khi đó dữ liệu không thể thay đổi được nữa. **ROM có thể nạp chương trình** (PROM - programable ROM) là loại mạch mà người dùng có thể nạp dữ liệu vào thông qua thiết bị “đốt” PROM. Khi đã nạp thì các dữ liệu trong PROM cũng không thể thay đổi. **PROM có thể xóa**, còn gọi là EPROM (erasable PROM) là loại ROM mà người dùng có thể nạp dữ liệu vào và các dữ liệu đó có thể xóa hoặc thay đổi bằng một thiết bị đặc biệt. EPROM có thể xóa bằng điện (electric EPROM) là loại ROM có thể nạp và xóa dữ liệu bằng điện được mà không phải sử dụng tia cực tím như với EPROM.

Trong các máy tính hiện đại, người ta thường sử dụng Flash BIOS dùng EEPROM. Như vậy nội dung BIOS của máy tính có thể được thay đổi để tương thích với những mở rộng và nâng cấp hệ thống, mà điều này là không thể thực hiện đối với những máy tính thế hệ cũ sử dụng BIOS dùng PROM hoặc EPROM.

BIOS gồm nhiều chương trình và hàm. Phần đầu của chương trình BIOS kiểm tra hệ thống máy tính, quá trình này gọi là POST. Nếu hệ thống sử dụng các Card (thẻ cắm) Plug and Play thì giai đoạn này chính là lúc máy tính truy nhập tham số của thẻ. BIOS nào cũng có chương trình “Setup BIOS” để người dùng tự chỉnh tham số các thiết bị ngoại vi.

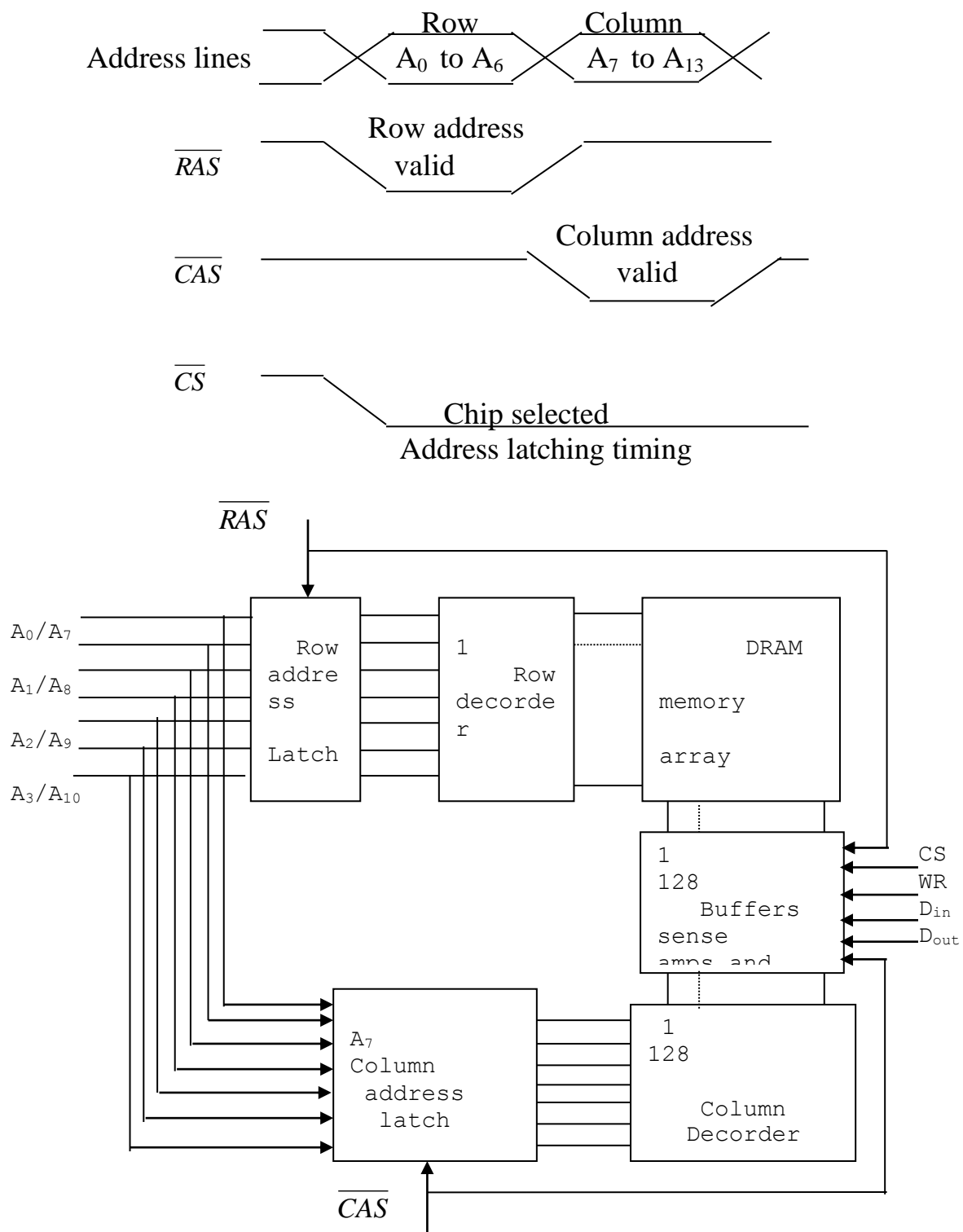
4.2.3. RAM.

Có thể chia RAM thành hai loại, RAM tĩnh (SRAM), có khả năng lưu giữ số liệu mãi mãi nếu như không mất nguồn nuôi. Và RAM động (DRAM), là loại RAM phải được “làm tươi” (refresh) tức là phải nạp lại dữ liệu đang được lưu trữ theo từng chu kỳ. “Làm tươi” bằng cách thực hiện thao tác đọc hoặc ghi nhắc lại. Cũng có thể “làm tươi” bằng những thao tác đặc biệt khác. Loại DRAM có mật độ phân tử nhớ cao nên giá thành khá rẻ so với SRAM. Các mạch nhớ DRAM được dùng phổ biến trong các thế hệ máy tính hiện nay.

Để tiết kiệm số đường địa chỉ và giảm số chân trên IC, hầu hết các loại DRAM đều dùng phương pháp địa chỉ multiplex. Trong quá trình đọc hay ghi các đường địa chỉ đầu tiên chứa các thông tin về hàng rồi tiếp sau mang thông tin về cột. Để kiểm soát thao tác này, người ta dùng đường dây \overline{RAS} và \overline{CAS} như trên

hình 4-3. Khi \overline{RAS} thấp thì thông tin trên các đường địa chỉ sẽ được mở thông qua mạch chốt địa chỉ hàng (row-address latch). Khi \overline{CAS} thấp thì thông tin trên các đường địa chỉ sẽ được mở thông qua mạch chốt địa chỉ cột (column-address latch).

Việc “làm tươi” bằng dữ liệu đọc, dữ liệu ghi hoặc bằng các thao tác riêng. Mạch điều khiển làm tươi phải chọn tuần tự từng hàng các phần tử nhớ, cứ mỗi hàng một lần, cho đến khi tất cả các hàng đều được “làm tươi”. Đó là phương pháp làm tươi từng đợt. Trong quá trình đó không được đọc hay ghi dữ liệu vào bộ nhớ cho đến khi kết thúc quá trình. Một cách khác là “làm tươi” từng hàng trong các chu kỳ rời rạc và gọi là làm tươi theo chu kỳ đơn.



Hình 4-3. Sơ đồ khối DRAM 16.384 bits(16Kb).

Bộ nhớ trong của máy tính dùng để chứa chương trình và số liệu của phần chương trình hạt nhân và các nhiệm vụ. Mỗi byte được gán cho một địa chỉ để vi lý và DMAC có thể truy nhập tới.

Bộ nhớ RAM ở những máy từ 386 trở đi có thể được tách riêng ra bộ nhớ đệm (cache memory), là RAM tĩnh với thời gian truy nhập nhanh, có kích thước dưới 1Mb được nối ngay vào bus nội bộ của máy tính sát ngay vi xử lý và được điều khiển bởi Cache controller. Phần còn lại là DRAM, chậm hơn nhưng rẻ hơn và có dung lượng lớn hơn. Hình 7.3 thể hiện sơ đồ khối bên trong một máy 386.

- Làm trọng tài bus (các việc về DMA và làm tươi bộ nhớ)
- Phát các tín hiệu địa chỉ hàng RAS và địa chỉ cột CAS đến các dãy nhớ của toàn bộ bộ nhớ DRAM trên MainBoard, phát tín hiệu ghi vào RAM
- Phát tín hiệu ready, tín hiệu Reset CPU
- Giao tiếp giữa đồng xử lý với CPU.

Controller ISA 82344 nối giữa bus local của CPU với bus hệ thống để làm các chức năng giao tiếp với CPU, system controller 346, data buffer 345, ROM, bus, các thiết bị ngoại vi như sau:

- Nhận các tín hiệu BE0# - BE3# của CPU, ROM8# và IOCHRDY từ bus ISA để sinh ra các tín hiệu chọn byte chẵn và byte lẻ SA0# và SBHE#
- Tạo các tín hiệu giao tiếp giữa 344, 345 và 346.
- Chứa khối điều khiển ngoại vi Peripheral Control gồm các vi mạch có độ tích hợp cực cao (VLSI) quen thuộc: hai 82C59 (ngắt), hai chip 82C37A (DMAC), vi mạch định thời 82C54, thanh ghi địa chỉ trang 74LS612, bộ driver cho loa, port B parallel I/O, đồng hồ thời gian thực và bộ đếm làm tươi bộ nhớ.
- Giải mã địa chỉ để tạo ra các tín hiệu chọn chip CS8042# cho controller bàn phím 8042 và ROMCS# để cho phép chọn ROM BIOS.

Vì mạch Peripheral Combo 82341 được ghép vào bus mở rộng của bus ISA, nó chứa các VLSI để thực hiện một số chức năng của các thiết bị ngoại vi sau đây:

- Hai cổng nối tiếp không đồng bộ 16C450
- Một cổng song song cho máy in
- Đồng hồ thời gian thực
- RAM sổ tay, các controller cho bàn phím và chuột.
- Interface cho đĩa cứng (tiêu chuẩn IDE).

Controller đĩa mềm 82077 có thể điều khiển tới 4 ổ đĩa mềm các loại 5"1/2 và 3"1/2.

4.2.4 Tổ chức bộ nhớ RAM của máy tính.

Xét trường hợp máy 386, nó có 32 bit địa chỉ, từ 00000000H đến FFFFFFFFH, ứng với 4 GByte không gian nhớ vật lý. Về quan điểm phần cứng, ta chia không gian đó thành 4 dãy nhớ độc lập nhau, là bank0 - bank3, mỗi bank kích thước 1 GByte. Chúng cần các tín hiệu Bank Enable BE0# tới BE3#. Trong hình 4.4 sau, ta thấy các địa chỉ A2 - A31 được đặt song song vào tất cả 4 bank nhớ. Còn mỗi bank nhớ chỉ cung cấp 1 byte số liệu cho 32 đường số liệu.

Ở chế độ thực, 386 chỉ dùng các đường địa chỉ A2 - A19 và 4 tín hiệu BE# dùng để chọn bank nhớ. Mỗi bank chỉ có 256 KByte.

Từ hình 4.4 ta thấy không gian nhớ vật lý được tổ chức thành dãy các từ kép (32bit). Do đó mỗi từ kép xếp đúng hàng (aligned) bắt đầu ở địa chỉ bội số của 4.

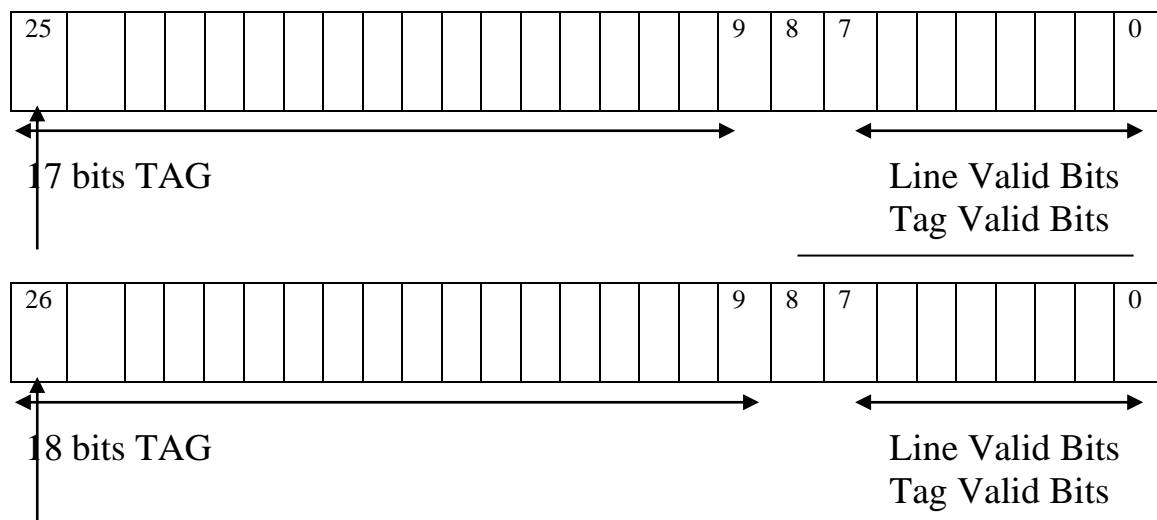
Dùng tổ hợp các tín hiệu BE# có thể truy nhập được vào các format khác nhau (byte, từ, từ kép) như hình 4.5. Việc truy nhập vào địa chỉ đầu của từ kép có

thể cần 1 chu kỳ bus (khi từ kép xếp đúng hàng) hoặc 2 chu kỳ bus (khi từ kép xếp lệch hàng, misaligned).

4.3. Hoạt động của Cache

Các hoạt động của Cache trực tiếp và Cache 2 đường được mô tả ở hình 4. 5 . Trong máy tính 386 toàn bộ không gian nhớ vật lý 4 GByte được chia thành $2^{17}-1$ trang nhớ 32 KByte. Vì máy 386 có tổ chức số liệu 32 bit, nên mỗi trang có 8Kb từ bép.

Controller chứa 1024 lối vào 26 bit, có tên là SET 0 - SET 1023 để chứa trạng thái của các ô nhớ của Cache Directory. trong trường hợp Cache trực tiếp, mỗi lối vào tương ứng với 8 dòng liên tiếp (từ kép) trong dãy nhớ Cache. Trong trường hợp Cache 2 đường, có hai Cache Directory là A và B ứng với các Bank A và Bank B của nhớ Cache, mỗi Bank chứa 4 KByte từ kép, do đó trong Controller chứa hai tập lối vào (Set Entry) dài 27 bit. Mỗi Set chỉ có 512 lối vào. Định dạng của thông tin đưa tới các lối vào gồm có 8 bit Line Valid Bits, Tag Valid Bit và Tag 17 bit (với Cache trực tiếp), 18 bit (với Cache 2 đường).



Hình 4.5. Format của Entry SET của Cache Directory trực tiếp và hai đường.

Phần TAG dài 17/18 bit chỉ ra số hiệu của 1 trong 131972 trang 32 KB (hoặc 262144 trang 16 KB) trong bộ nhớ chính. Còn TAG_BIT chỉ ra TAG có hữu hiệu hay không. Nếu TAG_BIT = 0 thì tất cả các dòng trong SET là không hữu hiệu. Nếu TAG_BIT = 1 thì mỗi bit trong 8 bit của LINE_VALID_BITS bằng 1 có nghĩa rằng dòng tương ứng trong Cache chứa thông tin hữu hiệu, tức là thông tin trong đó sẽ được cập nhật tự động.

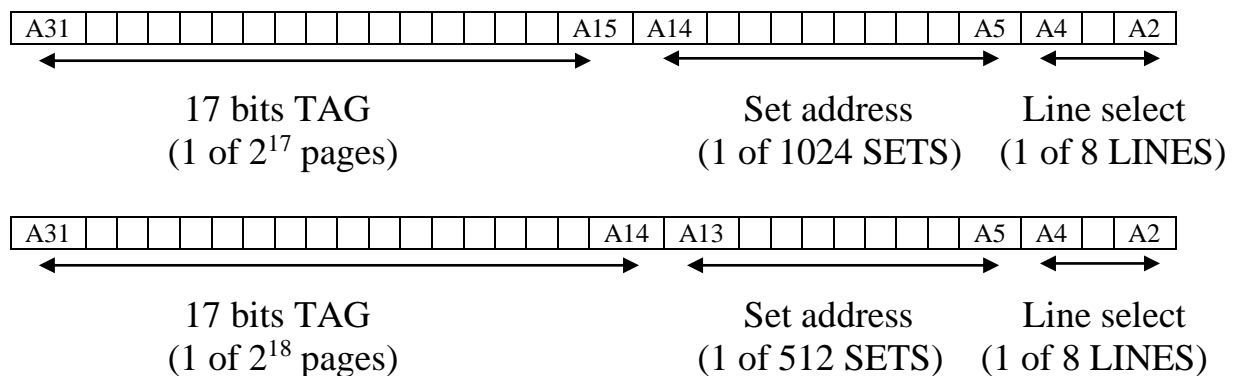
Ví dụ: Nếu SET 1 = 00005FFh, ta chuyển sang dạng nhị phân:
 SET 1 = 0000 0000 0000 0000 0101 1111 1111. Từ đó ta có:
 TAG = 0000 0000 0000 0000 010 = $2_{(10)}$

TAG_VALID = 1, do đó những dòng trong LINE_VALID_BIT = 1111 1111 sẽ hữu hiệu. Tức là tất cả 8 dòng trong Cache đều hữu hiệu.

*Cache trực tiếp.

Khi VXL 386 bắt đầu chu kỳ đọc nhớ, nó cấp địa chỉ song song ra cho 3 nơi là Latch địa chỉ của local bus của controller, lối vào địa chỉ của controller và interface nhớ Cache. khi đó, Cache Controller quyết định là VXL cần đọc từ bộ nhớ chính hay từ Cache. Nó thực hiện điều đó bằng cách thông dịch địa chỉ và so sánh với ENTRY của Cache Directory.

Hình 4.6. là các trường (field) của bit địa chỉ cho Cache trực tiếp cả Cache hai đường. Trong đó 17/ 18 bit lớn nhất A15 - A31 (hoặc A14 - A31) là TAG để chỉ ra trang của bộ nhớ chính cần đọc thông tin từ đó vào VXL. Các bit tiếp theo, A5 - A14 (hoặc A5 - A13) gọi là địa chỉ của SET của nhớ Cache, chỗ cần truy nhập vào. Còn 3 bit bé nhất A2 - A4 để chọn dòng trong SET.



Hình 4.6 . Các trường bit địa chỉ dùng cho Cache trực tiếp và hai đường.

Khi một địa chỉ do VXL đặt vào lối vào địa chỉ của Controller, phần SET của địa chỉ đó được dùng để chọn 1 trong 1024 ENTRY của SET trong Cache Directory. Sau đó Controller tiến hành 3 kiểm tra như sau:

- So sánh trường TAG trong địa chỉ với TAG trong ENTRY của SET đã được chọn, chúng phải trùng nhau.
- Bit TAG_VALID_BIT của ENTRY SET được chọn phải bằng 1.
- LINE_VALID_BIT của ENTRY tương ứng với giá trị trong phần LINE_SELECT của địa chỉ phải = 1.

Nếu cả ba điều kiện trên thoả mãn thì thông tin cần phải đọc từ bộ nhớ đã được lưu trong bộ nhớ Cache và hữu hiệu. Và Controller khởi đầu chu kỳ đọc dữ liệu từ Cache thay vì từ bộ nhớ chính. Đây là trường hợp trúng Cache.

Nếu hai điều kiện đầu thoả mãn, còn LINE_VALID_BIT = 0 thì trượt Cache, tức là ENTRY của SET trong Directory tương ứng với trang đúng của nhớ

chính, nhưng dòng từ kép cần phải đọc vào VXL lại chưa được chuyển sang Cache, gọi là trượt dòng. Khi đó VXL phải đọc từ bộ nhớ chính một từ kép, đồng thời được đưa vào nhớ Cache và LINE_VALID_BIT trong ENTRY của Cache Directory được xác định bằng 1. Do đó thông tin được đọc vào Cache và đánh dấu là hữu hiệu.

Nếu trong khi kiểm tra hoặc các TAG không khớp hoặc TAG_VALID_BIT = 0 thì xảy ra trượt TAG (tag miss). Đó là trường hợp đọc một trang đã không được Cache, hoặc đã Cache nhưng không hữu hiệu. Trong trường hợp này Controller phải khởi đầu một chu kỳ đọc từ bộ nhớ chính viết vào bộ nhớ Cache. Lúc đó TAG trong SET ENTRY của Directory được cập nhật bằng phần TAG của địa chỉ, TAG_VALID_BIT được lập bằng 1, một LINE_VALID_BIT do địa chỉ trở ra được lập bằng 1, một LINE_VALID_BITS bị xoá đi. Bằng cách này một trang hữu hiệu và ENTRY dòng được lập nên và tất cả các ENTRY khác trong SET baay giờ tương ứng với thông tin trong một trang khác của nhớ chính trở nên không hữu hiệu.

*** Cache hai đường.**

Ở các hình 4.6 đã nêu ra cách tổ chức nhớ Cache, cùng các format của ENTRY SET, các trường địa chỉ của cả hai trường hợp Cache trực tiếp và Cache hai đường.

Trong trường hợp (hình 4. 6) Cache hai đường ngoài hai Directory A và B ứng với hai bộ ENTRY, còn có thêm 512 cờ Least Recently Used dài 1 bit (LRU bit). Những cờ này theo dõi xem BANK A hoặc BANK B đang giữ thông tin lâu không sử dụng. Những cờ này được Controller kiểm tra bằng thuật toán thay thế những thông tin lâu không dùng.

Thao tác đọc thông tin từ nhớ Cache hai đường cũng giống như ở Cache trực tiếp. Biết rằng (ở sơ đồ h) SET_ADDRESS chỉ có 9 bit. Đầu tiên địa chỉ 9 bit này được dùng để chọn 1 trong 512 lối vào SET của cả hai Directory A và B. Tiếp theo TAG_ADDRESS 18 bit được so sánh với TAG trong mỗi lối vào SET, TAG_VALID_BITS được kiểm tra, và LINE_VALID_BIT tương ứng với mã của LINE_SELECT (A2 đến A4) được kiểm tra trong mỗi lối vào SET. Nếu ba điều kiện kiểm tra được thoả mãn đối với một trong hai lối vào SET thì ta nói là trúng Cache và thông tin của dòng được đọc vào VXL từ BANK tương ứng của nhớ Cache.

Mặt khác, sẽ xảy ra trượt Cache nếu không khớp các TAG hoặc nếu cả hai VALID_BIT bị xoá, hoặc nếu LINE_VALID_BIT không được lập trong bất cứ lối vào nào, khi đó algorithm sẽ kiểm tra bit cờ LRU đối với SET được chọn bởi địa chỉ SET để xác định xem lối vào của BANK A hay BANK B là lâu không được dùng hơn, sau đó thông tin được đọc vào từ bộ nhớ chính và viết vào BANK nhớ nào lâu không được dùng.

Làm tươi bộ nhớ DRAM

Bộ nhớ DRAM có các hàng cần phải được làm tươi trong mỗi chu kỳ 2mS. Mạch làm tươi trong chip nhớ phải kiểm tra điện áp các ô nhớ, nếu nó lớn hơn $V_{cc}/2$ thì nạp nó tới V_{cc} , nếu bé hơn $V_{cc}/2$ thì xả hết về 0V.

Để đọc một từ từ BANK nhớ DRAM, trước hết DRAM Controller hoặc một mạch khác cấp tín hiệu $WE\# = 1$. Sau đó gửi nửa thấp của địa chỉ, ứng với địa chỉ hàng, rồi tín hiệu $RAS\# = 0$. Sau 1 thời gian, controller cấp nửa địa chỉ cao, ứng với địa chỉ cột, rồi tín hiệu $CAS\# = 0$. Sau thời gian nhất định, từ cần có sẽ xuất hiện trên Output Data của nhớ.

Để viết vào DRAM, các tín hiệu cũng tương tự, ngoại trừ sau tín hiệu $CAS\# = 0$, controller cấp $WE\# = 0$ để quy định viết vào RAM.

Controller làm tươi DRAM bằng cách gửi ra mỗi địa chỉ trong 512 địa chỉ hàng và cấp $RAS\# = 0$ theo chu kỳ, khoảng 4mS. Việc làm tươi được tiến hành hoặc theo *burst mode* hoặc theo *distributed mode*. Trong burst mode toàn bộ 512 hàng được định địa chỉ và đánh nhịp lần lượt cách nhau 4mS. Còn ở distributed mode hàng được định địa chỉ và đánh nhịp sau 4/512 mS.

Những nhiệm vụ chính của việc điều khiển nhớ DRAM của máy tính là:

- Làm tươi mỗi ô nhớ sau một khoảng thời gian vài mS.
- Cấp hai nửa địa chỉ cùng các tín hiệu $RAS\#$, $CAS\#$ thích hợp.
- Bảo đảm thao tác đọc/viết và làm tươi không xảy ra đồng thời.
- Cấp tín hiệu đọc/viết để điều khiển chiều số liệu.

Mô tả sơ đồ Controller 8208 làm tươi 1 MByte cho hệ VXL 8086. Bộ nhớ chia thành 2 BANK (mỗi BANK 8 bit). Controller bảo đảm cấp các địa chỉ hàng và địa chỉ cột, tín hiệu $RAS\#$, $CAS\#$, và các tín hiệu READ/WRITE. Các chân trạng thái ra S0 - S3 của VXL đấu thẳng tới các chân vào của 8208. Controller giải mã các tín hiệu này để cho ra các tín hiệu đọc và viết mà VXL yêu cầu. Do đó, đa số thời gian của VXL được dùng để đọc byte/từ của RAM mà không cần có các chu kỳ chờ. Nếu trong khi 8208 đang ở giữa chu kỳ làm tươi nhớ mà VXL muốn đọc RAM thì 8208 lưu giữ AACK cao và buộc VXL cấp thêm một chu kỳ đợi để 8208 kịp hoàn thành chu kỳ làm tươi. Để tiết kiệm chân, không có các chân số liệu (để nạp từ điều khiển), chân PDI nối mass sẽ cho phép 8208 tự khởi đầu hoạt động trong đa số các ứng dụng. Còn các trường hợp khác thì chân PDI sẽ được điều khiển bởi một thanh ghi dịch vào song song - ra nối tiếp, nhờ đó từ điều khiển được nạp vào 8208. Sau khi Reset chân $WE/PCLK$ sẽ cấp ra một dãy xung đánh nhịp cho từ điều khiển từ thanh ghi dịch nạp vào 8208. Từ điều khiển được thực hiện bằng nối ở lối vào của thanh ghi dịch.

Ta cũng có thể dùng DMAC để làm tươi bộ nhớ. Hình 7. là ví dụ mạch 4 BANK với dung lượng 256KB nhớ. Ở đây máy tính dùng chế độ đọc DMA ảo. Bộ định thời 8253 lập trình để phát xung nhịp 15 μ S. Xung này được nối vào một trong các lối vào xin DMA (DMA Request) là DREQ0 của 8237 DMAC được lập trình để đọc từ nhớ và viết vào một cổng không tồn tại. Khi DMAC nhận xung này, nó gửi một tín hiệu HOLD_REQUEST tới VXL rồi VXL trả lời bằng tín hiệu HLDA

và đặt các chân của nó ở trạng thái trở kháng cao. Khi đó: 8237 chiếm lấy bus, gửi ra các địa chỉ nhớ, tín hiệu đọc nhớ và tín hiệu chấp nhận DMA kênh 0 (DACK0).

Tám bit địa chỉ thấp gửi tới nhớ, còn DACK0 để cung cấp xung RAS# cho các bank DRAM để làm tươi nhớ động. Sau mỗi thao tác DMA thanh ghi địa chỉ hiện hành trong DMAC được tự động tăng/giảm (tùy thuộc cách lập trình lúc đầu) để làm tươi hàng (row) nhớ sau. Nếu 8237 lập trình để truyền 64 kByte, khởi đầu ở địa chỉ 0, tăng đếm sau mỗi lần DMA, và tự khởi động (autoinitialize), thì dãy các địa chỉ gửi ra sẽ làm tươi tất cả 256 trong hàng DRAM. Mỗi hàng làm tươi 15ns.

Ví dụ với tần số clock 4.77MHz dùng trong IBM PC, một chu kỳ DMA để làm tươi mất 820 ns mỗi 15 ns, tức 5% thời gian của VXL.

Để kiểm tra Parity mỗi bank nhớ có 9 bit, 8 bit để giữ số liệu, bit thứ 9 là bit Parity. Mỗi mạch 74 LS280 dùng để phát/ kiểm parity cho mỗi byte và cất vào parity bit mỗi khi byte được viết vào nhớ. Khi 9 bit được đọc ra, parity được kiểm tra. Nếu parity sai thì tín hiệu báo lỗi sẽ được gửi tới cổng 8255 để cho VXL đọc. Khi bắt đầu bật máy, thì quá trình POST xảy ra, nó viết mẫu byte vào tất cả ô nhớ, rồi kiểm tra bằng cách đọc lại chúng cùng với parity bit.

Chuyển một mảng số liệu bằng DMA

Thường xuyên có các nhu cầu chuyển mảng số liệu nhớ và ngoại vi. Lúc đó ta dùng DMAC. Hình mô tả cơ chế hoạt động của DMAC với VXL để truyền số liệu giữa nhớ và ngoại vi (ổ đĩa thông minh).

Khi ta bật máy lúc đầu các khoá ở vị trí đóng từ VXL tới ngoại vi, và nhớ. Chúng ta lập trình để chạy DMAC, ví dụ để đọc file từ ổ đĩa để viết vào nhớ. Muốn thế phải gửi một loạt lệnh tới controller ổ đĩa yêu cầu nó đọc những block dữ liệu từ đĩa. Khi controller đã có byte đầu tiên, nó gửi DMA Request(DREQ) cho DMAC, nếu channel đó của DMAC không bị che chắn, DMAC gửi HOLD REQUEST tới chân HOLD của VXL, VXL treo các bus cao và gửi ra HLDA cho DMAC, khi DMAC nhận HLDA của VXL, nó cho tín hiệu điều khiển để đặt ba khoá về vị trí DMA, cất VXL ra, sau đó DMAC cho ra địa chỉ cấp cho nhớ, DMAC gửi DMA-Acknowledge (DACK0) cho ổ đĩa để nó đưa ra số liệu, cuối cùng nó cấp MEMW#=0 và IOR#=0 ra bus điều khiển, nhờ vậy liệu được đọc vào từ ngoại vi và viết ra ô nhớ, khi truyền số liệu hoàn thành DMAC thu lại tín hiệu HRQ, do đó VXL lấy lại các bus của nó cho đến lần DMA sau.

Hình là mạch chi tiết của sơ đồ hình. Trong đó 8237 là DMAC còn 8272 là controller ổ đĩa mềm, 8282 dùng để latch 8 bit địa chỉ gửi ra từ VXL (do ALE của 8086 điều khiển) hoặc 8237 (do AEN và AD dress STrobe điều khiển).

Khi đóng điện DMAC cấp AEN = 0, các vi mạch U1, U2, U4 được hữu hiệu. Và ALE từ VXL được dùng để đánh nhịp (STroBe) cho 3 vi mạch này. Do

đó chúng chốt các địa chỉ A0-A19 của VXL ra bus địa chỉ như trường hợp thông thường (không DMA).

Khi DMAC muốn chiếm lấy các bus, nó cấp AEN= 1, dẫn đến:

- Khoá không cho U1 làm việc, cắt các địa chỉ A0 -A7 từ VXL, DMAC trực tiếp cấp ra 8 địa chỉ thấp cho nhớ trong truyền số liệu,

- AEN =1 làm đổi vị trí Multiplex khiến cho việc đánh nhịp cho U2 thực hiện bởi ADSTB của DMAC. Để tiết kiệm chân, DMAC 8 bit địa chỉ cao qua các chân số liệu D0-D7, cùng với ADSTB=1 báo rằng đó là các địa chỉ cao A15- A8 do DMAC cấp cho qua nhớ latch U2.

- Cũng do AEN =1, các bit A16- A19 do U3 cấp từ các bit D10 -D13 do ta lập trình cứng .

- Cuối cùng, các tín hiệu điều khiển được đổi nối từ các output của VXL sang các output của DMAC (gồm IOR#, IO#, MEMU#, MEMR#).

Các buffer số liệu hai chiều 8286 cho phép có thể truyền 8 bit số liệu tới/từ controller đĩa từ/tới hoặc byte cao hoặc byte thấp của bộ nhớ. Bit địa chỉ A0 dùng để chọn đường cho hai byte nhớ chẵn/lẻ đó.

DMAC có 4 kênh (channel), nhiều thanh ghi trong để:

- | | |
|---------------------------------|---------------------------------|
| -Ghi địa chỉ nhớ cơ sở(16 bit). | -Ghi số đếm từ (word) nhớ cơ sở |
| -Địa chỉ nhớ hiện hành . | -Ghi địa chỉ tạm thời |
| -Ghi số đếm tạm thời. | -Ghi trạng thái |
| -Ghi địa chỉ lệch | -Ghi tạm thời |
| -Các thanh ghi mode | -Ghi chặn DMA |
| -Ghi yêu cầu xin DMA | |

DMAC có 4 chân địa chỉ và 2 bit vào IOR#, IOW# để điều khiển hoạt động đọc/viết các thanh ghi của nó. Nó còn có một flip flop để trữ địa chỉ byte cao/byte thấp đang có ở 8 chân số liệu của nó. Các flip flop này được lần lượt tự động lật trạng thái để cho phép cấp ra 16 bit địa chỉ nhờ chỉ một cổng 8 bit. Tất nhiên để điều khiển hoạt động của DMAC cần phải lập trình khởi đầu nó, và lập trình các hoạt động sau đó của nó. DMAC có thể lập trình để truyền 1 byte cho mỗi request, 1 khối các byte cho mỗi request, hay truyền cho đến khi nhận được 1 tín hiệu dừng từ chân vào/ra EOP#.

Đại thể phải làm các việc sau:

- Viết từ điều khiển vào địa chỉ trong 1101 để xoá flip flop trong
- Viết từ điều khiển vào địa chỉ trong 1000
- Viết từ mode cho mỗi channel (dùng địa chỉ trong 1011)
- Viết ra địa chỉ nhớ đầu tiên tới địa chỉ trong của thanh ghi cơ sở cho mỗi channel ta cần
- Viết ra số byte ta muốn truyền tới địa chỉ trong của thanh ghi đếm số lượng từ cơ sở cho mỗi kênh

- Viết từ/ các từ điều khiển để xoá mặt nạ cho channel/ các channel cần dùng.

Table a

SIGNALS						OPERATION
A3	A2	A1	A0	$\overline{\text{IOR}}$	$\overline{\text{IOW}}$	
1	0	0	0	0	1	READ STATUS REGISTER
1	0	0	0	1	0	WRITE COMMAND REGISTER
1	0	0	1	0	1	ILLEGAL
1	0	0	1	1	0	WRITE REQUEST REGISTER
1	0	1	0	0	1	ILLEGAL
1	0	1	0	1	0	WRITE SINGLE MASK REGISTER SET
1	0	1	1	0	1	ILLEGAL
1	0	1	1	1	0	WRITE MODE REGISTER
1	1	0	0	0	1	ILLEGAL
1	1	0	0	1	0	CLEAR BYTE POINTER FLIP/ FLOP
1	1	0	1	0	1	READ TEMPORARY REGISTER
1	1	0	1	1	0	MASTER CLEAR
1	1	1	0	0	1	ILLEGAL
1	1	1	0	1	0	CLEAR MASK REGISTER
1	1	1	1	0	1	ILLEGAL
1	1	1	1	1	0	WRITE ALL MASK REGISTER BITS

Table b

NAME	SIZE	NUMBER
BASE ADDRESS REGISTER	16 BITS	4
BASE WORD COUNT REGISTER	16 BITS	4
CURRENT ADDRESS REGISTER	16 BITS	4
CURRENT WORD COUNT REGISTER	16 BITS	4
TEMPORARY ADDRESS REGISTER	16 BITS	1
TEMPORARY WORD COUNT REGISTER	16 BITS	1
STATUS REGISTER	8 BITS	1
COMMAND REGISTER	8 BITS	1
TEMPORARY REGISTER	8 BITS	1
MODE REGISTER	6 BITS	4
MASK REGISTER	4 BITS	1
REQUEST REGISTER	4 BITS	1

4.4. Bộ nhớ ngoài

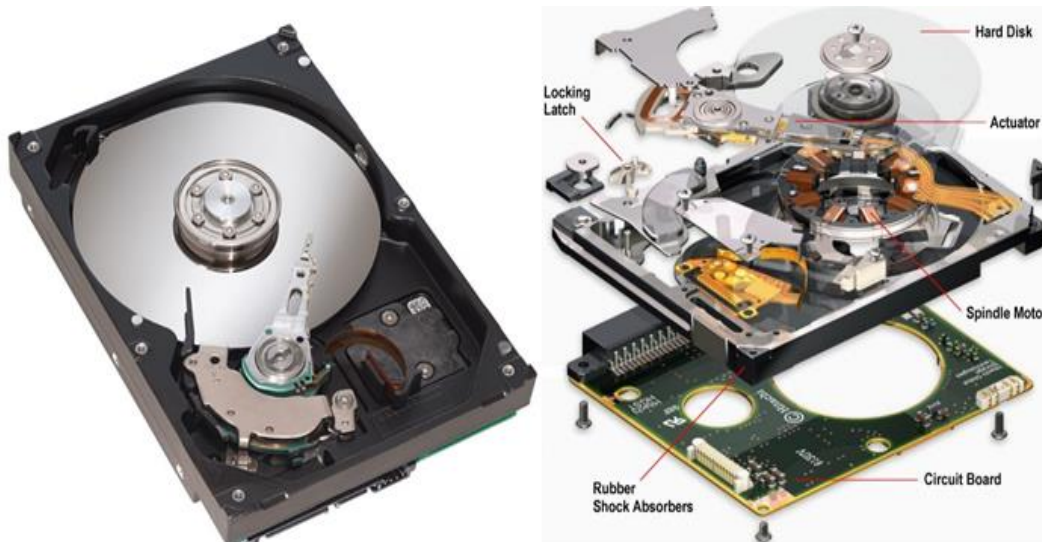
Ổ đĩa cứng (Hard disk driver- **HDD**) là thiết bị dùng để lưu trữ dữ liệu trên bề mặt các tấm đĩa hình tròn phủ vật liệu từ tính.

Ổ đĩa cứng là thiết bị dùng để lưu trữ thông tin dạng cố định. - "không thay đổi" (*non-volatile*), có nghĩa là chúng không bị mất dữ liệu khi ngừng cung cấp nguồn điện cho chúng.

Cấu tạo

- Bộ khung: làm bằng chất liệu nhôm, plastic để định vị, bảo đảm độ kín.

- Đĩa từ: làm bằng nhôm, hợp chất gốm và thủy tinh, 2 mặt được phủ lớp từ tính và lớp bảo vệ, được gắn trên cùng 1 trục.
- Đầu đọc/ghi: dùng đọc/ ghi dữ liệu, mỗi mặt đĩa có một đầu đọc riêng.
- Mạch điều khiển: truyền tín hiệu giữa máy tính và HDD.
- Cache: bộ nhớ đệm để lưu dữ liệu tạm thời.
- Moto: trục quay để làm quay đĩa từ.
- Landing Zone: vị trí tạm ngưng của đầu đọc/ ghi.
- Track: là những vòng tròn đồng tâm trên mỗi mặt đĩa.
- Sector: (cung) là phần tử trên track, mỗi sector có kích thước 512 byte để chứa dữ liệu.
- Cylinder: tập hợp những track đồng tâm của tất cả các lá đĩa.
- Cluster: tập hợp nhiều sector.



Hình 4-5. Cấu tạo bên trong ổ đĩa cứng

CHƯƠNG 5. HỆ THỐNG VÀO RA VÀ THIẾT BỊ NGOẠI VI

5.1. Tổng quan

5.1.1. Thiết bị ngoại vi

Chức năng: Chuyển đổi dữ liệu giữa bên trong và bên ngoài máy tính

Các thiết bị ngoại vi cơ bản

Thiết bị vào: bàn phím, chuột, máy quét...

Thiết bị ra: màn hình, máy in...

Thiết bị vào ra: các ổ đĩa...

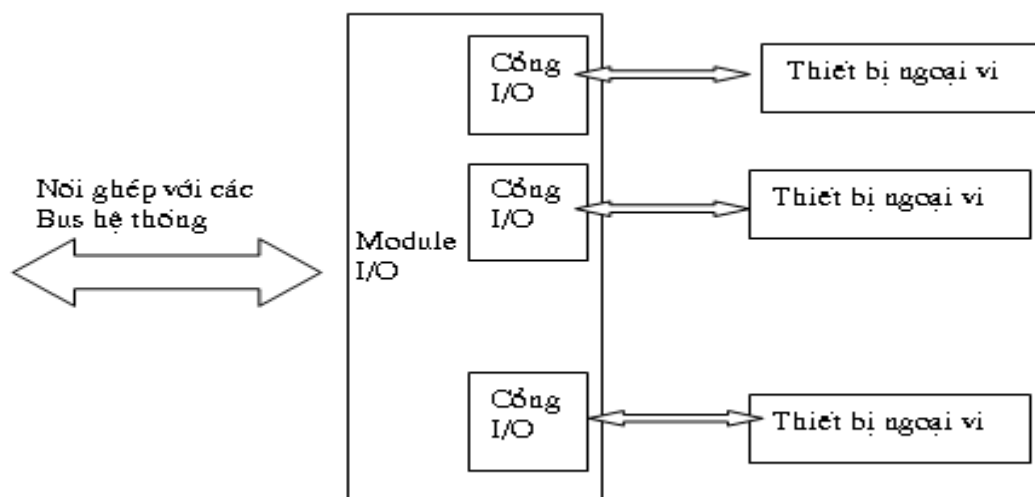
Thiết bị truyền thông: Modem, Nic . .

5.1.2. Module vào ra

Chức năng: Trao đổi thông tin giữa máy tính với bên ngoài

Các thao tác cơ bản: vào dữ liệu, ra dữ liệu

Các thành phần chính: Các thiết bị ngoại vi và các module vào ra (I/O module)



Hình 5-1: Sơ đồ liên kết của hệ thống vào ra

5.2. Vai trò của bộ phối ghép

Bộ phối ghép nằm trung gian giữa máy vi tính và các thiết bị ngoại, đóng vai trò trung chuyển dữ liệu (nhận và truyền) giữa chúng.

Khi truyền dữ liệu từ máy vi tính ra thiết bị ngoại, bộ phối ghép đóng vai trò nhận dữ liệu từ máy tính và là nguồn cấp dữ liệu cho thiết bị ngoại.

Khi truyền dữ liệu từ thiết bị ngoại vào máy vi tính, bộ phối ghép đóng vai trò nhận dữ liệu từ thiết bị ngoại và là nguồn cấp dữ liệu vào cho máy tính.

Nhiệm vụ của bộ phối ghép.

Bộ phối ghép làm nhiệm vụ phối hợp trao đổi dữ liệu giữa máy tính và thiết bị ngoài về mức và công suất của tín hiệu, về dạng tín hiệu, về tốc độ và phương thức trao đổi.

a. Phối hợp về mức và công suất tín hiệu

Mức tín hiệu của máy vi tính thường là mức (0V, 5V) trong khi của các thiết bị ngoài, hoặc ở mức cao ($\pm 15V$, $\pm 48V$) hoặc rất thấp ($\ll 1V$). Do đó, bộ phối ghép phải biến đổi các mức trên cho phù hợp.

Công suất của các tín hiệu trên bus dữ liệu của máy vi tính rất nhỏ (cỡ vài chục mA), trong khi cần công suất lớn hơn nhiều cho thiết bị ngoài. Do đó bộ phối ghép phải biến đổi công suất cho phù hợp.

Ở các ngõ vào và ngõ ra của bộ phối ghép thường dùng các mạch đệm ba trạng thái.

b. Phối hợp về dạng dữ liệu (tín hiệu).

Bộ phối ghép phải đảm bảo tính tương thích về cơ chế trao đổi dữ liệu giữa máy tính và thiết bị ngoài.

c. Phối hợp về tốc độ trao đổi dữ liệu.

Máy tính thường hoạt động với tốc độ cao, trong khi các thiết bị ngoài thường hoạt động chậm hơn. Do đó bộ phối ghép phải có khả năng cấp, nhận dữ liệu nhanh với máy tính, nhưng với thiết bị ngoài thì ngược lại.

d. Phối hợp về phương thức trao đổi dữ liệu.

Để đảm bảo sự trao đổi dữ liệu một cách tin cậy, cần có bộ phối ghép và phương thức trao đổi dữ liệu diễn ra theo một trình tự nhất định và hợp lý.

- *Nếu việc trao đổi dữ liệu do máy tính yêu cầu thì quá trình diễn ra như sau:*

Máy tính đưa lệnh điều khiển để khởi động bộ phối ghép hay thiết bị ngoài.

Máy tính đọc tín hiệu trả lời. Nếu có tín hiệu sẵn sàng mới trao đổi tin, nếu không, thêm một chu kỳ chờ và đọc lại trạng thái.

Máy tính trao đổi tin khi đọc thấy trạng thái sẵn sàng.

- *Nếu việc trao đổi tin do TBN yêu cầu:* để giảm thời gian chờ đợi trạng thái sẵn sàng của TBN, máy tính có thể khởi động TBN rồi thực hiện các nhiệm vụ khác. Việc trao đổi tin diễn ra khi:

TBN gửi yêu cầu trao đổi tin tới bộ xử lý ngắt của khối ghép nối, để đưa yêu cầu ngắt chung trình đến máy tính.

Nếu có nhiều thiết bị ngoài cùng gửi yêu cầu, KGN xử lý theo mức ưu tiên ngắt định trước, rồi đưa yêu cầu trao đổi tin cho máy tính.

Máy tính nhận yêu cầu, chuẩn bị trao đổi và gửi tín hiệu xác nhận sẵn sàng trao đổi.

KGN nhận và truyền tín hiệu xác nhận cho TBN.

TBN trao đổi tin với KGN và KGN trao đổi tin với máy tính (nếu là đưa tin vào) hoặc máy tính trao đổi tin với KGN và KGN trao đổi tin với TBN (nếu là đưa tin ra).

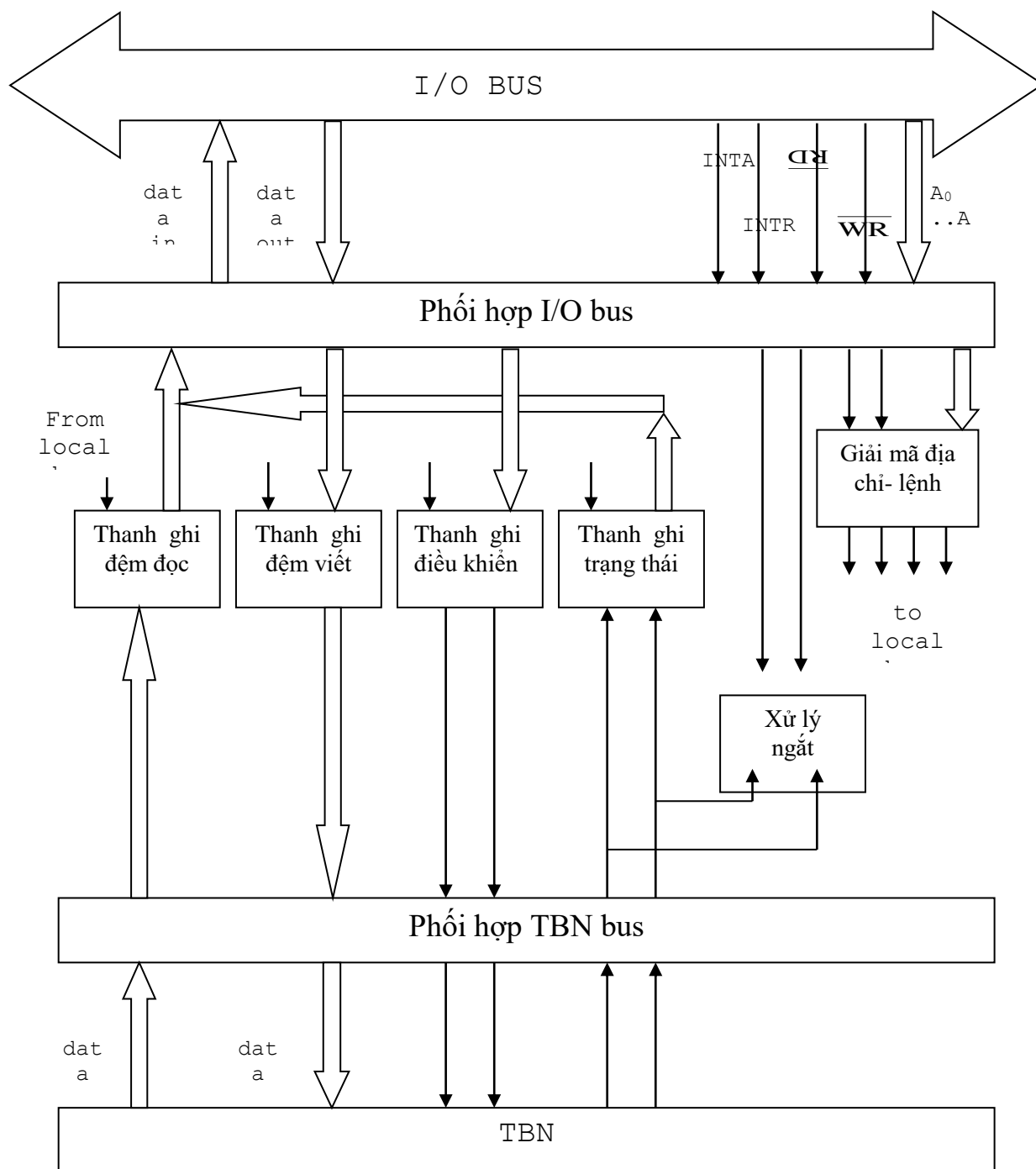
5.3. Cấu trúc chung của khối ghép nối

5.3.1. Nhiệm vụ của các khối trong khối ghép nối (KGN).

KGN có nhiệm vụ chung là nhận và chuyển tin giữa máy tính và TBN. Nhưng cụ thể, có những nhiệm vụ nhỏ khác nhau trong sơ đồ khối. Những nhiệm vụ và các khối tương ứng là:

- + *Ghép nối và biến đổi tin giữa MT - KGN và KGN - TBN về:*
 - Mức và công suất tín hiệu.
 - Dạng tin (song song, nối tiếp, tín hiệu số, tín hiệu analog).
- + *Giải mã địa chỉ, giải mã lệnh cho các thanh ghi đệm của KGN.*
- + *Ghi nhận trạng thái TBN hay yêu cầu trao đổi tin của TBN, xử lý yêu cầu ưu tiên, gửi yêu cầu vào MT và xác nhận trao đổi tin từ MT.*
- + *Ghi nhận, biến đổi dạng tin, phát tin cho thiết bị nhận tin.*
- + *Nhận và phát tín hiệu nhịp thời gian trao đổi tin cho các khối trong KGN và TBN.*

5.3.2. Sơ đồ khối.



Hình 5.2: Sơ đồ khối

a. Khối phối hợp đường dây MT.

Khối có nhiệm vụ:

- Phối hợp mức và công suất tín hiệu với bus I/O của MT.
- Cô lập bus I/O với các TBN khi không trao đổi tin.
- Điều khiển đưa tin ra, đưa tin vào bus I/O.

Các nhiệm vụ trên được thực hiện nhờ các vi mạch đệm ba trạng thái.

b. Khối giải mã địa chỉ - lệnh.

Mỗi thanh ghi đệm (điều khiển, trạng thái, số liệu đọc vào, số liệu đưa ra) của KGN được chọn để ghi và đọc tin nhờ các lệnh đọc, ghi từ khối giả mã địa chỉ - lệnh. Khối giải mã này là những vi mạch giải mã hay tổ hợp các công logic. Lối vào được nối với bus I/O của MT, để nhận các tín hiệu địa chỉ ($A_0 \dots A_n$), tín hiệu điều khiển đọc, ghi, các tín hiệu chốt địa chỉ, chốt dữ liệu. Lối ra của khối này là các tín hiệu đọc, ghi cho từng thanh ghi đệm của KGN.

c. Các thanh ghi đệm gồm:

- Thanh ghi điều khiển chế độ hoạt động, thanh ghi điều khiển TBN.
- Thanh ghi trạng thái hay yêu cầu trao đổi tin của TBN.
- Thanh ghi đệm số liệu ghi
- Thanh ghi đệm số liệu đọc.

d. Khối xử lý ngắt.

Khi nhận, che chắn yêu cầu trao đổi tin của TBN, xử lý ưu tiên và đưa yêu cầu trao đổi tin vào MT.

e. Khối phát nhịp thời gian.

Phát nhịp thời gian cho các hoạt động truyền và xử lý tin trong KGN hay TBN. Đôi khi, để đồng bộ, khối còn nhận tín hiệu nhịp đồng hồ từ MT.

g. Khối đệm TBN.

Khối có thể biến đổi mức (TTL), biến đổi công suất (cho các TBN là các mạch điều khiển công suất) và biến đổi về dạng tin.

h. Khối điều khiển:

Điều khiển hoạt động của các khối, như khối phát nhịp thời gian, chế độ hoạt động, vv... .

5.4. Giải mã địa chỉ cho bộ ghép nối.

Việc giải mã địa chỉ cho bộ ghép nối cũng gần giống như giải mã địa chỉ cho mạch nhớ. Chủ yếu ta nghiên cứu việc giải mã địa chỉ cho các cổng. Thông thường các cổng có địa chỉ 8 bit tại A0-A7 hoặc có địa chỉ 16 bit tại A0-A15. Tùy theo độ dài của toán hạng trong lệnh là 8 hay 16 bit ta sẽ có 1 cổng 8 bit hay 2 cổng 16 bit

có địa chỉ liên nhau để tạo nên từ với độ dài tương ứng. Trong thực tế ít có hệ sử dụng hết 256 cổng I/O khác nhau, nên ta chỉ xét ở đây các bộ giải mã địa chỉ 8 bit A0-A7 và mạch giải mã thông dụng như 74LS138 để tạo ra các xung chọn thiết bị.

TÀI LIỆU THAM KHẢO

1. *Kiến Trúc Máy Tính*; Nguyễn Đình Việt; Nhà xuất bản: Đại học quốc gia Hà Nội
2. *Giáo trình kiến trúc máy tính*; Msc Võ Văn Chín, Ths Nguyễn Hồng Vân, KS Phạm Hữu Tài; Khoa CNTT, Đại học Cần Thơ.
3. *William Stallings - Computer Organization and Architecture: Designing for Performance (8th edition)* .
4. *Andrew S. Tanenbaum - Structured Computer Organization -Fourth Edition.*
5. *Giáo trình kiến trúc máy tính- Võ Đức Khánh - ĐH Quốc Gia Tp.HCM*