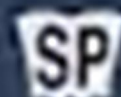


BỘ GIÁO DỤC VÀ ĐÀO TẠO
DỰ ÁN ĐÀO TẠO GIÁO VIÊN THCS
LOẠI NG 1718 - VTE (96)

PHẠO ĐỨC THÂN

CẤU TRÚC MÁY TÍNH PC



NHÀ XUẤT BẢN ĐẠI HỌC SƯ PHẠM

CHƯƠNG 1

KIẾN TRÚC CƠ BẢN CỦA MÁY TÍNH

§ 1. Những thành phần cơ bản của máy tính

Biểu diễn thông tin trong máy tính

I. Hệ đếm nhị phân và phương pháp biểu diễn thông tin trong máy tính.

1. Hệ nhị phân (Binary)

1.1. Khái niệm:

Hệ nhị phân hay hệ đếm cơ số 2 chỉ có hai con số 0 và 1. Đó là hệ đếm dựa theo vị trí. Giá trị của một số bất kỳ nào đó tùy thuộc vào vị trí của nó. Các vị trí có trọng số bằng bậc lũy thừa của cơ số 2. Chấm cơ số được gọi là chấm nhị phân trong hệ đếm cơ số 2. 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:

$$\begin{array}{ccccccc} 2^3 & 2^2 & 2^1 & 2^0 & 2^{-1} & 2^{-2} & \\ \text{MSB} & 1 & 0 & 1 & 0 & . & 1 & 1 & \text{LSB} \\ & & & & & & \text{Chấm nhị phân} \end{array}$$

Số nhị phân $(1010.11)_2$ có thể biểu diễn thành:

$$(1010.11)_2 = 1*2^3 + 0*2^2 + 1*2^1 + 0*2^0 + 1*2^{-1} + 1*2^{-2} = (10.75)_{10}.$$

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

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:

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

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

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

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

2. Hệ thập lục phân (Hexadecima).

2.1. 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 1.1 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

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

Người ta đã xây dựng bộ mã để biểu diễn cho các ký tự cũng như các con số Và các ký hiệu đặc biệt khác. 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: Đổi các ký tự BILL thành mã ASCII:

Ký tự	B	I	L	L
ASCII	1000010	1001001	1001100	1001100
HEXA	42	49	4C	4C

Bảng 1.2: Mã ASCII.

						Column bits(B ₇ B ₆ B ₅)							
Bits(row)						000	001	010	011	100	101	110	111
R	B ₄	B ₃	B ₂	B ₁		0	1	2	3	4	5	6	7
O						↓	↓	↓	↓	↓	↓	↓	↓
W													
0	0	0	0	0	→	NUL	DLE	SP	0	@	P	\	p
1	0	0	0	1	→	SOH	DC1	!	1	A	Q	A	q
2	0	0	1	0	→	STX	DC2	“	2	B	R	B	r
3	0	0	1	1	→	ETX	DC3	#	3	C	S	C	s
4	0	1	0	0	→	EOT	DC4	\$	4	D	T	D	t
5	0	1	0	1	→	ENQ	NAK	%	5	E	U	E	u
6	0	1	1	0	→	ACK	SYN	&	6	F	V	F	v
7	0	1	1	1	→	BEL	ETB	‘	7	G	W	G	w
8	1	0	0	0	→	BS	CAN	(8	H	X	H	x
9	1	0	0	1	→	HT	EM)	9	I	Y	I	y
A	1	0	1	0	→	LF	SUB	*	:	J	Z	J	z
B	1	0	1	1	→	VT	ESC	+	;	K	[K	{
C	1	1	0	0	→	FF	FS	-	<	L	\	L	
D	1	1	0	1	→	CR	GS	,	=	M]	M	}
E	1	1	1	0	→	SO	RS	.	>	N	^	N	~
F	1	1	1	1	→	SI	US	/	?	O	_	O	DEL

Control characters:

NUL = Null; DLE = Data link escape; SOH = Start Of Heading;

DC1 = Device control 1; DC2 = Device control 2; DC3 = Device control 3.

DC4 = Device control 4; STX = Start of text;

ETX = End of text;

EOT = End of transmission; ENQ = Enquiry;

NAK = Negative

acknowledge.

ACK = Acknowledge;

SYN = Synidle;

BEL = Bell.

ETB = End of transmission block; BS = Backspace; CAN = Cancel.

HT = Horizontal tab; EM = End of medium; LF = Line feed; SUB = Substitute.

VT = Vertical tab; ESC = Escape; FF = From feed; FS = File separator.

SO = Shift out; RS = Record separator; SI = Shift in; US = Unit separator.

4. Biểu diễn giá trị số trong máy tính.

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

4.2. Biểu diễn số thực(số có dấu chấm (phẩy) động).

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:

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

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

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:

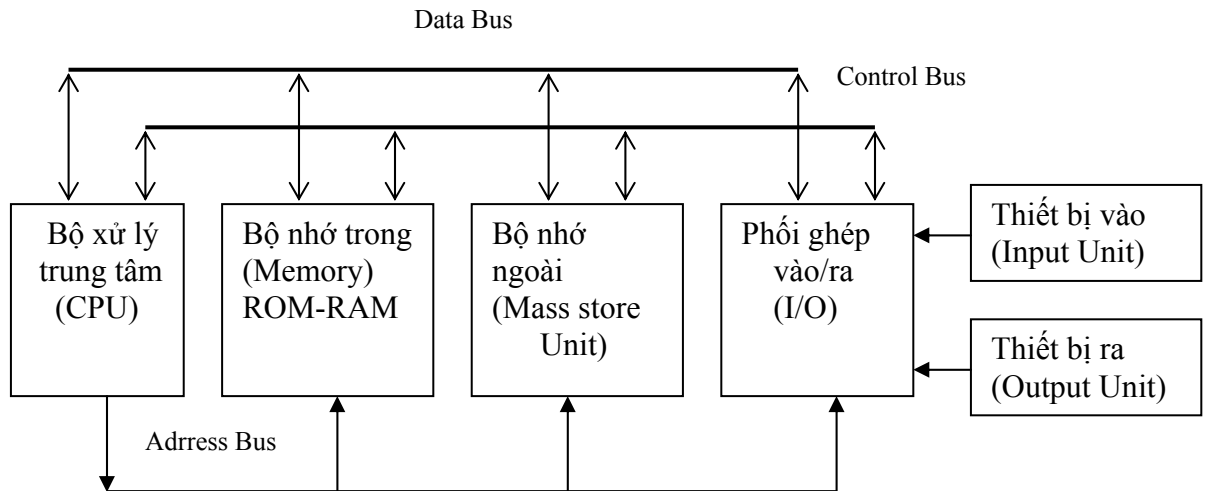
	<u>0100 0001 0001 1100 0000 0000 0000 0000</u>																															
bit	31 30				23 22																								0			

§ 2. Kiến trúc một máy tính đơn giản

2.1. Giới thiệu sơ lược cấu trúc của máy vi tính.

So với 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 2-1):

Hình 2-1: 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 ...

2.2 Lịch sử phát triển của CPU

2.2.1.-BXL 4 bit

4004 là BXL đầu tiên được Intel đưa ra tháng 11 năm 1971, có tốc độ 740KHz, khả năng xử lý 0,06 triệu lệnh mỗi giây (million instructions per second - MIPS); được sản xuất trên công nghệ 10 μ m, có 2.300 transistor (bóng bán dẫn), bộ nhớ mở rộng đến 640 byte.

2.2.2 BXL 8bit

8008 (năm 1972) được sử dụng trong thiết bị đầu cuối Datapoint 2200 của Computer Terminal Corporation (CTC). 8008 có tốc độ 200kHz, sản xuất trên công nghệ 10 μ m, với 3.500 transistor, bộ nhớ mở rộng đến

16KB. 8080 (năm 1974) sử dụng trong máy tính Altair 8800, có tốc độ gấp 10 lần 8008 (2MHz), sản xuất trên công nghệ 6 μm , khả năng xử lý 0,64 MIPS với 6.000 transistor, có 8 bit bus dữ liệu và 16 bit bus địa chỉ, bộ nhớ mở rộng tới 64KB. 8085 có tốc độ 2MHz, sản xuất trên công nghệ 3 μm , với 6.500 transistor, có 8 bit bus dữ liệu và 16 bit bus địa chỉ, bộ nhớ mở rộng 64KB.

2.2.3.-BXL 16bit

80186 (năm 1982) còn gọi là IAPX 186. Sử dụng chủ yếu trong những ứng dụng nhúng, bộ điều khiển thiết bị đầu cuối. Các phiên bản của 80186 gồm 10 và 12 MHz. 80286 (năm 1982) sử dụng công nghệ 1,5 μm , 134.000 transistor, bộ nhớ mở rộng tới 16 MB. Các phiên bản của 286 gồm 6, 8, 10, 12,5, 16, 20 và 25MHz.

2.2.4. BXL 32bit vi kiến trúc NetBurst (NetBurst micro-architecture)

Intel386 gồm các họ 386DX, 386SX và 386SL. Intel386DX là BXL 32 bit đầu tiên Intel giới thiệu vào năm 1985, 386 sử dụng các thanh ghi 32 bit, có thể truyền 32 bit dữ liệu cùng lúc trên bus dữ liệu và dùng 32 bit để xác định địa chỉ. Cũng như BXL 80286, 80386 hoạt động ở 2 chế độ: real mode và protect mode.

386SL (năm1990) được thiết kế cho thiết bị di động, sử dụng công nghệ 1 μm , 855.000 transistor, bộ nhớ mở rộng 4GB; gồm các phiên bản 16, 20, 25 MHz. 486DX sử dụng công nghệ 1 μm , 1,2 triệu transistor, bộ nhớ mở rộng 4GB; gồm các phiên bản 25 MHz, 33 MHz và 50 MHz (0,8 μm). Pentium sử dụng công nghệ 0,8 μm chứa 3,1 triệu transistor, có các tốc độ 60, 66 MHz (socket 4 273 chân, PGA). Các phiên bản 75, 90, 100, 120 MHz sử dụng công nghệ 0,6 μm chứa 3,3 triệu transistor (socket 7, PGA). Phiên bản 133, 150, 166, 200 sử dụng công nghệ 0,35 μm chứa 3,3 triệu transistor (socket 7, PGA). Pentium MMX sử dụng công nghệ 0,35 μm chứa 4,5 triệu transistor, có các tốc độ 166, 200, 233 MHz (Socket 7, PGA).

2.2.5. Pentium Pro:

Nối tiếp sự thành công của dòng Pentium, Pentium Pro được Intel giới thiệu vào tháng 9 năm 1995, sử dụng công nghệ 0,6 và 0,35 μm chứa 5,5 triệu transistor, socket 8 387 chân, Dual SPGA, hỗ trợ bộ nhớ RAM tối đa 4GB.

2.2.6. BXL Pentium II

Đầu tiên, tên mã Klamath, sản xuất trên công nghệ 0,35 μm , có 7,5 triệu transistor, bus hệ thống 66 MHz, gồm các phiên bản 233, 266, 300MHz. Pentium II, tên mã Deschutes, sử dụng công nghệ 0,25 μm , 7,5 triệu transistor, gồm các phiên bản 333MHz (bus hệ thống 66MHz), 350, 400, 450 MHz (bus hệ thống 100MHz). Celeron (năm 1998) được “rút gọn” từ kiến trúc BXL Pentium II, dành cho dòng máy cấp thấp.

2.2.7. Pentium III (năm 1999)

Bổ sung 70 lệnh mới (Streaming SIMD Extensions - SSE) giúp tăng hiệu suất hoạt động của BXL trong các tác vụ xử lý hình ảnh, audio, video và nhận dạng giọng nói. Pentium III gồm các tên mã Katmai, Coppermine và Tualatin. Coppermine sử dụng công nghệ 0,18 μm , 28,1 triệu transistor, bộ nhớ đệm L2 256 KB được tích hợp bên trong nhằm tăng tốc độ xử lý. Tualatin áp dụng công nghệ 0,13 μm có 28,1 triệu transistor, bộ nhớ đệm L1 32KB, L2 256 KB hoặc 512 KB tích hợp bên trong BXL, socket 370 FC-PGA (Flip-chip pin grid array), bus hệ thống 133 MHz. Có các tốc độ như 1133, 1200, 1266, 1333, 1400 MHz. Celeron Coppermine (năm 2000) được “rút gọn” từ kiến trúc BXL Pentium III Coppermine, còn gọi là Celeron II, được bổ sung 70 lệnh SSE. Sử dụng công nghệ 0,18 μm có 28,1 triệu transistor, bộ nhớ đệm L1 32KB, L2 256 KB tích hợp bên trong BXL, socket 370 FC-PGA, Có các tốc độ như 533, 566, 600, 633, 667, 700, 733, 766, 800 MHz (bus 66 MHz), 850, 900, 950, 1000, 1100, 1200, 1300 MHz (bus 100 MHz). Tualatin Celeron (Celeron S) (năm 2000) được “rút gọn” từ kiến trúc BXL Pentium III Tualatin, áp dụng công nghệ 0,13 μm , bộ nhớ đệm L1 32KB, L2 256 KB tích hợp, socket 370 FC-PGA, bus hệ thống 100 MHz, gồm các tốc độ 1,0, 1,1, 1,2, 1,3 và 1,4 GHz.

2.2.8. Pentium 4

Intel Pentium 4 (P4) là BXL thế hệ thứ 7 dòng x86 phổ thông, được giới thiệu vào tháng 11 năm 2000. P4 sử dụng vi kiến trúc NetBurst có thiết kế hoàn toàn mới so với các BXL cũ (PII, PIII và Celeron sử dụng vi kiến trúc P6). Một số công nghệ nổi bật được áp dụng trong vi kiến trúc NetBurst như Hyper Pipelined Technology mở rộng số hàng lệnh xử lý, Execution Trace Cache tránh tình trạng lệnh bị chậm trễ khi chuyển từ bộ nhớ đến CPU, Rapid Execution Engine tăng tốc bộ đồng xử lý toán học, bus hệ thống (system bus) 400 MHz và 533 MHz; các công nghệ Advanced Transfer Cache, Advanced Dynamic Execution, Enhanced

Floating point và Multimedia Unit, Streaming SIMD Extensions 2 (SSE2) cũng được cải tiến nhằm tạo ra những BXL tốc độ cao hơn, khả năng tính toán mạnh hơn, xử lý đa phương tiện tốt hơn. Pentium 4 đầu tiên (tên mã Willamette) xuất hiện cuối năm 2000 đặt dấu chấm hết cho "triều đại" Pentium III. Willamette sản xuất trên công nghệ 0,18 μm , có 42 triệu transistor (nhiều hơn gần 50% so với Pentium III), bus hệ thống (system bus) 400 MHz, bộ nhớ đệm tích hợp L2 256 KB, socket 423 và 478. P4 Willamette có một số tốc độ như 1,3, 1,4, 1,5, 1,6, 1,7, 1,8, 1,9, 2,0 GHz. P4 Northwood. Xuất hiện vào tháng 1 năm 2002, được sản xuất trên công nghệ 0,13 μm , có khoảng 55 triệu transistor, bộ nhớ đệm tích hợp L2 512 KB, socket 478. Northwood có 3 dòng gồm Northwood A (system bus 400 MHz), tốc độ 1,6, 1,8, 2,0, 2,2, 2,4, 2,5, 2,6 và 2,8 GHz. Northwood B (system bus 533 MHz), tốc độ 2,26, 2,4, 2,53, 2,66, 2,8 và 3,06 GHz (riêng 3,06 GHz có hỗ trợ công nghệ siêu phân luồng Hyper Threading - HT). Northwood C (system bus 800 MHz, tất cả hỗ trợ HT), gồm 2,4, 2,6, 2,8, 3,0, 3,2, 3,4 GHz. P4 Prescott (năm 2004). Là BXL đầu tiên Intel sản xuất theo công nghệ 90 nm, kích thước vi mạch giảm 50% so với P4 Willamette. Điều này cho phép tích hợp nhiều transistor hơn trên cùng kích thước (125 triệu transistor so với 55 triệu transistor của P4 Northwood), tốc độ chuyển đổi của transistor nhanh hơn, tăng khả năng xử lý, tính toán. Dung lượng bộ nhớ đệm tích hợp L2 của P4 Prescott gấp đôi so với P4 Northwood (1MB so với 512 KB). Ngoài tập lệnh MMX, SSE, SSE2, Prescott được bổ sung tập lệnh SSE3 giúp các ứng dụng xử lý video và game chạy nhanh hơn. Đây là giai đoạn "giao thời" giữa socket 478 - 775LGA, system bus 533 MHz - 800 MHz và mỗi sản phẩm được đặt tên riêng khiến người dùng càng bối rối khi chọn mua. Prescott A (FSB 533 MHz) có các tốc độ 2,26, 2,4, 2,66, 2,8 (socket 478), Prescott 505 (2,66 GHz), 505J (2,66 GHz), 506 (2,66 GHz), 511 (2,8 GHz), 515 (2,93 GHz), 515J (2,93 GHz), 516 (2,93 GHz), 519J (3,06 GHz), 519K (3,06 GHz) sử dụng socket 775LGA. Prescott E, F (năm 2004) có bộ nhớ đệm L2 1 MB (các phiên bản sau được mở rộng 2 MB), bus hệ thống 800 MHz. Ngoài tập lệnh MMX, SSE, SSE2, SSE3 tích hợp, Prescott E, F còn hỗ trợ công nghệ siêu phân luồng, một số phiên bản sau có hỗ trợ tính toán 64 bit.

Dòng sử dụng socket 478 gồm Pentium 4 HT 2.8E (2,8 GHz), 3.0E (3,0 GHz), 3.2E (3,2 GHz), 3.4E (3,4 GHz). Dòng sử dụng socket 775LGA gồm Pentium 4 HT 3.2F, 3.4F, 3.6F, 3.8F với các tốc độ tương ứng từ 3,2 GHz đến 3,8 GHz, Pentium 4 HT 517, 520, 520J, 521, 524, 530, 530J, 531, 540, 540J, 541, 550, 550J, 551, 560, 560J, 561, 570J, 571 với các tốc độ từ 2,8 GHz đến 3,8 GHz.

2.2.9. BXL Celeron

BXL Celeron được thiết kế với mục tiêu dung hòa giữa công nghệ và giá cả, đáp ứng các yêu cầu phổ thông như truy cập Internet, Email, chat, xử lý các ứng dụng văn phòng. Celeron Willamette 128 (2002), bản "rút gọn" từ P4 Willamette, sản xuất trên công nghệ 0,18 μm , bộ nhớ đệm L2 128 KB, bus hệ thống 400 MHz, socket 478. Celeron Willamette 128 hỗ trợ tập lệnh MMX, SSE, SSE2. Một số BXL thuộc dòng này như Celeron 1.7 (1,7 GHz) và Celeron 1.8 (1,8 GHz). Celeron NorthWood 128, "rút gọn" từ P4 Northwood, công nghệ 0,13 μm , bộ nhớ đệm tích hợp L2 128 KB, bus hệ thống 400 MHz, socket 478. Celeron NorthWood 128 cũng hỗ trợ các tập lệnh MMX, SSE, SSE2, gồm Celeron 1.8A, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8 tương ứng với các tốc độ từ 1,8 GHz đến 2,8 GHz. Celeron D (Prescott 256), được xây dựng từ nền tảng P4 Prescott, sản xuất trên công nghệ 90nm, bộ nhớ đệm tích hợp L2 256 KB (gấp đôi dòng Celeron NorthWood), bus hệ thống 533 MHz, socket 478 và 775LGA. Ngoài các tập lệnh MMX, SSE, SSE2, Celeron D hỗ trợ tập lệnh SSE3, một số phiên bản sau có hỗ trợ tính toán 64 bit. Celeron D gồm 310, 315, 320, 325, 325J, 326, 330, 330J, 331, 335, 335J, 336, 340, 340J, 341, 345, 345J, 346, 350, 351, 355 với các tốc độ tương ứng từ 2,13 GHz đến 3,33 GHz.

2.2.10. Pentium 4 Extreme Edition

Pentium 4 Extreme Edition (P4EE) xuất hiện vào tháng 9 năm 2003, là BXL được Intel "ưu ái" dành cho game thủ và người dùng cao cấp. P4EE được xây dựng từ BXL Xeon dành cho máy chủ và trạm làm việc. Ngoài công nghệ HT "đỉnh đám" thời bấy giờ, điểm nổi bật của P4EE là bổ sung bộ nhớ đệm L3 2 MB. Phiên bản đầu tiên của P4 EE (nhân Gallatin) sản xuất trên công nghệ 0,13 μm , bộ nhớ đệm L2 512 KB, L3 2 MB, bus hệ thống 800 MHz, sử dụng socket 478 và 775LGA, gồm P4 EE 3.2 (3,2 GHz), P4 EE 3.4 (3,4 GHz).

2.2.11. BXL 64 bit, vi kiến trúc NETBURST

P4 Prescott (năm 2004) Vi kiến trúc NetBurst 64 bit (Extended Memory 64 Technology - EM64T) đầu tiên được Intel sử dụng trong BXL P4 Prescott (tên mã Prescott 2M). Prescott 2M cũng sử dụng công nghệ 90 nm, bộ nhớ đệm L2 2 MB, bus hệ thống 800 MHz, socket 775LGA. Ngoài các tập lệnh MMX, SSE, SSE2, SSE3, công nghệ HT và khả năng tính toán 64 bit, Prescott 2M (trừ BXL 620) có hỗ trợ công nghệ Enhanced SpeedStep để tối ưu tốc độ làm việc nhằm tiết kiệm điện năng.

Các BXL 6x2 có thêm công nghệ ảo hóa (Virtualization Technology). Prescott 2M có một số tốc độ như P4 HT 620 (2,8 GHz), 630 (3,0 GHz), 640 (3,2 GHz), 650 (3,4 GHz), 660, 662 (3,6 GHz) và 670, 672 (3,8 GHz).

2.2.12. Pentium D (năm 2005)

Pentium D (tên mã Smithfield, 8xx) là BXL lõi kép (dual core) đầu tiên của Intel, được cải tiến từ P4 Prescott nên cũng gặp một số hạn chế như hiện tượng thất cổ chai do băng thông BXL ở mức 800 MHz (400 MHz cho mỗi lõi), Cùng sử dụng vi kiến trúc NetBurst, Pentium D (mã Presler, 9xx) được Intel thiết kế mới trên công nghệ 65nm, 376 triệu transistor, bộ nhớ đệm L2 4 MB (2x2 MB), hiệu năng cao hơn, nhiều tính năng mới và ít tốn điện năng hơn Smithfield. Pentium D 915 và 920 tốc độ 2,8 GHz, 925 và 930 (3,0GHz), 935 và 940 (3,2 GHz), 945 và 950 (3,4 GHz), 960 (3,6GHz). Presler dòng 9x0 có hỗ trợ Virtualization Technology.

2.2.13. Pentium Extreme Edition (năm 2005)

BXL lõi kép dành cho game thủ và người dùng cao cấp. Pentium EE sử dụng nhân Smithfield, Presler của Pentium D trong đó Smithfield sử dụng công nghệ 90nm, bộ nhớ đệm L2 được mở rộng đến 2 MB (2x1 MB), hỗ trợ tập lệnh MMX, SSE, SSE2, SSE3, công nghệ HT, Enhanced Intel SpeedStep Technology (EIST) và EM64T. Pentium 840 EE (3,20 GHz, bus hệ thống 800 MHz, socket 775LGA) là một trong những BXL thuộc dòng này.

2.2.14. BXL 64bit, kiến trúc Core

Tại diễn đàn IDF đầu năm 2006, Intel đã giới thiệu kiến trúc Intel Core với năm cải tiến quan trọng là khả năng mở rộng thực thi động (Wide Dynamic Execution), tính năng quản lý điện năng thông minh (Intelligent Power Capability), chia sẻ bộ nhớ đệm linh hoạt (Advanced Smart Cache), truy xuất bộ nhớ thông minh (Smart Memory Access) và tăng tốc phương tiện số tiên tiến (Advanced Digital Media Boost).

2.2.15. Intel Core 2 Duo

BXL lõi kép sản xuất trên công nghệ 65 nm, hỗ trợ SIMD instructions, công nghệ Virtualization Technology cho phép chạy cùng lúc nhiều HĐH, tăng cường bảo vệ hệ thống trước sự tấn công của virus (Execute Disable Bit), tối ưu tốc độ BXL nhằm tiết kiệm điện năng (Enhanced

Intel SpeedStep Technology), quản lý máy tính từ xa (Intel Active Management Technology). Ngoài ra, còn hỗ trợ các tập lệnh MMX, SSE, SSE2, SSE3, SSSE3.

Core 2 Duo (tên mã Conroe) có 291 triệu transistor, bộ nhớ đệm L2 4 MB, bus hệ thống 1066 MHz, socket 775LGA. Một số BXL thuộc dòng này: E6600 (2,4 GHz), E6700 (2,66 GHz). Core 2 Duo (tên mã Allendale) E6300 (1,86 GHz), E6400 (2,13 GHz) có 167 triệu transistor, bộ nhớ đệm L2 2MB, bus hệ thống 1066 MHz, socket 775LGA. E4300 (1,8 GHz) xuất hiện năm 2007 có bộ nhớ đệm L2 2 MB, bus 800 MHz, không hỗ trợ Virtualization Technology.

2.2.16. Core 2 Extreme

BXL lõi kép dành cho game thủ sử dụng kiến trúc Core, có nhiều đặc điểm giống với BXL Core 2 như công nghệ sản xuất 65 nm, hỗ trợ các công nghệ mới Enhanced Intel SpeedStep Technology, Intel x86-64, Execute Disable Bit, Intel Active Management, Virtualization Technology, Intel Trusted Execution Technology... các tập lệnh MMX, SSE, SSE2, SSE3, SSSE3.

2.2.17. Core 2 Extreme

Core 2 Extreme (tên mã Conroe XE) (tháng 7 năm 2006) với đại diện X6800 2,93 Ghz, bộ nhớ đệm L2 đến 4 MB, bus hệ thống 1066 MHz, socket 775LGA. Cuối năm 2006, con đường phía trước của BXL tiếp tục rộng mở khi Intel giới thiệu BXL 4 nhân (Quad Core) như Core 2 Extreme QX6700, Core 2 Quad Q6300, Q6400, Q6600 và BXL 8 nhân trong vài năm tới. Chắc chắn những BXL này sẽ thỏa mãn nhu cầu người dùng đam mê công nghệ và tốc độ.

Hiện đã có loại **CPU Quad-Core** (4 nhân). Hãng AMD đã cho ra công nghệ gồm 2 bộ xử lý, mỗi bộ 2-4 nhân. Tuy nhiên loại CPU này vẫn chưa có mặt trên thị trường.

2.3 Chất liệu và công nghệ chế tạo CPU

2.3.1. Chất liệu

Gốm và organic (hữu cơ) từ dòng Thoroughbred trở đi đều làm bằng organic. Hiện tại, công nghệ được áp dụng cho các CPU Chất liệu chủ yếu chế tạo cpu AMD là ceramic à MOS (Metal Oxide Semi-Conductor - bán dẫn ôxít kim loại), dựa vào một lớp ôxít kim loại nằm trên tấm

silicon kết nối bởi các đường hợp chất dẫn điện. Người ta đã cải tiến MOS thành CMOS (Complimentary MOS - MOS bổ trợ) hoạt động ở điện thế thấp. Đây là 2 công nghệ có mặt trong hầu hết các thiết bị máy tính. Để đáp ứng nhu cầu làm cho CPU ngày càng nhanh hơn, ít tiêu hao năng lượng hơn các công nghệ 0,25 \rightarrow 0,18 \rightarrow 0,13 micron lần lượt ra đời. Nhưng chính sự thu nhỏ các cầu nối trong CPU này khiến việc áp dụng MOS và CMOS trở nên ngày càng khó khăn hơn, do các cầu nối này nằm quá sát nhau nên dễ dẫn đến hiện tượng đóng điện chéo lên các cầu bên cạnh. Một nhược điểm quan trọng khác của công nghệ MOS là phần silicon ở giữa các cầu nối (có vai trò như một tụ điện) phải nạp được điện dung tối đa để có thể đóng - và lại phải thoát hết điện dung để có thể mở. Việc này tốn thời gian xử lý, và lãng phí thời gian xử lý trên CPU. Các nhà sản xuất CPU đã cải tiến MOS hiện có như việc thay oxit nhôm bằng oxit đồng làm tăng xung nhịp lên đáng kể. Nhưng để CPU có thể đạt tới tốc độ 5-10 GHz phải có một giải pháp khắc phục triệt để hơn nữa 2 nhược điểm nêu trên. Đó chính là công nghệ SOI (Silicon On Insulator). IBM đã phát triển công nghệ này từ năm 1990 cho CPU của IBM, với mục đích giảm điện năng sử dụng, tăng xung nhịp v.v...nhưng công nghệ này vẫn chưa thực sự được ứng dụng ngay cho đến cuối thế kỉ 20, khi việc tăng xung nhịp cho các dòng CPU hiện đại cần thêm các phương pháp sản xuất khác. Cải tiến SOI là điện dung của tụ silicon giữa các cầu được cực tiểu hoá làm giảm thời gian cần thiết để thoát/nạp, để mở và đóng cầu nối. Điều này giúp tăng xung nhịp lên rất nhiều. Sở dĩ SOI làm được điều đó là nhờ việc chèn vào giữa tấm silicon một lớp vật liệu cách điện và để lại một phần silicon nhỏ ở giữa các cầu nối. Lớp vật liệu cách điện này là một dạng của ôxít silicon được tạo ra bằng kỹ thuật SIMOX (Seperation by Implantation of Oxygen) - khí ôxi được ép lên bề mặt của silicon wafer ở áp suất và nhiệt độ cao, khi đó silicon phản ứng với ôxi tạo nên 1 lớp ôxít silicon bám vào silicon wafer bên dưới. SOI sẽ không thay thế hoàn toàn MOS/CMOS mà chỉ tối ưu hoá cho hai công nghệ này:

- CPU dùng SOI sẽ nhanh hơn đến 30% so với CPU dùng MOS/CMOS nếu có cùng một xung đồng hồ như nhau.
- Yêu cầu về điện năng thấp hơn nhiều so với MOS/CMOS (ít hơn khoảng 50%), CPU sẽ chạy mát hơn - vượt qua một trở ngại lớn của việc nâng tốc độ các bộ xử lý.
- Cho phép thu nhỏ công nghệ sản xuất CPU xuống 0.09 micron hay thấp hơn cùng với SOI có nghĩa rằng các bộ vi xử lý sẽ được tăng tốc rất nhanh và tốc độ 5-10GHz sẽ sớm đạt được. Thế nhưng SOI cần có silicon đạt độ nguyên chất 100% - thứ mà công nghệ hiện nay chưa sản xuất được. Isonics là 1 công ty đang nghiên cứu sản xuất loại silicon wafer

này. AMD thực sự trông đợi vào SOI để khắc phục những nhược điểm của CPU như tiêu tốn nhiều điện năng và chạy nóng hơn. bộ xử lý K8 của IBM, hay còn gọi là Hammer dùng công nghệ SOI đang được mong đợi. Nội lực công nghệ - HyperTransport, Cool'n'Quiet. AMD đặc biệt ưu ái CPU 64 bit với công nghệ 'siêu chuyển' HyperTransport và tự điều chỉnh hoạt động Cool'n'Quiet. HyperTransport giúp việc truyền thông tin giữa các chip (cầu nam, cầu bắc, BXL, bộ nhớ,...) nhanh hơn, khả năng 'nói chuyện' với một chip hoặc thiết bị khác nhanh hơn với lượng tiêu thụ lớn hơn. HyperTransport làm cho đường truyền rộng hơn, do đó tốc độ truyền nhanh và nhiều hơn. Công nghệ này có thể áp dụng cho tất cả băng thông của bo mạch chủ, từ chipset đến BXL, bộ nhớ, AGP, PCI,...Cool'n'Quiet là một cải tiến khác dành cho dòng BXL 64 bit, tốc độ và điện năng tiêu thụ của BXL sẽ được điều chỉnh tự động. Nếu có ít ứng dụng được chạy (BXL xử lý ít) thì Cool'n'Quiet sẽ giảm tốc độ và điện thế BXL, ngược lại, khi cần xử lý nhiều thì BXL sẽ được tăng tốc độ và điện thế.

2.4 Nguyên tắc hoạt động của CPU

CPU (Central Processing Unit) – cũng được gọi là microprocessor hay processor – là một đơn vị xử lý dữ liệu trung tâm. Cách nó xử lý dữ liệu như thế nào hoàn toàn phụ thuộc vào chương trình được viết từ trước. Chương trình nói chung có thể là một bảng tính, một bộ xử lý từ hay một game nào đó. Nó chỉ tuân theo các thứ tự (được gọi là các chỉ lệnh hay các lệnh) có bên trong chương trình.

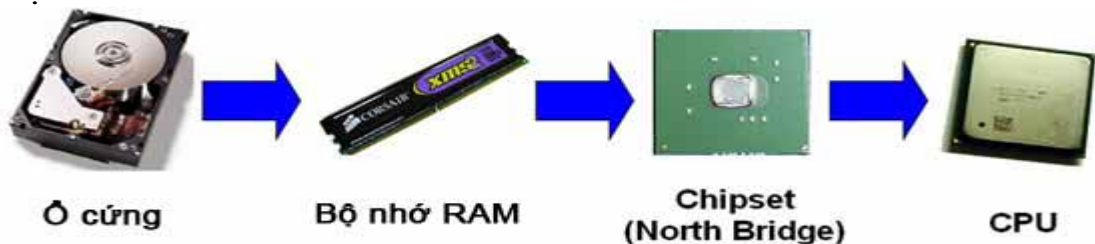
Khi một chương trình nào đó được chạy thì thứ tự được thực hiện như sau:

a. Chương trình đã lưu bên trong ổ đĩa cứng sẽ được đưa vào bộ nhớ RAM. Ở đây chương trình chính là một loạt các chỉ lệnh đối với CPU.

b. CPU sử dụng mạch phân cứng được gọi là memory controller để tải dữ liệu chương trình từ bộ nhớ RAM.

c. Lúc đó dữ liệu bên trong CPU sẽ được xử lý.

d. Những gì diễn ra tiếp theo sẽ phụ thuộc vào chương trình vừa được nạp. CPU có thể tiếp tục tải và thực thi chương trình hoặc có thể thực hiện một công việc nào đó với dữ liệu đã được xử lý, như việc hiển thị kết quả thực hiện nào đó lên màn hình.

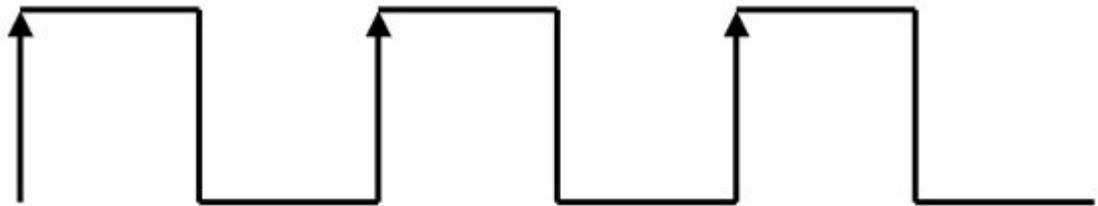


Hình 2.4: Dữ liệu lưu được đưa vào CPU

Sự truyền tải dữ liệu giữa ổ đĩa cứng và bộ nhớ RAM được thực hiện mà không sử dụng đến CPU, như vậy nó sẽ làm cho hệ thống hoạt động nhanh hơn. Phương pháp này được gọi là bus mastering hay DMA (Direct Memory Access). Các bộ vi xử lý của AMD dựa trên sockets 754, 939 và 940 (Athlon 64, Athlon 64 X2, Athlon 64 FX, Opteron và một số mô hình Sempron) có một memory controller được nhúng bên trong. Điều đó có nghĩa rằng với các bộ vi xử lý này, CPU truy cập trực tiếp bộ nhớ RAM.

2.4.1.Clock

Clock chính là một tín hiệu được sử dụng để đồng bộ hóa mọi thứ bên trong máy tính. Hãy xem trong hình 2.4.1, đây chính là một xung clock điển hình: nó là một xung hình vuông biến thiên ở mức “0” và “1” với một tốc độ được cố định. Trên hình vẽ ta có thể thấy 3 chu kỳ của xung clock này. Bắt đầu của mỗi một chu kỳ khi tín hiệu clock biến thiên từ “0” lên “1”; chúng được đánh dấu nó bằng một mũi tên. Tín hiệu clock được đo theo đơn vị có tên gọi là Hertz (Hz), đây là số chu kỳ clock trong mỗi giây đồng hồ. Một xung clock 100MHz có nghĩa là trong một giây đồng hồ có 100 triệu chu kỳ xung nhịp.



Hình 2: Tín hiệu xung clock

Trong máy tính, tất cả các bộ định thời đều được đo dưới dạng các chu kỳ clock. Ví dụ, một bộ nhớ RAM có độ trễ là “5” thì điều đó có nghĩa là nó sẽ giữ chậm 5 chu kỳ xung nhịp để thực hiện công việc cung cấp dữ liệu. Trong CPU, tất cả các chỉ lệnh giữ chậm một số chu kỳ xung clock nào đó để được thực thi. Ví dụ, một chỉ lệnh nào đó có thể được giữ chậm đến 7 chu kỳ xung clock để được thực thi xong.

CPU biết được bao nhiêu chu kỳ xung clock mà mỗi chỉ lệnh cần, nó biết được điều này bởi CPU giữ một bảng liệt kê các thông tin này. Chính vì vậy nếu CPU có hai chỉ lệnh được thực thi và nó biết rằng chỉ lệnh đầu tiên sẽ giữ chậm 7 chu kỳ xung clock để thực thi thì nó sẽ tự động thực thi chỉ lệnh kế tiếp vào chu kỳ clock thứ 8. Rõ ràng đây là một cách lý giải chung cho CPU với một khối thực thi – các bộ vi xử lý hiện đại có một số khối thực thi làm việc song song và nó có thể thực thi chỉ lệnh thứ hai tại cùng thời điểm với chỉ lệnh đầu. Điều này được gọi là kiến trúc “superscalar”.

Nếu so sánh hai CPU giống nhau, CPU nào chạy ở tốc độ clock cao hơn sẽ

nhanh hơn. Trong trường hợp này, với một tốc độ clock cao hơn, thời gian giữa mỗi chu kỳ clock sẽ ngắn hơn, vì vậy những công việc sẽ được thực thi tốn ít thời gian hơn và hiệu suất sẽ cao hơn. Tuy nhiên khi so sánh hai bộ vi xử lý khác nhau thì điều này hoàn toàn không đúng.

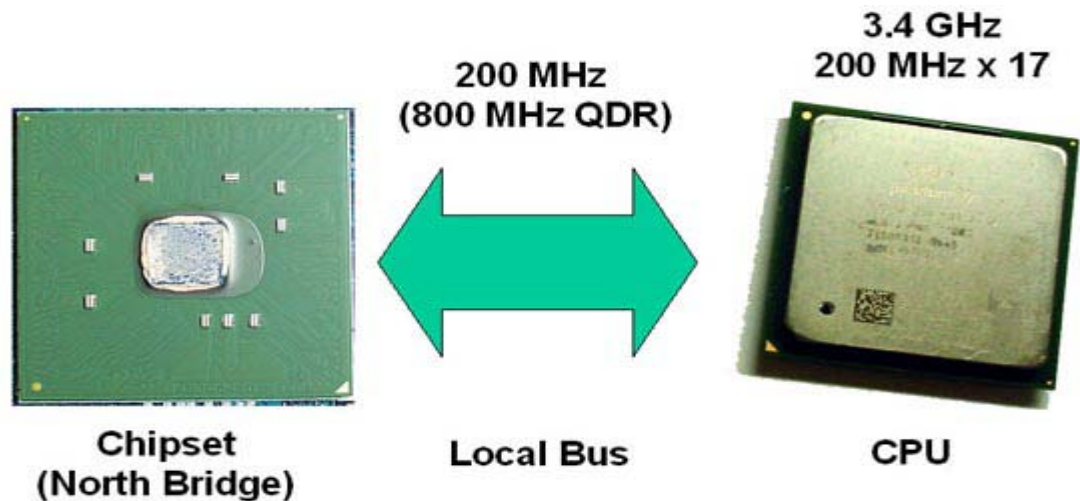
Nếu ta lấy hai bộ vi xử lý có kiến trúc khác nhau – ví dụ, khác nhau về nhà sản xuất như Intel và AMD – những thứ bên trong hai CPU này là hoàn toàn khác nhau. Như đã đề cập, mỗi chỉ lệnh cần đến một số chu kỳ clock nhất định để được thực thi. Chúng ta hãy nói rằng bộ vi xử lý “A” cần đến 7 chu kỳ clock để thực thi một chỉ lệnh nào đó và bộ vi xử lý “B” cần 5 chu kỳ clock để thực hiện một chỉ lệnh tương tự. Nếu chúng đang chạy với cùng một tốc độ clock thì bộ vi xử lý “B” sẽ nhanh hơn, vì nó có thể xử lý chỉ lệnh này tốn ít thời gian hơn. Với các CPU hiện đại, có nhiều vấn đề cần phải xem xét đến hiệu suất này, vì các CPU có số lượng khối thực thi khác nhau, kích thước cache khác nhau, các cách truyền tải dữ liệu bên trong CPU cũng khác nhau, cách xử lý các chỉ lệnh bên trong các khối thực thi và tốc độ clock khác nhau với thế giới thực bên ngoài,...

Khi tín hiệu clock của bộ vi xử lý cao thì có một vấn đề mà chúng ta gặp phải. Bo mạch chủ, nơi mà bộ vi xử lý được cài đặt không thể làm việc bằng cách sử dụng cùng tín hiệu clock. Nếu xem bo mạch chủ, ta sẽ thấy một số đường và rãnh. Các đường và rãnh này là những mạch in nối một số mạch của máy tính. Vấn đề ở đây là với tốc độ clock cao, các dây mạch in này sẽ bắt đầu làm việc như anten, chính vì vậy tín hiệu, thay vì đến vị trí cần đến ở phía cuối đầu dây lại biến mất, được truyền đi như các sóng vô tuyến.

2.4.2 External Clock

Vì vậy các nhà sản xuất CPU đã bắt đầu sử dụng một khái niệm mới, khái niệm được gọi là nhân xung clock, ứng dụng này bắt đầu được sử dụng trong bộ vi xử lý 486DX2. Với cơ chế này (được sử dụng trong tất cả các CPU ngày nay), CPU có một clock ngoài (external clock) được sử dụng khi truyền tải dữ liệu vào ra bộ nhớ RAM (sử dụng north bridge chip) và một clock trong cao hơn.

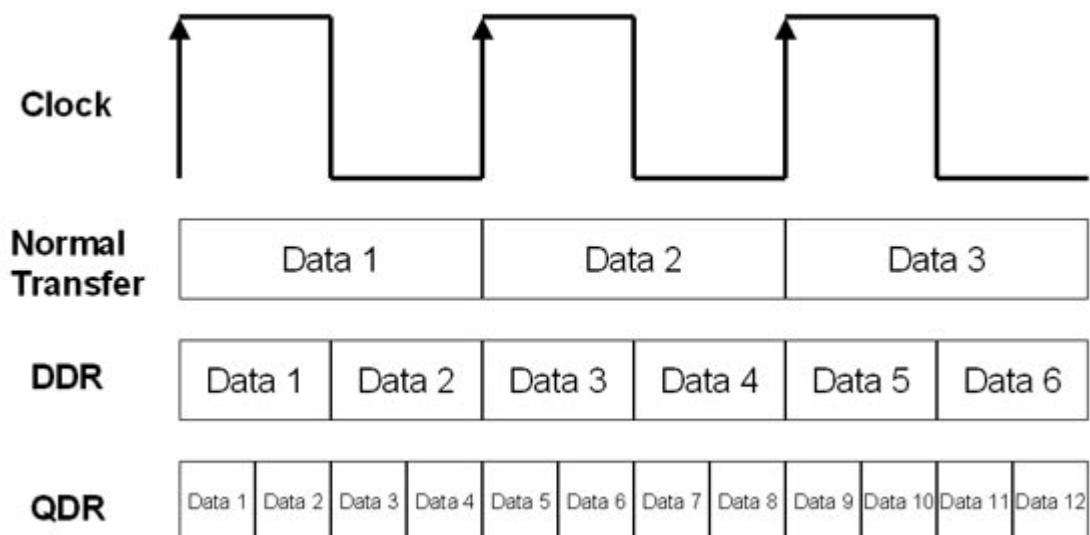
Để đưa ra một ví dụ thực, trong số 3.4 GHz Pentium 4 thì con số “3.4 GHz” chính là clock trong của CPU, clock này đạt được bằng cách nhân 17 với clock ngoài là 200 của nó. Mô phỏng ví dụ này trong hình 2.4.2



Hình 2.4.2: Clock trong và ngoài trên Pentium 4 3.4 GHz.

Sự khác nhau lớn giữa clock trong và clock ngoài trên các CPU hiện đại là cách vượt qua nhược điểm từ tính như đã nói trên để tăng hiệu suất máy tính. Tiếp tục với ví dụ về Pentium 4 3.4 GHz ở trên, nó phải giảm tốc độ của nó đi 17 lần khi thực hiện đọc dữ liệu từ bộ nhớ RAM! Trong suốt quá trình này, nó làm việc như một CPU với tốc độ 200MHz.

Một số kỹ thuật được sử dụng để tối thiểu hóa ảnh hưởng của sự khác nhau clock này. Một trong số chúng là sử dụng cache nhớ bên trong CPU. Phương pháp khác là truyền tải nhiều khối dữ liệu trên mỗi một chu kỳ clock. Các bộ vi xử lý của hai hãng Intel và AMD đều sử dụng tính năng này, tuy nhiên trong khi CPU của AMD truyền tải hai dữ liệu trên một chu kỳ clock thì các CPU của Intel truyền tải 4 dữ liệu trên mỗi chu kỳ.



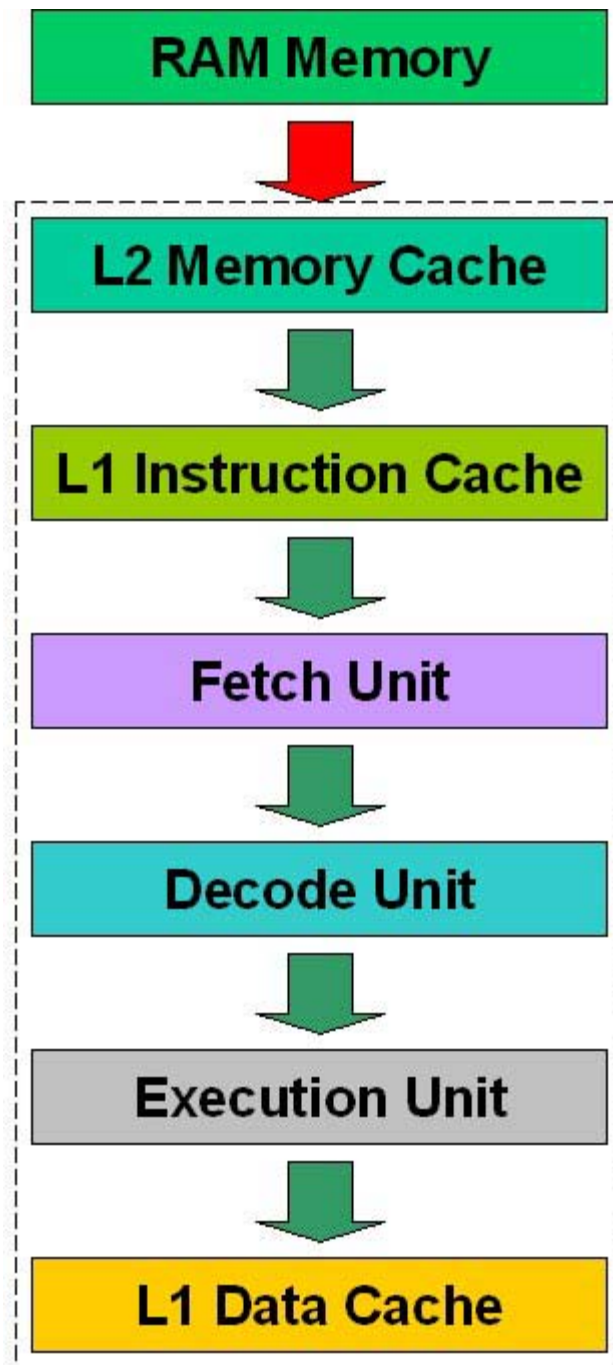
Hình 2.4.3: Truyền tải nhiều dữ liệu trên mỗi chu kỳ clock

Chính vì điều đó nên các CPU của AMD được liệt vào loại có tốc độ gấp hai clock ngoài thực. Ví dụ, một CPU của AMD với external clock là 200MHz được liệt vào CPU có clock ngoài là 400MHz. Điều tương tự cũng được áp dụng đối với các CPU của Intel, với external clock là 200MHz thì CPU của nó sẽ có tốc độ clock ngoài là 800Mhz.

Kỹ thuật truyền tải hai dữ liệu trên mỗi một chu kỳ clock được gọi là DDR (Dual Data Rate), còn kỹ thuật truyền tải 4 dữ liệu trên một chu kỳ clock được gọi là QDR (Quad Data Rate).

2.4.3 Sơ đồ khối của một CPU

Trên hình 2.4.4 sơ đồ khối cơ bản của một CPU hiện đại. Có nhiều sự khác nhau giữa các kiến trúc của AMD và Intel. Việc hiểu được các kiến thức cơ bản này sẽ là một bước để ta có thể hiểu được cách các CPU của Intel và AMD làm việc như thế nào và sự khác nhau giữa chúng. Dòng nét chấm trên hình 2.4.4 thể hiện phần “body” của CPU, vì bộ nhớ RAM được đặt bên ngoài CPU. Đường dữ liệu giữa bộ nhớ RAM và CPU thường là 64-bit (hoặc 128-bit khi sử dụng cấu hình bộ nhớ kênh dual), đang sử dụng clock nhớ hoặc clock ngoài của CPU (clock thấp). Số lượng bit đã sử dụng và tốc độ clock có thể được kết hợp trong một khối có tên gọi là tốc độ truyền tải, tính theo MB/s. Để tính toán tốc độ truyền tải, công thức được thực hiện tính tốc độ này bằng số bit x clock/8. Với hệ thống sử dụng các bộ nhớ DDR400 trong cấu hình kênh đơn (64 bit) thì tốc độ truyền tải sẽ là 3.200MB/s, còn với hệ thống tương tự sử dụng các bộ nhớ kênh dual (128 bit) sẽ có tốc độ truyền tải bộ nhớ là 6.400 MB/s.



Hình 2.4.4. Sơ đồ khối cơ bản của một CPU

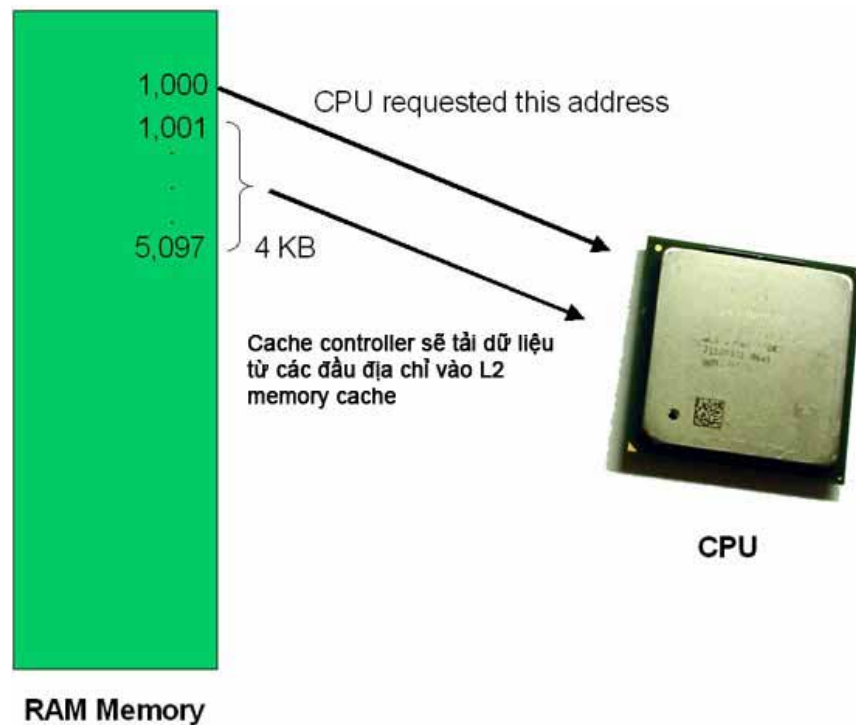
Tất cả các mạch bên trong phần đánh dấu chấm chạy ở tốc độ clock trong của CPU. Phụ thuộc vào CPU mà một số phần bên trong có nó có thể chạy ở tốc độ clock cao hơn. Cũng vậy, đường dữ liệu giữa các khối CPU có thể rộng hơn, nghĩa là truyền tải nhiều bit hơn trên mỗi chu kỳ clock 64 và 128. Ví dụ, đường dữ liệu giữa bộ nhớ cache L2 và cache chỉ lệnh L1 trên các bộ vi xử lý hiện đại thường là 256 bit. Số bit được truyền tải trên mỗi chu kỳ clock càng cao thì sự truyền tải sẽ được thực hiện càng nhanh (hay nói cách

khác, tốc độ truyền tải sẽ cao hơn). Trên hình 2.4.4, mũi tên giữa bộ nhớ RAM và cache nhớ L2; mũi tên giữa các khối khác để diễn tả tốc độ clock khác nhau và bề rộng của đường dữ liệu đã sử dụng.

2.4.4 Memory Cache

Memory Cache là một kiểu bộ nhớ hiệu suất cao, cũng được gọi là bộ nhớ tĩnh. Kiểu bộ nhớ đã sử dụng trên bộ nhớ RAM chính của máy tính được gọi là bộ nhớ động. Bộ nhớ tĩnh tiêu tốn nhiều năng lượng điện hơn, đắt hơn và có kích thước vật lý lớn hơn so với bộ nhớ động, tuy nhiên nó lại chạy nhanh hơn. Nó có thể làm việc với cùng tốc độ clock của CPU, điều mà bộ nhớ động không thể thực hiện được. Khi CPU cần nạp dữ liệu ở ngoài, nó phải làm việc ở tốc độ clock thấp hơn do vậy mà kỹ thuật cache nhớ được sử dụng ở đây để khắc phục nhược điểm này. Khi CPU nạp dữ liệu từ một vị trí nhớ nào đó thì mạch điều khiển memory cache controller nạp vào cache nhớ một khối dữ liệu bên dưới vị trí hiện hành mà CPU đã nạp. Vì các chương trình được thực hiện theo thứ tự nên vị trí nhớ tiếp theo mà CPU sẽ yêu cầu có thể là bị trí ngay dưới vị trí nhớ mà nó đã nạp. Do memory cache controller đã nạp rất nhiều dữ liệu dưới vị trí nhớ đầu tiên được đọc bởi CPU nên dữ liệu kế tiếp sẽ ở bên trong cache nhớ, chính vì vậy CPU không cần phải thực hiện thao tác lấy dữ liệu bên ngoài: nó đã được nạp vào bên trong cache nhớ nhúng trong CPU, chính vì nhúng trong CPU mà chúng có thể truy cập bằng tốc độ clock trong.

Cache controller luôn luôn quan sát các vị trí nhớ đã và đang được nạp dữ liệu từ một vài vị trí nhớ sau khi vị trí nhớ vừa được đọc. Một ví dụ thực tế, nếu một CPU đã nạp dữ liệu được lưu tại địa chỉ 1.000 thì cache controller sẽ nạp dữ liệu từ “n” địa chỉ sau địa chỉ 1.000. Số “n” được gọi là trang; nếu một bộ vi xử lý này làm việc với 4KB trang (giá trị điển hình) thì nó sẽ nạp dữ liệu từ các địa chỉ 4.096 dưới vị trí nhớ hiện



Hình 2.4.5: Memory cache controller làm việc như thế nào

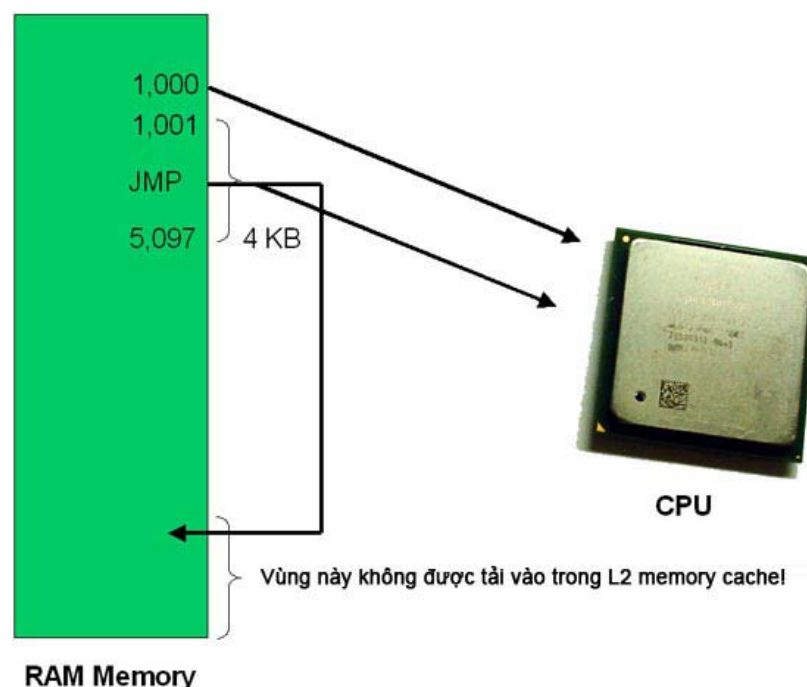
hành đang được nạp (địa chỉ 1.000 trong ví dụ). 1KB bằng 1.024 byte, do đó là 4,096 chứ không phải 4,000. Chúng tôi đã thể hiện ví dụ này trên hình 2.4.5. Memory cache càng lớn thì cơ hội cho dữ liệu yêu cầu bởi CPU ở đây càng cao, chính vì vậy CPU sẽ giảm sự truy cập trực tiếp vào bộ nhớ RAM, do đó hiệu suất hệ thống tăng (hãy nên nhớ rằng khi CPU cần truy cập trực tiếp vào bộ nhớ RAM thì nó phải thực hiện ở tốc độ clock thấp hơn nên giảm hiệu suất của toàn hệ thống).

Chúng ta gọi là “hit” khi CPU nạp một dữ liệu yêu cầu từ cache và “miss” nếu dữ liệu yêu cầu không có ở đó và CPU cần phải truy cập vào bộ nhớ RAM của hệ thống. L1 và L2 tương ứng là “Level 1” và “Level 2”, được đại diện cho khoảng cách chúng cách lõi CPU (khởi thực thi). Một sự ngờ vực hay có ở đây là tại sao có đến 3 bộ nhớ Cache (L1 data cache, L1 instruction cache và L2 cache). Hãy chú ý trên hình 2.4.5 và thấy được rằng L1 instruction cache làm việc như một “input cache”, trong khi đó L1 data cache làm việc như một “output cache”. L1 instruction cache – thường nhỏ hơn L2 cache – chỉ hiệu quả khi chương trình bắt đầu lặp lại một phần nhỏ của nó (loop), vì các chỉ lệnh yêu cầu sẽ gần hơn với khối tìm nạp. Trên trang chi tiết kỹ thuật của một CPU, L1 cache có thể được thể hiện bằng một hình ảnh hoàn toàn khác. Một số nhà máy sản xuất liệt kê hai L1 cache riêng biệt (đôi khi gọi cache chỉ lệnh là “I” và cache dữ liệu là “D”), một số hãng ghi số lượng của cả hai là 128 KB nhưng điều đó có nghĩa là 64 KB cho

cache chỉ lệnh và 64 KB cho cache dữ liệu. Mặc dù vậy đối với các CPU Pentium 4 và Celeron đời mới dựa trên socket 478 và 775 thì không có hiện tượng này. Các bộ vi xử lý Pentium 4 (và các bộ vi xử lý Celeron sử dụng socket 478 và 775) không có L1 instruction cache mà thay vào đó chúng có một trace execution cache, đây là cache được đặt giữa khối giải mã và khối thực thi. Chính vì vậy đây là L1 instruction cache nhưng tên đã được thay đổi và ở một vị trí cũng khác. Chúng ta đang đề cập đến điều này là vì đây là một lỗi rất thường xảy ra khi nghĩ rằng các bộ vi xử lý Pentium 4 không có L1 instruction cache. Vậy khi so sánh Pentium 4 với các CPU khác mọi người hãy nghĩ rằng L1 cache của nó nhỏ hơn nhiều.

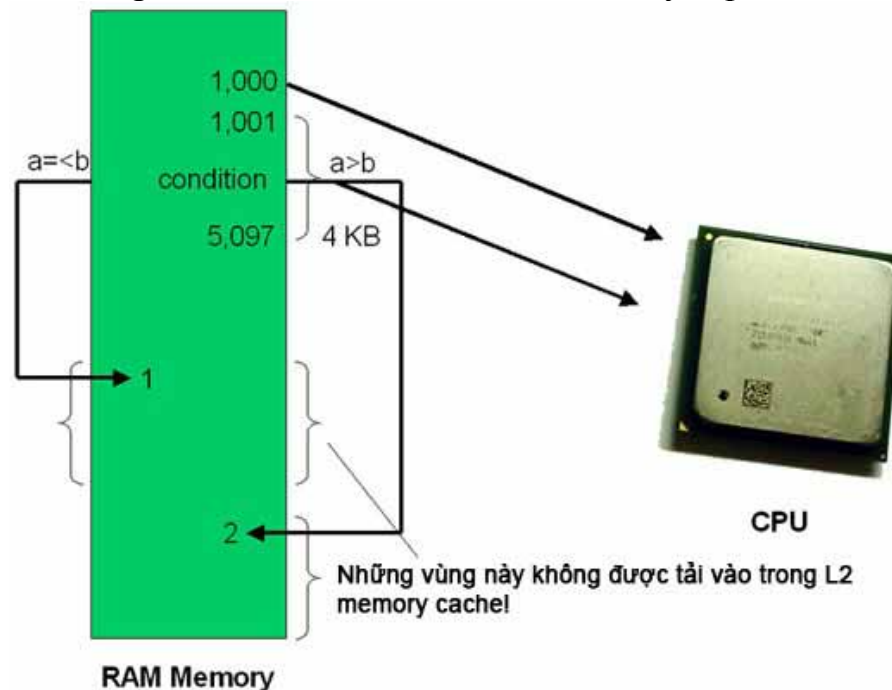
2.4.6 Rẽ nhánh

Nhưng chúng tôi đã đề cập đến một vài lần từ trước, một trong những vấn đề chính đối với các CPU là có quá nhiều “miss” đối với cache, vì khối tìm nạp phải truy cập trực tiếp vào bộ nhớ RAM (chậm), nên làm chậm cả hệ thống. Thường sử dụng cache nhớ tránh được rất nhiều vấn đề này nhưng có một giải pháp điển hình có thể giải quyết vấn đề này đó là rẽ nhánh: Nếu ở giữa chương trình có một chỉ lệnh JMP (“jump” hoặc “go to”) gửi chương trình đến một vị trí nhớ khác hoàn toàn, vị trí mới này sẽ không được nạp trong L2 memory cache, mà chỉ làm cho khối tìm nạp vào vị trí đó một cách trực tiếp trong bộ nhớ RAM. Để giải quyết vấn đề này, cache controller của các CPU hiện đại phân tích khối nhớ mà nó đã nạp và bất cứ khi nào có tìm thấy một chỉ lệnh JMP thì nó sẽ nạp khối nhớ này vào vị trí đó trong L2 memory cache trước khi CPU xử lý chỉ lệnh JMP đó.



Hình 2.4.6. Giải pháp nhánh không điều kiện

Điều này quả mang lại sự thực thi dễ dàng hơn nhiều, vấn đề ở đây là khi chương trình có một rẽ nhánh điều kiện, nghĩa là địa chỉ mà chương trình sẽ vào phụ thuộc vào một điều kiện vẫn chưa được biết. Ví dụ, nếu $a \leq b$ vào địa chỉ 1, hoặc nếu $a > b$ thì vào địa chỉ 2. Minh họa ví dụ này trên hình 2.4.7. Điều này sẽ tạo ra một “miss” đối với cache, vì các giá trị của a và b hoàn toàn không được biết đến và cache controller sẽ chỉ đang xem xét các chỉ lệnh giống JMP. Giải pháp thực hiện ở đây là: cache controller nạp cả hai điều kiện vào cache nhớ. Sau khi CPU xử lý chỉ lệnh rẽ nhánh, nó sẽ đơn giản loại bỏ một trường hợp không được chọn. Việc nạp bộ nhớ cache với dữ liệu không cần thiết sẽ tốt hơn so với việc truy cập vào bộ nhớ RAM.



Hình 2.4.7: Giải pháp rẽ nhánh có điều kiện

2.4.7 Việc xử lý chỉ lệnh

Khối tìm nạp chịu hoàn toàn trách nhiệm về việc nạp các chỉ lệnh từ bộ nhớ. Đầu tiên, nó xem xem chỉ lệnh được yêu cầu bởi CPU có trong L1 instruction cache hay không. Nếu không có ở đây, nó sẽ vào L2 memory cache. Nếu chỉ lệnh cũng không có trong L2 memory cache thì nó sẽ phải nạp trực tiếp từ bộ nhớ RAM. Khi ta bật máy tính, tất cả các cache đều trống rỗng, tuy nhiên khi hệ thống bắt đầu nạp hệ điều hành, CPU bắt đầu xử lý các chỉ lệnh đầu tiên từ ổ cứng và cache controller bắt đầu nạp các cache và đó là những gì bắt đầu để chuẩn bị thực hiện xử lý một chỉ lệnh. Sau khi khối tìm nạp đã có được chỉ lệnh cần thiết cho CPU để được xử lý, nó gửi chỉ lệnh này đến khối giải mã. Khối giải mã sẽ chỉ ra chỉ lệnh này thực hiện những nhiệm vụ gì. Nó thực hiện điều đó bằng cách hỏi ý kiến bộ nhớ ROM

tồn tại bên trong CPU, được gọi là microcode. Mỗi chỉ lệnh mà CPU hiểu đều có một microcode của nó. Microcode sẽ “ra lệnh” cho CPU thực hiện những gì. Nó giống như hướng dẫn từng bước trong các tài liệu hướng dẫn. Ví dụ, nếu chỉ lệnh đã nạp bổ sung $a+b$ thì microcode của nó sẽ bảo với khối giải mã rằng nó cần có hai tham số a và b . Khối giải mã sau đó sẽ yêu cầu khối tìm nạp lấy dữ liệu có trong hai vị trí nhớ kế tiếp, phù hợp với các giá trị của a và b . Sau khi khối giải mã “dịch” xong chỉ lệnh và lấy được tất cả dữ liệu cần thiết để thực thi chỉ lệnh, nó sẽ gửi tất cả dữ liệu này và hướng dẫn từng bước về cách thực thi chỉ lệnh đó đến khối thực thi. Khối thực thi sẽ thực thi chỉ lệnh này. Trên các CPU hiện đại, ta sẽ thấy có nhiều khối thực thi làm việc song song. Điều này được thực hiện để tăng hiệu suất của CPU. Ví dụ, một CPU có 6 khối thực thi sẽ có thể thực thi đến 6 chỉ lệnh song song đồng thời, chính vì vậy theo lý thuyết nó hoàn toàn có thể thực hiện được một hiệu suất bằng với 6 bộ vi xử lý mà chỉ có một khối thực thi. Kiểu kiến trúc này được gọi là kiến trúc “superscalar”. Thông thường các CPU hiện đại không có nhiều khối thực thi giống nhau; chúng có các khối thực thi dành riêng cho mỗi loại chỉ lệnh. Một ví dụ dễ hiểu nhất ở đây là FPU, Float Point Unit, khối chịu trách nhiệm thực thi các chỉ lệnh toán học phức tạp. Thường giữa khối giải mã và khối thực thi có một khối (gọi là khối gửi đi hoặc lập biểu) chịu trách nhiệm về việc gửi chỉ lệnh đến đúng khối thực thi, có nghĩa là nếu là một chỉ lệnh toán học thì nó sẽ gửi chỉ lệnh đó đến FPU chứ không gửi đến khối thực thi chung. Cũng vì vậy các khối thực thi chung được gọi là ALU (Arithmetic and Logic Unit). Cuối cùng, khi việc xử lý được thực hiện, các kết quả sẽ được gửi đến L1 data cache. Tiếp tục ví dụ $a+b$ của chúng ta, kết quả sẽ được gửi ra L1 data cache. Kết quả này có thể sau đó được gửi lại đến bộ nhớ RAM hoặc đến một địa điểm khác như video card chẳng hạn. Tuy nhiên điều này sẽ phụ thuộc vào chỉ lệnh kế tiếp sẽ được xử lý tiếp theo (chỉ lệnh kế tiếp có thể là in kết quả ra màn hình). Một tính năng thú vị khác mà tất cả các bộ vi xử lý đều có đó là “pipeline” – trong thiết kế máy tính đây là một tuyến lắp ráp thuộc phần cứng làm tăng tốc độ xử lý các lệnh thông qua quá trình thực hiện, truy tìm và ghi trở lại. Thiết kế này có khả năng có một số chỉ lệnh khác ở một số tầng khác của CPU ở cùng thời điểm. Sau khi khối tìm nạp đã gửi chỉ lệnh đến khối giải mã, nó sẽ không làm gì (nhàn rỗi)? Vậy về việc thay thế không làm gì bằng cách cho khối này lấy chỉ lệnh kế tiếp thì sao? Khi chỉ lệnh đầu tiên vào tới khối thực thi, khối chỉ lệnh có thể gửi chỉ lệnh thứ hai đến khối giải mã và lấy chỉ lệnh thứ ba, và quá trình cứ tiếp tục như vậy. Trong CPU hiện đại có pipeline 11 tầng (mỗi tầng là một khối của CPU), nó sẽ có thể có đến 11 chỉ lệnh bên trong tại cùng một thời điểm. Trong thực tế, khi tất cả các CPU hiện đại đều có kiến trúc “superscalar” thì số chỉ lệnh đồng thời bên trong CPU sẽ cao hơn. Cũng vậy, với CPU pipeline có 11 tầng, một chỉ lệnh được

thực thi hoàn toàn sẽ phải chuyển qua 11 khối. Nếu càng có nhiều số tầng hay khối như vậy thì lượng thời gian mà mỗi chỉ lệnh giữ chậm để được thực thi sẽ nhiều hơn. Hay nói cách khác, hãy nhớ rằng một số chỉ lệnh có thể chạy bên trong CPU cùng một thời điểm. Chỉ lệnh đầu tiên đã nạp bởi CPU có thể giữ chậm 11 bước để được xử lý xong, nhưng khi nó đi ra thì chỉ lệnh thứ hai sẽ cũng được xử lý ngay sau đó (chỉ mất một số bước giữ chậm chứ không phải là toàn bộ 11 tầng). Có một số mẹo khác được sử dụng bởi các CPU hiện đại nhằm tăng hiệu suất hệ thống.

Chúng sẽ xét hai trong số chúng, đó là thực thi không tuân theo thứ tự (OOO) và thực thi có suy đoán

a. Thực thi không tuân theo thứ tự (OOO)

Hãy nhớ rằng chúng tôi đã nói rằng các CPU hiện đại có một số khối thực thi làm việc song song và có một số kiểu khác đối với các khối thực thi, như ALU - khối thực thi chung, và FPU – khối thực thi toán học. Hãy lấy một ví dụ chung để hiểu rõ vấn đề này, chúng ta hãy cho CPU ví dụ có 6 cỗ máy thực thi, 4 chỉ lệnh chung (generic instruction) cho ALU và 2 chỉ lệnh toán học (math instruction) cho FPU. Chúng ta cũng cho rằng chương trình có thứ tự chỉ lệnh dưới đây.

1. *chỉ lệnh chung (ALU)*
2. *chỉ lệnh chung*
3. *chỉ lệnh chung*
4. *chỉ lệnh chung*
5. *chỉ lệnh chung*
6. *chỉ lệnh chung*
7. *chỉ lệnh toán học (FPU)*
8. *chỉ lệnh chung*
9. *chỉ lệnh chung*
10. *chỉ lệnh toán học*

Điều gì sẽ xảy ra? Khối gửi đi/lập lịch sẽ gửi 4 chỉ lệnh đầu tiên đến các khối ALU nhưng sau đó chỉ lệnh thứ 5 CPU sẽ cần phải đợi cho một chỉ lệnh của ALU của chúng được giải phóng để tiếp tục xử lý, vì lúc này tất cả

4 khối thực thi chung đều bận cả. Điều này không tốt bởi vì chúng ta vẫn có 2 chỉ khối toán học (FPU) chưa dùng đến, rõ ràng chúng đang trong chế độ nhàn rỗi. Chính vì vậy, một thực thi không tuân theo thứ tự (OOO) (tất cả các CPU hiện đại đều có tính năng này) sẽ xem chỉ lệnh kế tiếp xem nó có thể được gửi đến một trong hai khối thực thi đang nhàn rỗi kia không. Trong ví dụ của chúng ta, nó không thể, vì chỉ lệnh thứ 6 cũng cần đến một khối thực thi chung (ALU) để xử lý. Cổ máy thực thi không tuân theo thứ tự vẫn tiếp tục công việc tìm kiếm của nó và tìm ra rằng chỉ lệnh thứ 7 là một chỉ lệnh toán học và có thể được thực thi tại khối thực thi toán học đang nhàn rỗi. Do các khối thực thi toán học khác vẫn đang nhàn rỗi nên nó sẽ vào chương trình để tìm kiếm chỉ lệnh toán học khác. Trong ví dụ của chúng ta, nó sẽ nhảy qua chỉ lệnh thứ 8 và 9 và nạp chỉ lệnh thứ 10.

Trong ví dụ của chúng ta, các khối thực thi sẽ luôn xử lý tại cùng một thời điểm, các chỉ lệnh được thực thi lúc này là chỉ lệnh thứ 1, 2, 3, 4, 7 và 10. Tên OOO đến từ thực tế rằng CPU không cần phải đợi mà nó có thể kéo một chỉ lệnh ở cuối chương trình và xử lý nó trước các chỉ lệnh ở trên. Rõ ràng cổ máy thực thi không tuân theo thứ tự OOO không thể mãi tìm kiếm một chỉ lệnh nếu không có chỉ lệnh nào cần (ví dụ như trong ví dụ trên là không có chỉ lệnh toán học chẳng hạn). Cổ máy này của tất cả các CPU có một giới hạn nhất định về số lượng chỉ lệnh mà có thể tìm (thường là 512).

b. Thực thi có suy đoán

Hãy cho rằng một trong những chỉ lệnh chung là một chỉ lệnh rẽ nhánh có điều kiện. Vậy cổ máy thực thi OOO sẽ thực hiện những gì? Nếu CPU bổ sung một tính năng gọi là thực thi có suy đoán (tất cả các CPU hiện đại đều có), nó sẽ thực thi cả hai nhánh. Xem xét ví dụ bên dưới.

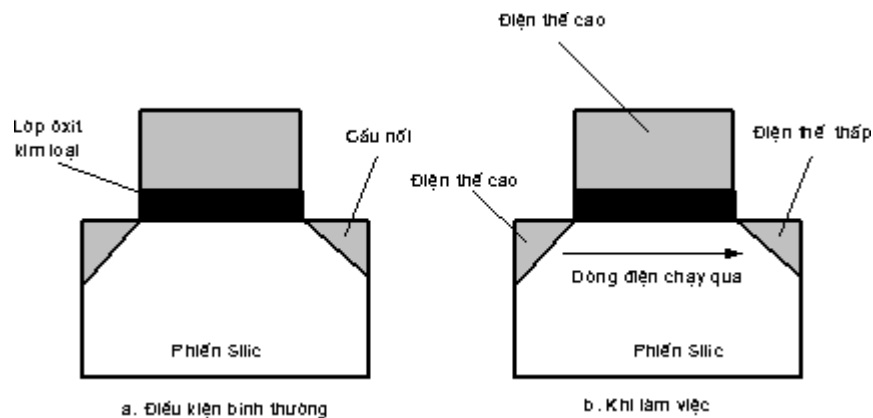
1. *chỉ lệnh chung*
2. *chỉ lệnh chung*
3. *nếu $a \leq b$ tới chỉ lệnh 15*
4. *chỉ lệnh chung*
5. *chỉ lệnh chung*
6. *chỉ lệnh chung*
7. *chỉ lệnh toán học*
8. *chỉ lệnh chung*
9. *chỉ lệnh chung*
10. *chỉ lệnh toán học*
- ...
15. *chỉ lệnh toán học*
16. *chỉ lệnh chung*
- ...

Khi cỗ máy thực thi không theo thứ tự phân tích chương trình này, nó sẽ kéo chỉ lệnh 15 vào FPU, lúc này FPU đang nhàn rỗi. Chính vì vậy tại thời điểm này, chúng ta có cả hai nhánh cùng được xử lý đồng thời. Nếu khi CPU kết thúc việc xử lý chỉ lệnh thứ ba biết được $a > b$ thì CPU sẽ loại bỏ việc xử lý của chỉ lệnh 15. Ta có thể nghĩ điều này gây tốn thời gian nhưng trong thực tế nó hoàn toàn không tốn thời gian. Nó hoàn toàn không đáng bao nhiêu để CPU thực thi chỉ lệnh riêng đó, vì FPU kiểu gì cũng nhàn rỗi. Mặt khác nếu $a \leq b$ thì CPU sẽ có được mức lợi về hiệu suất ở đây, vì khi chỉ lệnh thứ ba yêu cầu chỉ lệnh 15, đây là chỉ lệnh đã được xử lý rồi, tiếp theo đó là chỉ lệnh 16, và các chỉ lệnh sau đó. Chỉ lệnh 16 cũng đã được xử lý bởi cỗ máy thực thi không theo thứ tự.

2.5. CÔNG NGHỆ SOI

2.5.1 Các công nghệ chế tạo vi mạch hiện tại

Vật liệu bán dẫn là một loại vật liệu không dẫn điện ở điều kiện thường nhưng dẫn điện ở một điều kiện đặc biệt nào đó. Công nghệ hiện tại dựa vào một lớp ôxít kim loại nằm trên phiến silíc kết nối bởi các đường hợp chất dẫn điện. Lớp kim loại ôxít đóng vai trò như một transistor, khi được nối với nguồn có điện thế cao, lớp ôxít này làm cho phần silíc bên dưới trở nên dẫn điện và cho dòng điện được truyền từ cầu nối này qua cầu nối kia, tạo thành các vi mạch điện tử thuộc loại “bật/tắt” hay “1/0” - nguồn gốc của công nghệ máy vi tính hiện đại (Hình 2.5.1).



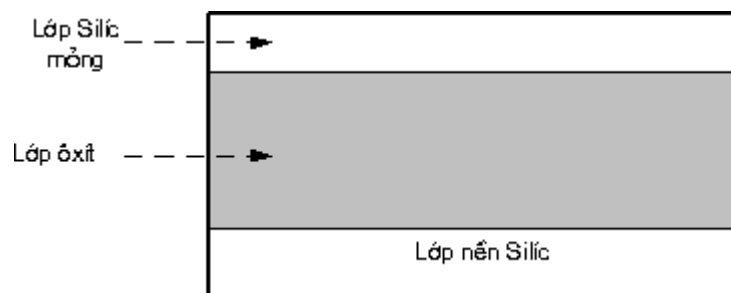
Hình 2.5.1. Nguyên lý làm việc của vi mạch điện tử

Khi làm việc, dòng điện sẽ chạy từ cầu nối có điện thế cao sang cầu nối có điện thế thấp mỗi khi phiến silíc dẫn điện. Người ta điều khiển việc này bằng cách cho dòng điện đi qua lớp ôxít bên trên khi nào cần dẫn điện và ngắt khi không cần. Công nghệ này được gọi là công nghệ MOS

(Metal Oxide Semi-Conductor - bán dẫn ôxít kim loại). Một công nghệ khác nữa là CMOS (Complimentary MOS - MOS bổ trợ), CMOS chỉ yêu cầu điện thế thấp chạy qua lớp ôxít kim loại, ngược với MOS. Hầu hết các thiết bị bán dẫn, đặc biệt là máy tính, đều dùng một hoặc cả hai công nghệ này. Các cầu nối trên càng ngày càng nhỏ đi cùng với sự thu nhỏ của liên kết CPU qua các công nghệ $0,25 \rightarrow 0,18 \rightarrow 0,13\mu\text{m}$... Công nghệ nói trên càng ngày càng khó áp dụng mà không xảy ra hiện tượng đóng điện chéo qua các cầu khác không liên quan nằm bên cạnh do chúng nằm quá sát nhau. Do vậy, công nghệ này cần phải được thay đổi nếu muốn có được những bước tiến mới trong sản xuất các linh kiện bán dẫn nói chung, và sản xuất CPU nói riêng. Các cải tiến khác cho công nghệ MOS/CMOS có sẵn cũng mang đến một sự tiến bộ nào đó, bằng chứng là cả AMD và Intel đều đã sản xuất sản phẩm của mình bằng công nghệ $0,13\mu\text{m}$.

2.5.2 Công nghệ SOI

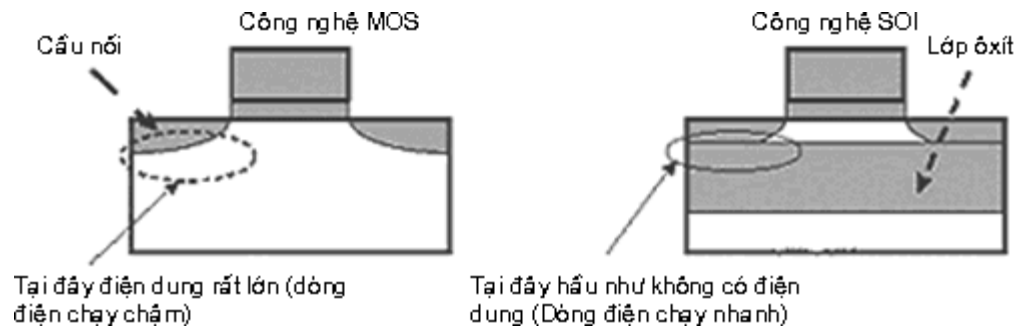
Trong công nghệ SOI, một lớp vật liệu cách điện được chèn vào giữa phiến silíc, để lại một phần silíc nhỏ ở giữa các cầu nối (Hình 2.5.2). Lợi thế của SOI là với sự chèn thêm lớp cách điện này, điện dung của tụ silíc giữa các cầu được cực tiểu hoá, do đó giảm thời gian cần thiết để thoát/nạp, để mở và đóng cầu nối. Điều này giúp tăng số công việc xử lý được trong một đơn vị thời gian.



Hình 2.5.2. Công nghệ SOI

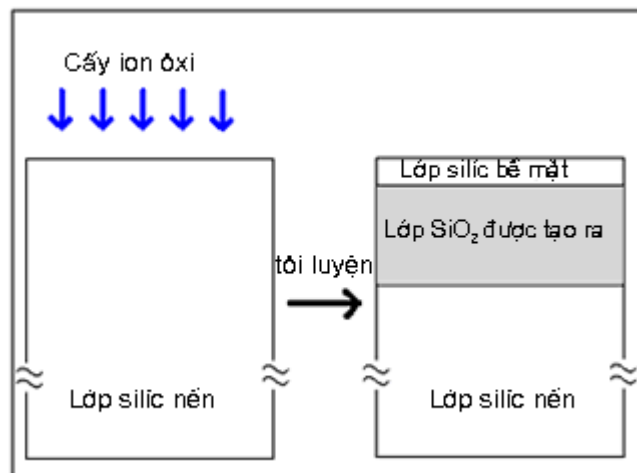
Hình 2.5.3 là một ví dụ so sánh giữa một mạch điện MOS và một mạch điện SOI. Điểm bất lợi của vi xử lý dùng công nghệ MOS là phần silíc ở giữa các cầu nối (có vai trò như một tụ điện) phải nạp được điện dung tối đa để có thể đóng - và lại phải thoát hết điện dung khi mở. Việc này tốn thời gian xử lý, lãng phí thời gian xử lý trên CPU và là điều mà cả các nhà sản xuất lẫn chúng ta đều không mong muốn. Còn đối với công nghệ SOI thì phần silíc giữa các cầu nhỏ, thời gian tích điện nhỏ, tốc độ nhanh. Lớp cách điện được dùng trong công nghệ SOI phổ biến là một dạng của

ôxít silíc hay thậm chí thuỷ tinh, nhưng có cấu trúc khác với cấu trúc pha lê dẫn điện của phiến silíc.



Hình 2.5.3. So sánh công nghệ MOS và SOI

Về mặt hoá học, rất khó có thể ghép được 2 lớp silíc có cấu trúc pha lê và không phải pha lê với nhau. Hãng IBM đã sử dụng một kĩ thuật có tên là SIMOX (Seperation by Implantation of Oxygen - ngăn cách bởi phương pháp cấy khí ôxi) để tạo một lớp ngăn cách bằng ôxít silíc (SiO_2) trên phiến silíc. Khí ôxi được ép lên bề mặt của bề mặt phiến silíc ở áp suất và nhiệt độ cao, khi đó silíc phản ứng với ôxi tạo nên một lớp ôxít silíc bám vào phiến silíc bên dưới. Tức là họ không tìm cách hàn gắn hai phần silíc và ôxít silíc vào nhau mà tạo một lớp ôxít silíc ngay trên phần silíc có sẵn (Hình 2.5.4).



Hình 2.5.4. Phương pháp SIMOX

a. Ưu điểm của SOI

SOI có nhiều ưu điểm. Thứ nhất, việc giảm thời gian đóng mở các cầu nối có nghĩa rằng các bộ vi xử lý dùng công nghệ này sẽ nhanh hơn đến

30% so với các bộ vi xử lý dùng công MOS/CMOS nếu có cùng một xung đồng hồ như nhau. Một ưu điểm nữa của SOI là các vi xử lý dùng công nghệ này sẽ yêu cầu công suất thấp hơn nhiều so với MOS/CMOS. Một xu hướng vài năm gần đây là khi mọi người sử dụng nhiều công nghệ tiên tiến hơn thì công suất của các bộ vi xử lý càng tăng theo. Ví dụ, vi xử lý 486 yêu cầu công suất khoảng 5W, trong khi đó một vi xử lý Pentium tiêu tốn khoảng 10W và một vi xử lý Pentium II 400MHz có công suất tiêu thụ khoảng 28W. Công suất tăng có nghĩa là hạn chế những ứng dụng của các bộ vi xử lý, đặc biệt là trong các ứng dụng di động. Khả năng của công nghệ SOI là yêu cầu một nguồn công suất thấp xuất phát từ thực tế mạch điện SOI có thể hoạt động tại điện thế thấp với cùng hiệu suất như công nghệ CMOS tại điện thế cao. Do đó, SOI sẽ có một tác động rất lớn vào các ứng dụng yêu cầu công suất thấp chẳng hạn như các ứng dụng vô tuyến và xách tay.

Bên cạnh đó, SOI cho phép thu nhỏ vi mạch lại đáng kể. Việc thu nhỏ tiến trình sản xuất xuống 90nm (0,09 μ m) hay thấp hơn cùng với SOI có nghĩa rằng các bộ vi xử lý sẽ được tăng tốc rất nhanh và tốc độ 5-10GHz sẽ sớm đạt được.

b. Tương lai của công nghệ SOI

Tuy SOI có rất nhiều ưu điểm so với MOS/CMOS nhưng nó sẽ không thay thế hoàn toàn MOS/CMOS mà chỉ tối ưu hoá cho hai công nghệ này. SOI sẽ được kết hợp với các công nghệ khác để tạo ra các loại vi xử lý mới. AMD, Intel và IBM đang nghiên cứu công nghệ 90nm, bước tiếp theo trong quá trình phát triển công nghệ chế tạo vi mạch. Intel hi vọng sẽ đưa ra bộ vi xử lý Pentium 4 dựa trên công nghệ này vào nửa cuối năm 2003, trong khi đó các sản phẩm của AMD sẽ được đưa vào sản xuất trong quý 4 năm 2003 và đưa ra thị trường vào quý 1 năm 2004. Và vừa qua, IBM và AMD đã ký một thoả thuận cùng nghiên cứu và phát triển các loại vi xử lý mới dựa trên công nghệ 65nm và 45nm. Đây thực sự là một bước tiến to lớn trong công nghệ chế tạo vi mạch và thúc đẩy việc ứng dụng rộng rãi vi mạch vào tất cả các lĩnh vực như công nghệ thông tin, viễn thông và tự động hoá.

2.6 kiến trúc Pentium M

khi tất cả các CPU mới của Intel sử dụng kiến trúc Pentium M, việc nghiên cứu kiến trúc này là một việc quan trọng để từ đó ta có thể hiểu sâu được kiến trúc của các CPU Core Solo hay Core Duo (Yonah) và cũng hiểu được

lớp nền tảng cho việc tiến tới kiến trúc lõi siêu nhỏ (Core microarchitecture), được sử dụng bởi các CPU Merom, Conroe và Woodcrest.

Pentium M được xây dựng dựa trên kiến trúc thế hệ thứ 6 của Intel, cùng được sử dụng trong các CPU Pentium Pro, Pentium II và Pentium III, tuy nhiên lại không trên Pentium 4 như nhiều ta nghĩ, mục đích của nó nhằm vào các máy tính di động. Ta có thể nghĩ Pentium M như một Pentium III được nâng cao. Nhưng cần chú ý để không nhầm lẫn Pentium M với Pentium III. Đôi khi Pentium M còn được gọi là Centrino. Quả thực nó có thể được gọi như vậy khi ta có một laptop CPU Pentium M, chipset Intel 855 hay 915 và Intel/PRO wireless LAN. Chính vì vậy nếu ta có một laptop được xây dựng trên Pentium M mà không có những điều kiện bổ sung như trên thì không thể được coi là Centrino. Cơ bản về cách kiến trúc P6 làm việc như thế nào và những điểm gì mới khi so sánh Pentium M với Pentium III. Cũng vì vậy mà trong này ta sẽ biết thêm được về cách làm việc của các CPU Pentium Pro, Pentium II, Pentium III và Celeron (chúng cũng chính là các mô hình dựa trên P6, nghĩa là slot 1 và socket 370). Trước khi tiếp tục, chúng ta hãy xem xét đến sự khác nhau giữa các CPU Pentium M và Pentium III:

Nhìn bên ngoài, Pentium M làm việc giống như Pentium 4, truyền tải 4 dữ liệu trên một chu kỳ clock. Kỹ thuật này được gọi là QDR (Quad Data Rate – Gấp bốn lần tốc độ dữ liệu) và làm cho bus nội bộ có hiệu suất tăng gấp 4 lần với tốc độ clock thực của nó, ta có thể xem bảng dưới đây.

Clock thực	Hiệu suất	Tốc độ truyền
100 MHz	400 MHz	3.2 GB/s
133 MHz	533 MHz	4.2 GB/s

- L1 memory cache: Hai L1 memory cache 32 KB, một cho dữ liệu và một cho chỉ lệnh (Pentium III có hai L1 memory cache 16 KB).
- L2 memory cache: 1 MB trên các mô hình 130 nm (lõi “Banias”) hay 2 MB trên các mô hình 90 nm (lõi “Dothan”). Pentium II chỉ có đến 512 KB. Celeron M, phiên bản rẻ tiền nhất của Pentium M cũng có 512 KB L2 memory cache. Hỗ trợ cho các chỉ lệnh SSE2.
- Dự báo nhánh cao cấp: Dự báo nhánh đã được thiết kế lại (và được dựa trên mạch của Pentium 4) để cải thiện hiệu suất.
- Sự hợp nhất nhiều hoạt động nhỏ: Bộ giải mã chỉ lệnh hợp nhất được hai hành động nhỏ thành một để có thể tiết kiệm được năng lượng và

cải thiện hiệu suất. Chúng ta sẽ nói kỹ hơn về vấn đề này ở phần dưới.

- Công nghệ SpeedStep nâng cao, đây là công nghệ cho phép các CPU có thể giảm được clock trong chế độ nhàn rỗi để tiết kiệm thời gian sống của pin. Một số tính năng nhằm tiết kiệm cho pin cũng đã được bổ sung vào kiến trúc siêu nhỏ của Pentium M, vì mục đích của các CPU này ban đầu được thiết kế cho máy tính di động.

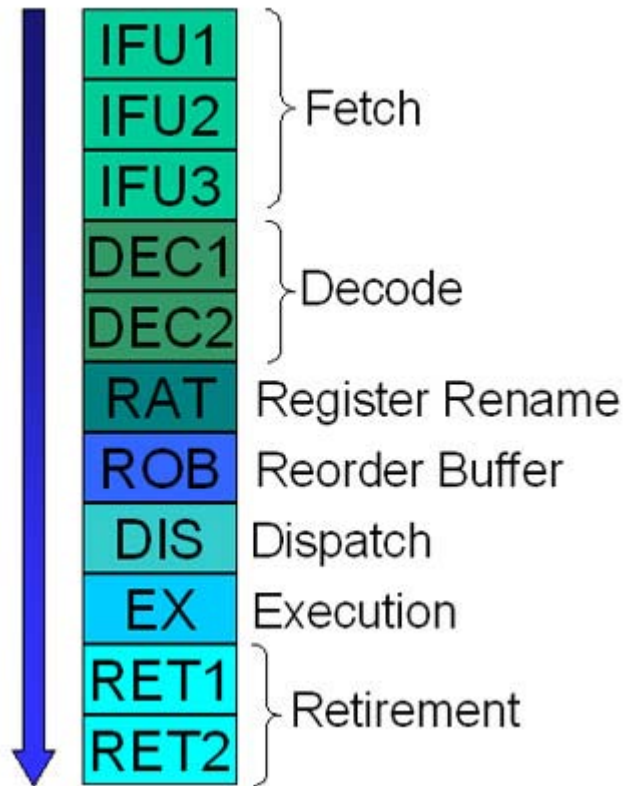
Bây giờ chúng ta hãy đi xem xét sâu hơn về kiến trúc của Pentium M.

2.6.1 Nguyên lý của Pentium M

Nguyên lý là một danh sách tất cả các tầng mà chỉ lệnh đã cho phải được thực thi theo đúng thuật toán. Intel đã không tiết lộ các nguyên lý của Pentium M, chính vì vậy chúng ta sẽ nói về nguyên lý của Pentium III. Nguyên lý của Pentium M có thể sẽ có nhiều tầng hơn so với Pentium III nhưng việc phân tích nó sẽ cho chúng ta có được ý tưởng về kiến trúc của Pentium M làm việc như thế nào.

Hãy nhớ rằng, nguyên lý làm việc của Pentium 4 có đến 20 tầng và nguyên lý làm việc của các CPU Pentium 4 mới hơn được dựa trên lõi “Prescott” có đến 31 tầng.

Trên hình 2.6.1 ta có thể thấy được nguyên lý 11 tầng của Pentium III



Hình 2.6.1: Nguyên lý của Pentium III

Dưới đây là giải thích một cách cơ bản về mỗi tầng, giải thích sẽ làm sáng tỏ cách mỗi chỉ lệnh được gán được thực hiện như thế nào bởi các bộ vi xử lý lớp P6. Đây chỉ là tóm tắt và những giải thích cụ thể dễ hiểu

IFU1: Nạp một dòng (32 byte tương đương với 256 bit) từ chỉ lệnh L1 cache và lưu nó vào trong bộ đệm luồng chỉ lệnh (Instruction Streaming Buffer).

* IFU2: Nhận dạng các chỉ lệnh đường biên (16byte tương đương với 128bit). Vì các chỉ lệnh x86 không có một chiều dài cố định nên tầng này đánh dấu vị trí mà mỗi chỉ lệnh bắt đầu và kết thúc bên trong 16byte đã được nạp. Nếu có bất kỳ nhánh nào bên trong 16byte thì địa chỉ có nó sẽ được lưu tại Branch Target Buffer (BTB), chính vì vậy CPU có thể sử dụng những thông tin này sau trên mạnh tiên đoán nhánh của nó.

* IFU3: Đánh dấu đơn vị giải mã chỉ lệnh của mỗi chỉ lệnh phải được gửi. Có ba khối giải mã chỉ lệnh khác nhau mà chúng ta sẽ đề cập đến chúng trong phần dưới.

* DEC1: Giải mã chỉ lệnh x86 thành những chỉ lệnh nhỏ RISC (các hoạt động nhỏ). Vì CPU có đến 3 bộ giải mã chỉ lệnh nên nó có thể giải mã được đến 3 chỉ lệnh cùng lúc.

* DEC2: Gửi các chỉ lệnh nhỏ vừa được giải mã vào hàng đợi chỉ lệnh đã giải mã (Decoded Instruction Queue), hàng đợi này có khả năng lưu trữ

được đến 6 chỉ lệnh nhỏ. Nếu chỉ lệnh đã được chuyển đổi nhiều hơn 6 chỉ lệnh nhỏ thì tầng này cần phải được lặp lại để không bỏ sót chúng.

* RAT: Vì kiến trúc P6 thực hiện việc thi hành out-of-order (không tuân theo thứ tự, viết tắt là OOO), nên giá trị của thanh ghi đã cho có thể được thay đổi bởi một chỉ lệnh được thực thi trước vị trí chương trình diễn ra, sửa dữ liệu cần thiết cho chỉ lệnh khác. Chính vì vậy để giải quyết được kiểu xung đột này, tại tầng này, thanh ghi gốc được sử dụng bởi chỉ lệnh sẽ được thay đổi thành 40 thanh ghi bên trong mà kiến trúc siêu nhỏ mà P6 có.

* ROB: Tại tầng này, ba chỉ lệnh nhỏ được giải mã sẽ nạp vào Reorder Buffer (ROB). Nếu tất cả dữ liệu đều cần thiết cho việc thực thi của một chỉ lệnh nhỏ đã được cung cấp và nếu có một khe hở tại hàng đợi chỉ lệnh đã giải mã Reservation Station thì chỉ lệnh này sẽ được chuyển vào hàng đợi này.

* DIS: Nếu chỉ lệnh đã giải mã này lại không được gửi đến hàng đợi trên thì nó có thể được thực hiện tại tầng này. Chỉ lệnh giải mã sẽ được gửi đến khối thực thi thích hợp.

* EX: Chỉ lệnh được giải mã sẽ được thực thi tại khối thực thi này. Mỗi một chỉ lệnh đã giải mã này chỉ cần một chu kỳ xung nhịp để được thực thi.

* RET1: Kiểm tra tại bộ đệm Reorder Buffer xem có bất kỳ chỉ lệnh đã giải mã nào được đánh dấu như “đã thực thi” không.

* RET2: Khi tất cả các chỉ lệnh đã giải mã có liên quan đến chỉ lệnh x86 thực sự đã được xóa hết khỏi bộ đệm Reorder Buffer và tất cả các chỉ lệnh nhỏ (đã được giải mã) có liên quan với chỉ lệnh x86 hiện hành đã được thực thi, thì các chỉ lệnh này sẽ được xóa khỏi bộ đệm Reorder Buffer và các thanh ghi x86 sẽ được nâng cấp (tiến trình được quay trở về tầng RAT). Tiến trình trở lại làm việc phải được thực hiện theo thứ tự. Ba chỉ lệnh đã giải mã có thể được xóa khỏi bộ đệm Reorder Buffer trong mỗi một chu kỳ clock.

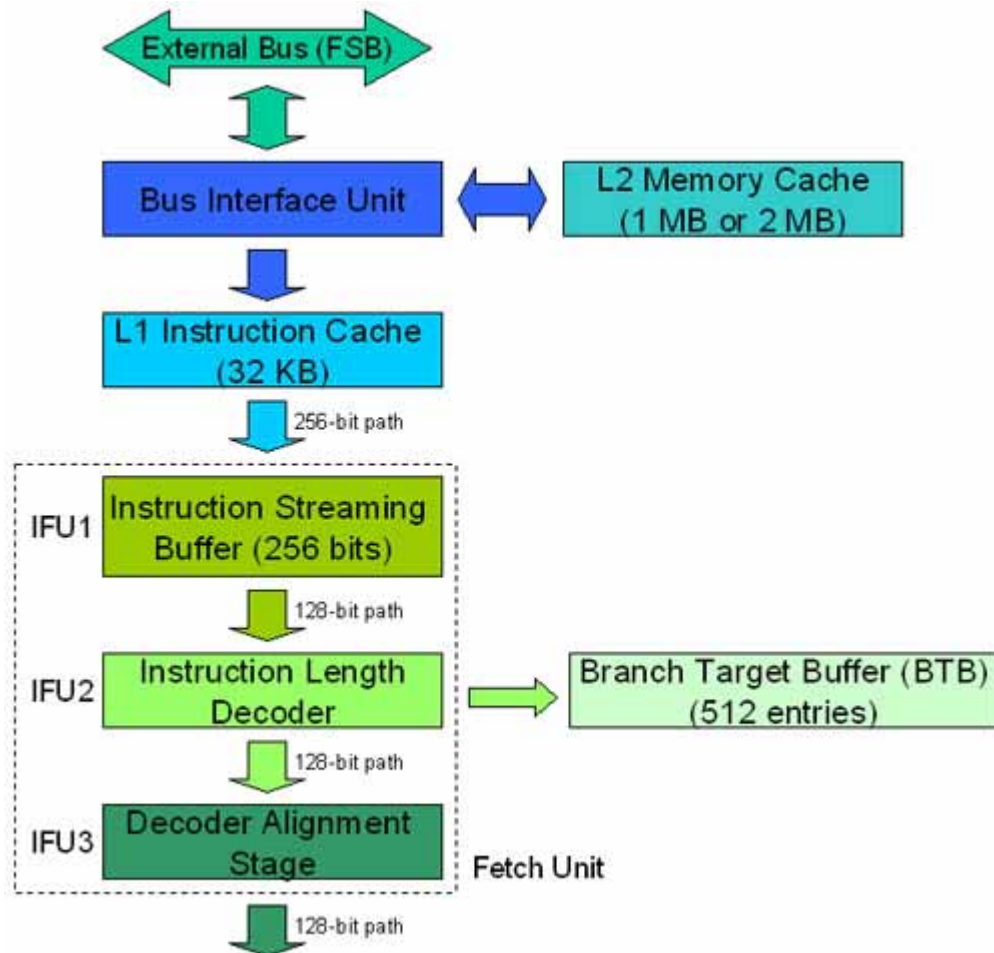
2.6.2. Memory Cache và Khối tìm nạp

Nhưng chúng tôi đã đề cập từ trước, L2 memory cache của Pentium M có thể là 1 MB trên các mô hình 130 nm (lỗi “Banias”) hay 2 MB trên các mô hình 90 nm (lỗi “Dothan”). Trong khi đó nó có hai memory cache L1, một cái là 32KB cho chỉ lệnh và cái kia là 32KB cho dữ liệu. Như đã giải thích ở phần trước, khối tìm nạp được chia thành 3 tầng. Trong hình 2.6.3, ta có thể xem được cách khối tìm nạp làm việc như thế nào.

Khối tìm nạp nạp dòng thứ nhất (32 bytes = 256 bits) vào bộ đệm luồng chỉ lệnh của nó (Instruction Streaming Buffer). Sau đó bộ giải mã chiều dài chỉ lệnh sẽ nhận ra các ranh giới chỉ lệnh bên trong mỗi 16byte. Vì chỉ lệnh x86 không có chiều dài cố định nên tầng này sẽ đánh dấu vị trí mỗi chỉ lệnh bắt đầu và kết thúc bên trong 128bit đã được nạp. Nếu có một chỉ lệnh nhánh

nào đó bên trong 128 bit đó thì địa chỉ sẽ được lưu vào Branch Target Buffer (BTB), chính vì vậy CPU của ta có thể sử dụng các thông tin này sau trên mạnh dự báo nhánh của nó. BTB có 512 đầu vào.

Sau khi tầng Decoder Alignment Stage đánh dấu khối giải mã chỉ lệnh nào thì mỗi chỉ lệnh sẽ được gửi đi. Có 3 khối giải mã chỉ lệnh khác nhau sẽ giới thiệu ở phần dưới đây.



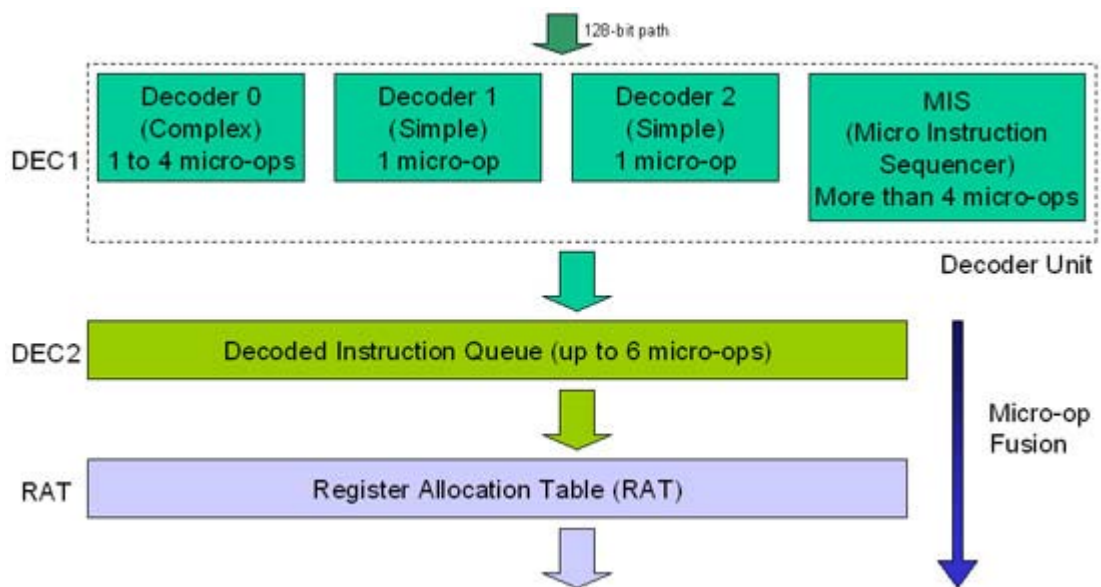
Hình 2.6.3: Khối tìm nạp

2.6.4 Giải mã chỉ lệnh và thay đổi tên cho thanh ghi

Vì kiến trúc P6 sử dụng cho các bộ vi xử lý Pentium Pro kiến trúc CISC/RISC lai nên bộ vi xử lý phải chấp nhận các chỉ lệnh CISC và cũng được biết đến với tư cách là các chỉ lệnh x86, điều này là do tất cả các phần mềm cung cấp ngày nay đều được viết bằng kiểu chỉ lệnh này. CPU chỉ sử dụng RISC không phải là tạo ra cho máy tính, vì nó không chạy phần mềm hiện nay như Windows và Office.

Vì vậy, giải pháp được sử dụng bởi tất cả các bộ vi xử lý hiện đang cung cấp

trên thị trường ngay nay từ cả Intel và AMD là đều sử dụng giải mã CISC/RISC. Bên trong, CPU xử lý các chỉ lệnh RISC nhưng front-end của nó lại chỉ chấp nhận các chỉ lệnh CISC x86. Các chỉ lệnh CISC x86 được đề cập đến như chỉ lệnh thông thường còn các chỉ lệnh RISC bên trong được đề cập đến như các chỉ lệnh đã được giải mã. Mặc dù vậy, các chỉ lệnh đã được giải mã RISC không thể được truy cập một cách trực tiếp, do đó chúng ta không thể tạo phần mềm dựa trên các chỉ lệnh này để vòng tránh qua bộ giải mã. Cũng vậy, mỗi CPU sử dụng các chỉ lệnh RISC của riêng nó, các chỉ lệnh này không được công bố và không tương thích với chỉ lệnh đã giải mã từ các CPU khác. Điều đó có nghĩa là các chỉ lệnh đã giải mã của Pentium M khác hoàn toàn với chỉ lệnh đã giải mã của Pentium 4, sự khác biệt này chính là từ các chỉ lệnh giải mã Athlon 64. Phụ thuộc vào độ phức tạp của chỉ lệnh x86 mà nó phải được chuyển thành các chỉ lệnh giải mã RISC. Bộ giải mã chỉ lệnh Pentium M làm việc giống như trên hình 2.6.3. Như những gì ta có thể quan sát thấy, có ba bộ giải mã và một bộ xếp dãy chỉ lệnh đã giải mã (MIS). Hai bộ giải mã được tối ưu hóa cho các chỉ lệnh đơn giản, trong đó các chỉ lệnh đơn giản là chỉ lệnh thường chỉ là một chỉ lệnh giải mã. Kiểu chỉ lệnh này được chuyển đổi như một chỉ lệnh giải mã. Một bộ giải mã được tối ưu hóa cho các chỉ lệnh x86 phức tạp, chỉ lệnh này có thể được chuyển đổi thành 4 chỉ lệnh đã giải mã. Nếu chỉ lệnh x86 quá phức tạp, có nghĩa là nó chuyển đổi tới hơn bốn chỉ lệnh giải mã thì nó sẽ được gửi đến MIS là bộ nhớ ROM, gồm có một danh sách các chỉ lệnh có thể được dùng để thay thế cho x86 trên.



Hình 2.6.3: Bộ giải mã và đổi tên thanh ghi

Bộ giải mã chỉ lệnh có thể chuyển đổi lên đến 3 chỉ lệnh x86 trên mỗi một chu kỳ clock, một bộ giải mã phức tạp Decoder 0 và hai bộ giải mã đơn giản

1 và 2, điều này làm cho chúng ta có cảm giác hàng đợi chỉ lệnh đã được giải mã (Decoded Instruction Queue) có thể lên đến 6 chỉ lệnh giải mã trên mỗi chu kỳ clock, kịch bản có thể khi Decoder 0 gửi 4 chỉ lệnh đã giải mã và hai bộ giải mã kia gửi mỗi bộ một chỉ lệnh đã được giải mã – hoặc khi MIS được sử dụng. Các chỉ lệnh x86 phức tạp sử dụng (MIS) Micro Instruction Sequencer có thể dữ chậm một số chu kỳ clock khi giải mã, điều đó phụ thuộc vào số lượng chỉ lệnh được giải mã sẽ tạo ra từ sự chuyển đổi. Ta cần nên lưu ý rằng Decoded Instruction Queue chỉ có thể giữ được đến 6 chỉ lệnh đã giải mã, chính vì vậy nếu có hơn 6 chỉ lệnh giải mã được sinh ra bởi bộ giải mã cộng với MIS thì một chu kỳ khác sẽ được sử dụng để gửi các chỉ lệnh hiện hành trong hàng đợi tới Register Allocation Table (RAT), làm trống hàng đợi và chấp nhận các chỉ lệnh đã giải mã mà không phù hợp với nó trước đó. Pentium M sử dụng một khái niệm mới đối với kiến trúc P6, khái niệm này được gọi là hợp nhất chỉ lệnh giải mã. Trên Pentium M, mỗi một bộ giải mã nối hai chỉ lệnh đã giải mã thành một. Chúng sẽ chỉ được tách ra khi được thực thi, tại tầng thực thi. Trên kiến trúc P6, mỗi chỉ lệnh có chiều dài 118 bit. Pentium M thay vì làm việc với các chỉ lệnh 118bit, nó làm việc với các chỉ lệnh có chiều dài 236bit mà chính là kích thước nối của hai chỉ lệnh 118bit. Cần phải lưu ý rằng các chỉ lệnh đã giải mã liên tục có chiều dài là 118bit, còn những gì được thay đổi là chúng được truyền tải thành một nhóm gồm hai chỉ lệnh cơ bản này.

Ý tưởng đằng sau phương pháp này là để tiết kiệm năng lượng và tăng hiệu suất. Việc gửi một chỉ lệnh có kích thước 236bit dài sẽ nhanh hơn việc gửi hai chỉ lệnh 118bit. Thêm vào đó, CPU sẽ tiêu tốn ít nguồn điện hơn vì sẽ có ít chỉ lệnh đã giải mã lưu thông bên trong nó. Các chỉ lệnh được gắn sau đó sẽ gửi đến bảng Register Allocation Table (RAT). Kiến trúc CISC x86 chỉ có 8 thanh ghi 32bit đó là EAX, EBX, ECX, EDX, EBP, ESI, EDI và ESP. Số lượng này là quá thấp vì các CPU hiện đại có thể thực thi mã out-of-order, và nó sẽ “phá hỏng” nội dung bên trong thanh ghi đã có, từ đó gây ra hỏng các chương trình.

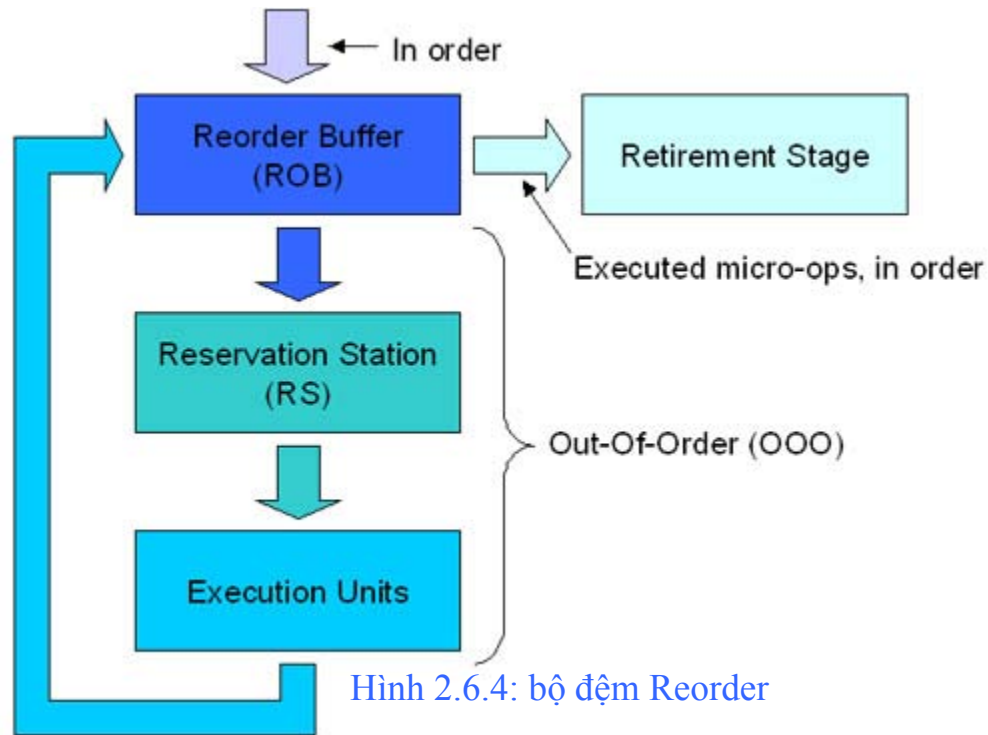
Chính vì vậy, tại tầng này, bộ vi xử lý thay đổi tên và nội dung của các thanh ghi đã được sử dụng bởi chương trình thành một trong 40 thanh ghi bên trong đã có (mỗi một thanh ghi này có 80 bit rộng, như vậy việc chấp nhận cả dữ liệu nguyên và dữ liệu thay đổi), cho phép chỉ lệnh có thể chạy tại cùng một thời điểm với chỉ lệnh khác mà sử dụng cũng cùng một thanh ghi chuẩn, hoặc thậm chí out-of-order, có nghĩa là cho phép chỉ lệnh thứ hai có thể chạy trước chỉ lệnh thứ nhất dù là chúng cùng chung trên một thanh ghi.

2.6.5 Bộ đệm Reorder Buffer

Khi các chỉ lệnh x86 và chỉ lệnh đã được giải mã có kết quả truyền tải giữ

các tầng CPU theo cùng một thứ tự thì chúng sẽ xuất hiện trên chương trình đang chạy.

Khi vào ROB, các chỉ lệnh đã giải mã có thể được nạp và thực thi out-of-order bởi các khối thực thi. Sau khi thực thi, các chỉ lệnh được gửi trở lại về Reorder



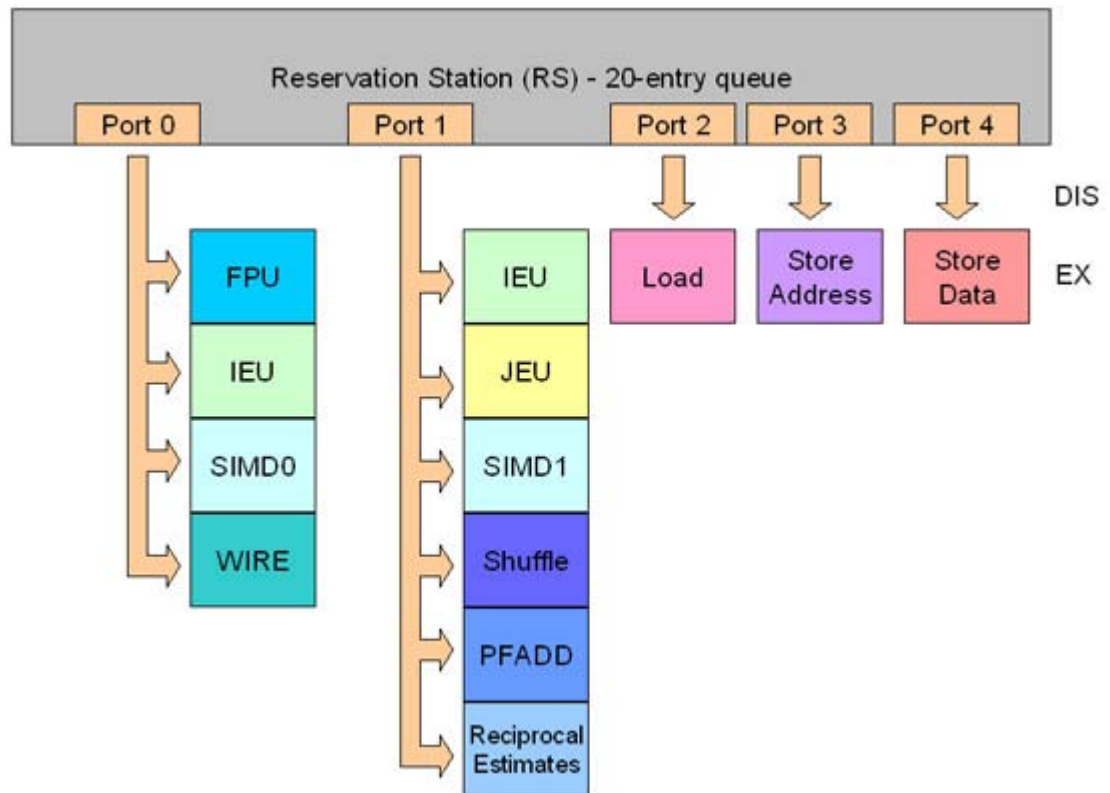
Hình 2.6.4: bộ đệm Reorder

Buffer. Sau đó tại tầng cuối cùng (Retirement), các chỉ lệnh đã thực thi được xuất ra khỏi bộ đệm Reorder Buffer với cùng thứ tự mà chúng đã nạp vào, có nghĩa là chúng được chuyển theo thứ tự. Trên hình 2.6.4, ta có thể có được ý tưởng về cách chúng làm việc như thế nào. Trên hình 2.6.4, chúng ta đã đơn giản hóa trạm dành riêng (Reservation Station) và các khối thực thi để có thể tạo sự dễ hiểu cho bộ đệm này.

2.6.6 Reservation Station và các khối thực thi

Như chúng ta đã đề cập từ trước, Pentium M sử dụng các chỉ lệnh được nối (thường là hai chỉ lệnh được nối với nhau) từ khối giải mã đến vị trí các cổng gửi đi được đặt trên Reservation Station. Reservation Station gửi đi các chỉ lệnh giải mã một cách riêng biệt (đã tách ghép đôi).

Pentium M có 5 cổng như vậy, các cổng này được đánh số từ 0 đến 4 trên Reservation Station. Mỗi cổng được kết nối đến một hoặc nhiều khối thực thi, các ta có thể xem trên hình 2.6.6.



Hình 2.6.6: Reservation Station và các khối thực thi

Dưới đây là giải thích vắn tắt về mỗi khối thực thi có trên CPU này:

- IEU: Instruction Execution Unit – Khối thực thi chỉ lệnh là nơi các chỉ lệnh thường được thực thi. Cũng được biết đến trong các sách giới thiệu về cấu trúc máy tính với tên ALU (Khối logic số học - Arithmetic and Logic Unit). Các chỉ lệnh thông thường này cũng được hiểu là các chỉ lệnh “integer”.
- FPU: Floating Point Unit là nơi các chỉ lệnh toán học phức tạp được thực thi. Trước kia, khối này cũng có tên gọi là “math co-processor” – khối đồng xử lý toán học.
- SIMD: là nơi các chỉ lệnh SIMD được thực thi, nghĩa là MMX, SSE và SSE2.
- WIRE: Các hàm phức tạp.
- JEU: Jump Execution Unit xử lý các nhánh và cũng được biết đến là Branch Unit.
- Shuffle: Khối này thực thi một loại chỉ lệnh của SSE có tên gọi là “shuffle”.
- PFADD: Thực thi một chỉ lệnh SSE có tên gọi PFADD (Packed FP Add) và cả các chỉ lệnh COMPARE, SUBTRACT, MIN/MAX và CONVERT. Khối này được cung cấp riêng, chính vì vậy nó có thể bắt đầu việc thực thi một chỉ lệnh giải mã mới mỗi chu kỳ clock dù là nó

không hoàn tất được sự thực thi của chỉ lệnh đã giải mã trước. Khối này có một độ trễ ba chu kỳ clock, nghĩa là nó sẽ giữ chậm 3 chu kỳ clock đối với mỗi chỉ lệnh đã được xử lý.

- Reciprocal Estimates: Thực thi hai chỉ lệnh SSE, một được gọi là RCP (Reciprocal.Estimate) và một gọi là RSQRT (Reciprocal Square Root Estimate).
- Load: Khối này dùng để xử lý các lệnh hỏi dữ liệu để được đọc từ bộ nhớ RAM.
- Store Address: Khối xử lý các chỉ lệnh hỏi dữ liệu để được ghi tại bộ nhớ RAM. Khối này cũng có tên gọi là AGU, Address Generator Unit. Kiểu chỉ lệnh này sử dụng cả hai khối Store Address và Store Data tại cùng một thời điểm.
- Store Data: Xử lý các chỉ lệnh hỏi dữ liệu để ghi vào bộ nhớ RAM. Loại chỉ lệnh này sử dụng cả hai khối Store Address và Store Data tại cùng một thời điểm.

Chỉ lệnh phức tạp có thể mất đến vài chu kỳ clock để được xử lý. Chúng ta hãy lấy một ví dụ của cổng 0, nơi mà khối floating point unit (FPU) có mặt ở đó. Trong khi khối này đang xử lý một chỉ lệnh rất phức tạp, mất đến vài clock để thực thi thì cổng 0 sẽ không ngừng hoạt động: nó luôn luôn gửi các chỉ lệnh đơn giản đến IEU mặc dù khi đó FPU lại đang rất bận. Chính vì vậy, mặc dù tốc độ gửi đi lớn nhất là 5 chỉ lệnh giải mã trên mỗi một chu kỳ clock, nhưng thực tế CPU có thể tăng lên đến 12 chỉ lệnh giải mã tại cùng một thời điểm. Các chỉ lệnh yêu cầu CPU để có thể đọc dữ liệu được lưu trữ tại địa chỉ RAM đã cho, Khối lưu trữ địa chỉ (Store Address Unit) và lưu trữ dữ liệu (Store Data Unit) được sử dụng tại cùng một thời điểm, một dùng cho định địa chỉ và một dùng cho đọc dữ liệu. Đây là lý do tại sao cổng 0 và cổng 1 có nhiều khối thực thi. Nếu chú ý một chút thì sẽ thấy được Intel đã đặt trên cùng một cổng cả khối “nhanh” và ít nhất cùng với một khối “chậm” (phức tạp). Chính vì vậy, trong khi khối phức tạp đang bận xử lý dữ liệu thì các khối khác có thể vẫn nhận các chỉ lệnh đã giải mã từ cổng gửi đi tương ứng của nó. Ý tưởng này là để giữ tất cả các khối thực thi luôn làm việc.

Sau mỗi một chỉ lệnh đã giải mã được thực thi, nó lại trở về bộ đệm Reorder Buffer, đây chính là nơi chờ của nó được thiết lập chế độ thực thi. Sau đó tại tầng cuối (Retirement Stage), các chỉ lệnh đã giải mã có cơ “thực thi” của chúng sẽ được xóa khỏi bộ đệm Reorder Buffer theo thứ tự ban đầu của nó (nghĩa là theo thứ tự mà chúng đã được giải mã) và sau đó các thanh ghi x86 được cập nhật (ngược lại bước của tầng đặt lại tên của thanh ghi). Có thể có đến 3 chỉ lệnh giải mã được xóa bỏ từ bộ đệm Reorder Buffer trên mỗi một chu kỳ clock. Sau đó, mỗi chỉ lệnh này được

thực thi hoàn toàn.

2.6.7 Công nghệ SpeedStep nâng cao

Công nghệ SpeedStep đã được tạo ra để tăng thời gian sống của pin và nó đã được giới thiệu đầu tiên trong các bộ vi xử lý của Pentium III, M. Phiên bản đầu tiên của công nghệ này cho phép các CPU có thể chuyển giữa hai tần số clock một cách động. Chế độ tần số thấp (LFM), chế độ cho phép thời lượng sống của pin lớn nhất, và chế độ tần số cao (HFM), chế độ cho phép chạy CPU tại tốc độ lớn nhất. CPU có hai tỉ lệ nhân clock. Tỉ lệ LFM là tỉ lệ factory-lock và ta không thể thay đổi được tỉ lệ này.

Pentium M đã giới thiệu công nghệ SpeedStep nâng cao (Enhanced SpeedStep Technology), công nghệ này là công nghệ có một vài cấu hình clock và điện áp khác giữa LFM (cố định là 600 MHz) và HFM. Một ví dụ để các ta có thể dễ hiểu hơn trong trường hợp này, bảng cấu hình clock và điện áp cho 1.6 GHz Pentium M dựa trên công nghệ 130nm:

Điện áp	Clock
1.484 V	1.6 GHz
1.42 V	1.4 GHz
1.276 V	1.2 GHz
1.164 V	1 GHz
1.036 V	800 MHz
0.956 V	600 MHz

Mỗi một mô hình của Pentium M lại có một bảng điện áp/clock của riêng nó. Cần phải chú ý một điều rằng khi không cần tốn nhiều năng lượng đối với laptop thì không những chỉ giảm tốc độ clock mà còn giảm cả điện áp, việc giảm điện áp sẽ giúp giảm tiêu tốn rất nhiều pin máy.

Công nghệ Enhanced SpeedStep làm việc bằng cách kiểm tra các thanh ghi model cụ thể MSR (Model Specific Registers) của CPU, thành phần này được gọi là Performance Counter. Với thông tin thu nhận từ bộ phận này, CPU có thể giảm hoặc tăng clock/điện áp của nó phụ thuộc vào khả năng sử dụng của CPU. Đơn giản nếu ta tăng yêu cầu sử dụng CPU thì nó sẽ tăng clock/điện áp còn nếu ta giảm hiệu suất sử dụng CPU thì nó sẽ giảm clock/điện áp.

Enhanced SpeedStep chỉ là một trong những nâng cao đã được thực hiện với kiến trúc siêu nhỏ Pentium M nhằm mục đích tăng thời lượng sử dụng của pin.

2.7 kiến trúc Core của Intel

Kiến trúc Core của Intel đã xuất hiện vào năm 2006, đây là kiến trúc được sử dụng trên tất cả các CPU mới vào thời điểm này của Intel như Merom, Conroe và Woodcrest. Kiến trúc mới này được xây dựng trên kiến trúc của Pentium M và có thêm một số tính năng mới. Thứ đầu tiên mà ta cần phải lưu ý đó là phần tên, kiến trúc Core không có liên quan gì với các CPU Core Solo và Core Duo của Intel. Core Single là một CPU Pentium M được sản xuất ở công nghệ 65 nm, còn các CPU Core Duo – trước đây được gọi là Yonah – là loại CPU dual-core công nghệ 65 nm dựa trên kiến trúc của Pentium M.

Pentium M được xây dựng trên kiến trúc thế hệ thứ 6 của Intel, kiến trúc này cũng được sử dụng trong các CPU Pentium Pro, Pentium II, Pentium III và các CPU trước đây của Celeron chứ không phải trên Pentium 4 như ta vẫn nghĩ, ý tưởng ban đầu được nhắm đến các máy tính di động. Nếu ta có thể nghĩ Pentium M là một Pentium III nâng cao thì cũng có thể nghĩ kiến trúc Core là một Pentium M nâng cao.

Kiến trúc Core sử dụng cấu trúc 14 tầng. Cấu trúc này là một danh sách tất cả các tầng mà một chỉ lệnh được cho phải trải qua khi thực thi hoàn tất. Intel đã không tiết lộ cấu trúc của Pentium M và chính vì vậy cho tới nay họ vẫn chưa công bố những chỉ dẫn của mỗi tầng trong kiến trúc Core. Pentium III đã sử dụng cấu trúc 11 tầng, Pentium 4 ban đầu có 20 tầng và các CPU Pentium 4 mới hơn dựa trên lõi “Prescott” được biết có đến 31 tầng.

2.7.1 Cache nhớ và khối tìm nạp

Hãy nhớ rằng Cache nhớ là bộ nhớ tốc độ cao (SRAM) được nhúng vào bên trong CPU, sử dụng để lưu dữ liệu mà CPU có thể cần đến. Nếu dữ liệu được yêu cầu bởi CPU không có trong Cache nhớ thì nó sẽ phải truy cập vào bộ nhớ RAM chính, điều này sẽ làm giảm tốc độ của CPU vì bộ nhớ RAM được truy cập bằng sử dụng tốc độ clock ngoài của CPU. Ví dụ, trên một CPU 3,2GHz, Cache nhớ được truy cập ở tốc độ 3,2GHz nhưng bộ nhớ RAM chính chỉ được truy cập ở tốc độ clock 800MHz. Kiến trúc Core được tạo bằng việc có khái niệm multi-core, nghĩa là có nhiều chip trên một đóng gói. Trên Pentium D, phiên bản dual-core của Pentium 4, mỗi core đều có Cache nhớ L2 của riêng nó. Vấn đề với hai Cache riêng ở đây là tại một thời điểm nào đó khi một lõi này sử dụng hết Cache nhớ trong khi lõi kia lại không sử dụng hết hiệu suất trên Cache nhớ L2 của riêng nó. Khi xảy ra điều này thì lõi đầu tiên phải truy cập và lấy dữ liệu từ bộ nhớ RAM chính, thậm chí Cache nhớ L2 của lõi thứ hai là hoàn toàn trống rỗng mà lẽ ra có thể

được sử dụng để lưu dữ liệu, tránh tình trạng lõi phải truy cập trực tiếp vào bộ nhớ RAM chính. Đối với kiến trúc Core, vấn đề này đã được giải quyết. Cache nhớ L2 được chia sẻ, có nghĩa là cả hai lõi đều có thể sử dụng Cache nhớ L2 một cách chung nhau, cấu hình động sẽ được thực hiện cho mỗi Cache. Ví dụ với một CPU có 2 MB L2 cache, một lõi có thể đang sử dụng 1,5MB còn lõi kia sử dụng 512 KB (0.5 MB), ngược lại với tỷ lệ chia cố định 50-50 như đã được sử dụng trước đây trong các CPU dual-core. Khối tiền tìm nạp được chia sẻ giữa các lõi, nghĩa là nếu hệ thống Cache nhớ đã nạp một khối dữ liệu để được sử dụng bởi lõi đầu tiên thì lõi thứ hai cũng có thể sử dụng dữ liệu đã được nạp trên Cache này rồi. Trong các kiến trúc trước, nếu lõi thứ hai cần dữ liệu giống như dữ liệu đã được nạp vào Cache của lõi đầu tiên thì nó vẫn phải truy cập thông qua bus ngoài (điều đó khiến CPU làm việc ở tốc độ clock ngoài, có tốc độ clock thấp hơn tốc độ clock trong) hoặc thậm chí lấy dữ liệu cần thiết trực tiếp từ bộ nhớ RAM của hệ thống. Intel cũng đã cải thiện khối tiền tìm nạp của CPU, đưa ra các mẫu theo cách mà CPU hiện đang lấy dữ liệu từ bộ nhớ để đoán thử dữ liệu mà CPU sẽ tìm nạp tiếp theo là gì và nạp nó vào Cache nhớ trước khi CPU yêu cầu. Ví dụ, nếu CPU đã nạp dữ liệu từ địa chỉ 1, sau đó yêu cầu dữ liệu trên địa chỉ 3 và sau đó yêu cầu tiếp dữ liệu trên địa chỉ 5 thì khối tiền tìm nạp sẽ đoán rằng chương trình sẽ nạp dữ liệu từ địa chỉ 7 và nó sẽ nạp từ địa chỉ này ra Cache nhớ trước khi CPU yêu cầu đến nó. Quả thực ý tưởng này không có gì mới mẻ và tất cả các CPU từ Pentium Pro sẽ dùng một số kiểu dự đoán để cung cấp Cache nhớ L2. Trên kiến trúc Core, Intel đã có một chút nâng cao về tính năng này bằng cách tạo ra một khối tiền tìm nạp tìm kiếm các mẫu trong dữ liệu tìm nạp thay vì các bộ chỉ thị tĩnh của dữ liệu mà CPU sẽ yêu cầu tiếp theo.

2.7.2 Bộ giải mã chỉ lệnh: Macro-Fusion

Một khái niệm mới được giới thiệu trong kiến trúc Core đó là macro-fusion. Macro-fusion là khả năng gắn (joining) hai chỉ lệnh x86 vào thành một chỉ lệnh micro-op. Cách làm này có thể cải thiện được hiệu suất của CPU và tiêu tốn ít năng lượng của CPU hơn vì nó sẽ chỉ thực thi một chỉ lệnh micro-op thay vì hai. Mặc dù vậy cơ chế này lại bị hạn chế đối với các chỉ lệnh so sánh và các chỉ lệnh rẽ nhánh có điều kiện (có nghĩa là các chỉ lệnh CMP và Jcc). Ví dụ, chúng ta hãy xem đoạn chương trình dưới đây:

```
...  
load eax, [mem1]  
cmp eax, [mem2]
```

jne target

...

Đoạn chương trình này sẽ thực hiện nạp thanh ghi 32 bit EAX bằng dữ liệu được chứa trong vị trí nhớ 1, so sánh giá trị của nó với dữ liệu có trong vị trí nhớ 2 và nếu chúng khác nhau thì (jne = jump if not equal), thì chương trình sẽ truy cập vào địa chỉ “target”, còn nếu bằng nhau thì chương trình sẽ tiếp tục trên vị trí hiện hành. Với macro-fusion, các chỉ lệnh so sánh (cmp) và rẽ nhánh (jne) sẽ được hợp nhất vào một chỉ lệnh micro-op. Chính vì vậy sau khi chuyển qua bộ giải mã chỉ lệnh, phần chương trình này sẽ giống như dưới đây:

...

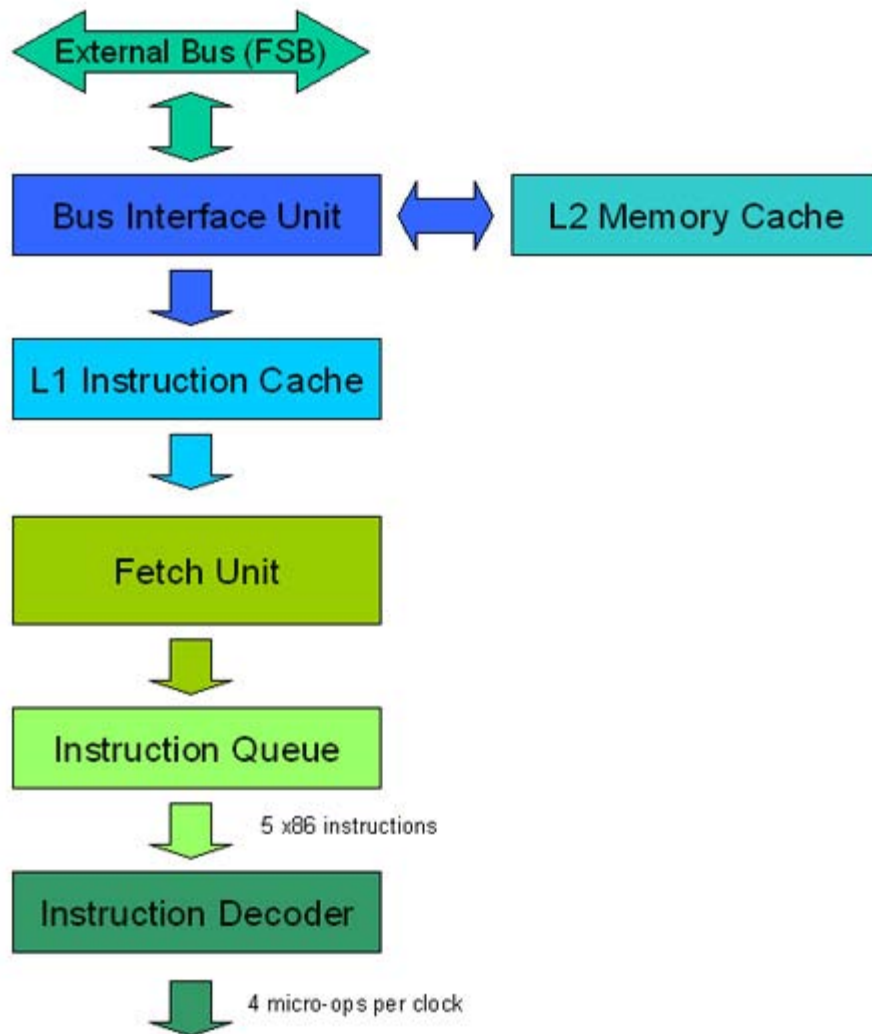
load eax, [mem1]

cmp eax, [mem2] + jne target

...

Như những gì thấy ở trên, chúng ta đã lưu một chỉ lệnh. Càng ít chỉ lệnh được thực thi thì máy tính của ta sẽ thực hiện việc thực thi nhiệm vụ nhanh hơn và tốn ít công suất tiêu thụ hơn. Bộ giải mã chỉ lệnh có trên kiến trúc Core có thể giải mã 4 chỉ lệnh trên một chu kỳ clock, trong khi đó ở các CPU trước như Pentium M và Pentium 4 thì chỉ có thể giải mã được đến 3. Ở đây bộ giải mã chỉ lệnh của kiến trúc Core kéo đến 5 chỉ lệnh mỗi lần vào hàng đợi chỉ lệnh, thậm chí nó còn có thể giải mã đến 4 chỉ lệnh trên một chu kỳ clock. Chính vì vậy nếu hai trong số 5 chỉ lệnh được nối thành một thì bộ giải mã vẫn có thể giải mã bốn chỉ lệnh trên một chu kỳ clock. Và nó sẽ ở chế độ nhàn rỗi cục bộ bất cứ khi nào macro-fusion xảy ra, nghĩa là bộ giải mã sẽ chỉ cung cấp ba chỉ lệnh nối micro-op ở đầu ra của nó trong khi có khả năng cung cấp đến bốn.

Trong hình 2.7.1 bên dưới ta có thể thấy những thông tin tóm tắt đã giải thích ở trên.



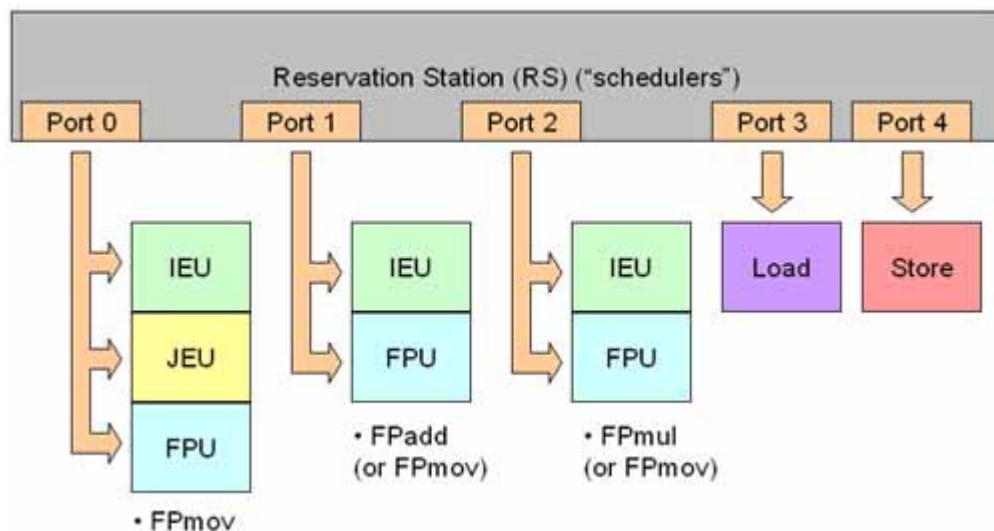
Hình 2.7.2: Khối tìm nạp và bộ giải mã chỉ lệnh trong kiến trúc Core

2.7.3 Khối thực thi

Pentium M có 5 cổng gửi đi được đặt trên trạm dành riêng Reservation Station của nó, nhưng chỉ có hai cổng được sử dụng để gửi đi các chỉ lệnh micro-ops đến các khối thực thi. Ba chỉ lệnh còn lại được sử dụng bởi các khối có liên quan đến bộ nhớ (Load, Store Address và Store Data). Kiến trúc Core cũng có 5 cổng gửi đi như vậy nhưng ba trong số chúng được sử dụng cho việc gửi các chỉ lệnh nổi micro-ops đến các khối thực thi. Điều đó có nghĩa rằng các CPU đang sử dụng kiến trúc Core đó có thể gửi ba chỉ lệnh micro-ops đến khối thực thi trên một chu kỳ clock. Kiến trúc Core cung cấp một FPU mở rộng và một IEU mở rộng (ALU) khi chúng ta mang ra so với kiến trúc Pentium M. Điều này có nghĩa rằng kiến trúc Core có thể xử lý đến ba chỉ lệnh số nguyên trên một chu kỳ clock, trong khi Pentium M chỉ có

hai. Tuy nhiên không phải tất cả các chỉ lệnh toán học đều có thể được thực thi trên tất cả các FPU. Như những gì ta có thể quan sát được trong hình 2.7.3, các toán tử nhân floating-point chỉ có thể được thực thi trong FPU thứ ba và phần thêm vào floating-point chỉ có thể được thực thi trên FPU thứ hai. Các chỉ lệnh Fpmov có thể được thực thi trên FPU thứ nhất hoặc trên hai FPU khác nếu không có chỉ lệnh phức tạp hơn (FPadd or FPMul) đã sẵn sàng được gửi đến chúng. Các chỉ lệnh MMX/SSE đều được xử lý bởi FPU.

Trong hình 2.7.3 ta sẽ thấy sơ đồ khối chính của các khối thực thi trong kiến trúc Core.



Hình 2.7.3: Các khối thực thi trong kiến trúc Core

Một sự khác nhau lớn giữa hai kiến trúc Pentium M và Pentium 4 với kiến trúc Core là trên kiến trúc Core, các khối Load và Store có khối tạo địa chỉ của riêng nó nhúng trong. Pentium 4 và Pentium M có các khối tạo địa chỉ riêng và trên Pentium 4 ALU đầu tiên được sử dụng để lưu dữ liệu trên bộ nhớ.

Dưới đây là những giải thích nhỏ về những khối có trong CPU này:

- IEU: Instruction Execution Unit là nơi các chỉ lệnh được thực thi. Khối này cũng được biết đến là khối ALU (Arithmetic and Logic Unit). Các chỉ lệnh thông thường cũng được biết là các chỉ lệnh số nguyên.
- JEU: Jump Execution Unit xử lý rẽ nhánh và cũng được biết đến với tên Branch Unit.
- FPU: Floating-Point Unit. Khối này chịu trách nhiệm cho việc thực thi các biểu thức toán học floating-point và cũng cả các chỉ lệnh MMX và

SSE. Trong CPU này, các FPU không “hoàn thiện” vì một số kiểu chỉ lệnh (FPmov, FPadd và FPMul) chỉ được thực thi trên các FPU nào đó:

- FPadd: Chỉ có FPU này mới có thể xử lý các chỉ lệnh cộng floating-point như ADDPS.
- FPMul: Chỉ có FPU này mới có thể xử lý các chỉ lệnh nhân floating-point như MULPS
- FPmov: Các chỉ lệnh cho việc nạp hoặc copy một thanh ghi FPU, như MOVAPS (được dùng để truyền tải dữ liệu đến thanh ghi SSE 128-bit XMM). Kiểu chỉ lệnh này có thể được thực thi trên các FPU, nhưng chỉ trên các FPU thứ hai và thứ ba nếu các chỉ lệnh FPadd hay FPMul không có trong Reservation Station.
- Load: khối này dùng để xử lý các chỉ lệnh yêu cầu dữ liệu được đọc từ bộ nhớ RAM.
- Store Data: Khối này xử lý các chỉ lệnh yêu cầu dữ liệu được ghi vào bộ nhớ RAM.

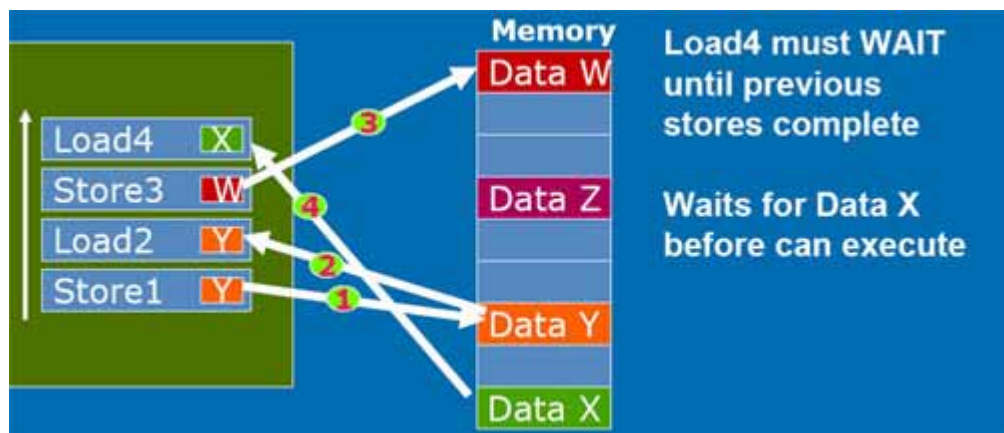
Lưu ý: rằng các chỉ lệnh phức tạp có thể mất đến một số chu kỳ clock trong khi xử lý. Chúng ta hãy lấy một ví dụ về cổng 2, nơi khối FPMul nằm ở đó. Khi khối này đang xử lý một chỉ lệnh rất phức tạp phải mất đến vài chu kỳ clock để được thực thi thì cổng 2 sẽ không chết: nó sẽ luôn gửi các chỉ lệnh đơn giản đến IEU trong khi FPU đang bận.

2.7.4 Đường dẫn 128-bit bên trong và kiến trúc nhớ mới

Một tính năng khác có trong kiến trúc Core là đường dẫn dữ liệu 128 bit bên trong. Trong các CPU trước, đường dẫn dữ liệu bên trong chỉ có 64bit. Đây là một vấn đề đối với các chỉ lệnh SSE, chỉ lệnh được gọi là XMM có dài 128 bit. Chính vì vậy khi thực thi một chỉ lệnh đã biến đổi thành 128 bit dữ liệu thì toán tử này được chia thành hai toán tử 64bit. Đường dữ liệu 128 bit mới làm cho kiến trúc Core trở nên nhanh hơn trong việc xử lý các chỉ lệnh SSE có 128 bit dữ liệu. Kiến trúc nhớ mới là kỹ thuật tăng tốc thực thi các chỉ lệnh có liên quan đến bộ nhớ. Tất cả các CPU của Intel từ Pentium Pro đều có cơ chế không tuân theo trình tự (out-of-order), cơ chế này cho phép CPU có thể thực thi các chỉ lệnh không phụ thuộc theo bất cứ một thứ tự nào. Những gì xảy ra với các chỉ lệnh liên quan đến bộ nhớ được thực thi theo kiểu truyền thống diễn ra theo một thứ tự giống hệt với thứ tự chúng xuất hiện trong chương trình. Ví dụ, nếu chương trình gốc có một chỉ lệnh như “store 10 at address 5555” và sau đó là một chỉ lệnh “load data stored at 5555”, thì chúng sẽ không thể được đảo ngược (nghĩa là được thực thi không tuân theo thứ tự) hoặc chỉ lệnh thứ hai sẽ lấy sai dữ liệu, vì dữ liệu ở địa chỉ

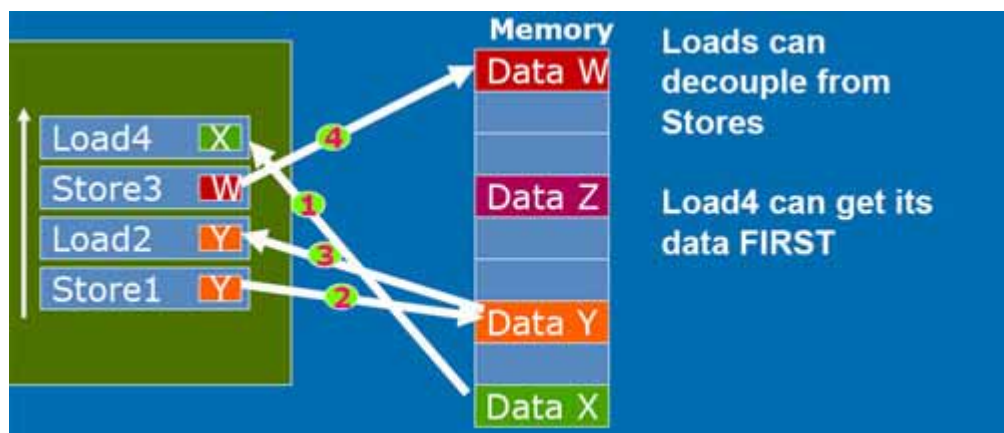
5555 đã bị thay đổi bởi chỉ lệnh thứ nhất. Những gì mà cơ chế kiến trúc nhớ mới thực hiện là định vị và thực thi các chỉ lệnh có liên quan đến bộ nhớ để có thể thực thi không theo thứ tự, tăng tốc độ thực thi của chương trình (chúng tôi sẽ giải thích thêm về điều này được thực hiện như thế nào).

Trong hình 2.7.4, ta có một ví dụ về một CPU không có cơ chế nhớ này (nghĩa là tất cả các CPU không được xây dựng trên kiến trúc Core). Như những gì ta có thể nhìn thấy, CPU phải thực thi các chỉ lệnh khi chúng xuất hiện trong chương trình gốc. Ví dụ, chỉ lệnh “Load4” không liên quan tới bất kỳ dữ liệu nào và có thể được thực thi trước, mặc dù vậy nó vẫn phải đợi tất cả các chỉ lệnh khác.



Hình 2.7.4: CPU không có kiến trúc nhớ mới

Trong hình 2.7.5, ta sẽ thấy cách chương trình trong hình 2.7.4 làm việc như thế nào trên CPU có kiến trúc Core. Nó “biết” rằng chỉ lệnh “Load4” không có liên quan đến các chỉ lệnh khác và có thể được thực thi trước.



Hình 2.7.5: CPU với cơ chế nhớ mới.

Điều này đã cải thiện được hiệu suất của CPU vì lúc này chỉ lệnh “Load4” sẽ được thực thi ngay từ đầu, CPU có dữ liệu cần thiết cho việc thực thi các chỉ

lệnh khác cần đến giá trị “X” để được thực thi.

Trong các CPU thông thường, nếu sau khi chỉ lệnh “Load4” này có chỉ lệnh “Add 50”, thì chỉ lệnh “Add 50” (và tất cả các chỉ lệnh khác phụ thuộc vào kết quả đó) sẽ phải đợi các chỉ lệnh khác như đã thể hiện trong hình 2.7.4 để được thực thi. Với kiến trúc nhớ mới này, các chỉ lệnh đó có thể được thực thi sớm, vì CPU lúc này sẽ có được giá trị “X” từ sớm.

2.7.5 Điều chỉnh tiết lưu công suất

Với việc điều chỉnh tiết lưu công suất tiên tiến, kiến trúc Core làm tiết kiệm được nhiều năng lượng tiêu thụ hơn so với các CPU trước đó. Tính năng này cho phép CPU có thể tắt các khối đang không được sử dụng ở thời điểm đó. Ý tưởng này thậm chí còn cho phép nhiều ưu việt hơn vì khi CPU có thể tắt các phân cụ thể bên trong mỗi khối CPU để tiết kiệm năng lượng, tốn ít công suất và cải thiện được thời gian sử dụng của pin (trong trường hợp xét đến các CPU di động).

Một khả năng tiết kiệm năng lượng khác của kiến trúc Core là chỉ bật các bit cần thiết trong các bus bên trong CPU. Nhiều bus bên trong của CPU được kích thước một cách cồng kềnh và lãng phí. Chính vì vậy thay cho việc bật tất cả ví dụ 480 làn dữ liệu của một bus nào đó thì CPU có thể chỉ cần bật 32 làn dữ liệu của nó, tất cả các dữ liệu trong làn đó đều cần thiết cho việc truyền tải chỉ lệnh 32bit.

* Có thể hơi khó hiểu ở tuyên bố này, vì ta vẫn thường nghe thấy rằng kiến trúc của Intel sử dụng các chỉ lệnh 32bit, vì vậy chúng tôi cần đưa ra giải thích này để làm sáng tỏ vấn đề. Bên trong CPU, những gì được xem xét ở một chỉ lệnh là mã thao tác (opcode) của chỉ lệnh đó (ngôn ngữ máy tương đương với ngôn ngữ chỉ lệnh assembly) cộng với tất cả các dữ liệu được yêu cầu. Điều này là vì để được thực thi, chỉ lệnh phải nhập vào cơ chế thực thi “hoàn tất”, nghĩa là cùng với tất cả các dữ liệu được yêu cầu. Cũng theo cách đó, kích thước của mỗi mã thao tác chỉ lệnh x86 là một biến và không cố định là 32bit như những gì ta vẫn nghĩ. Ví dụ, một chỉ lệnh “mov eax, (32-bit data)”, dùng để lưu (32-bit data) và thanh ghi EAX của CPU được xem xét bên trong như một chỉ lệnh 40bit (mov eax dịch vào 8-bit opcode cộng với 32bit dữ liệu). Việc các chỉ lệnh có số chiều dài khác nhau là những gì đặt trưng cho tập chỉ lệnh CISC (Complex Instruction Set Computing).

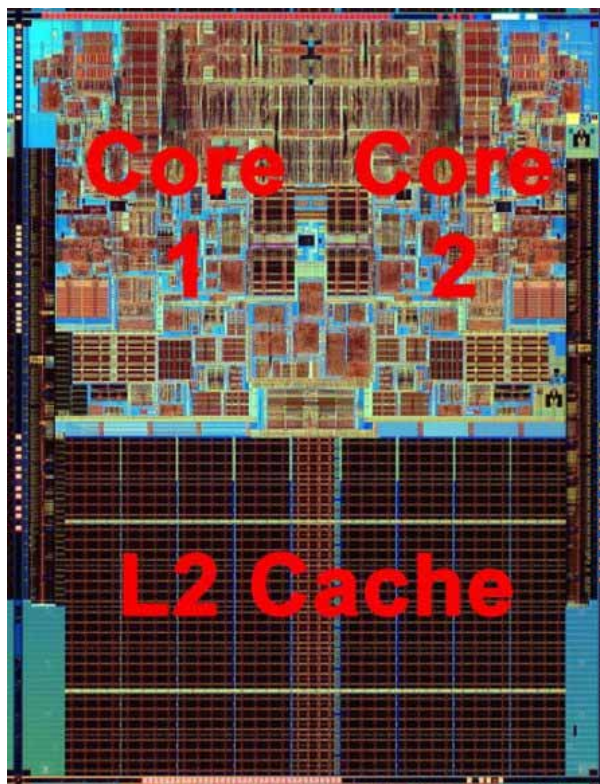
2.8 Các mô hình Core 2 Duo

các model của Core 2 Duo, Core 2 Quad và Core 2 Extreme cho tới thời điểm này và sẽ giới thiệu thêm về một số tính năng chính của chúng.

Dưới đây là những tóm tắt về các tính năng chính của họ Core 2:

- Kiến trúc lõi
- Cache nhớ chỉ lệnh L1 32KB và Cache nhớ dữ liệu L1 32KB cho mỗi lõi.
- Công nghệ Dual-core hoặc công nghệ quad-core.
- Quá trình sản xuất 65 nm
- Socket 775.
- 800 MHz (200 MHz x 4), tốc độ clock ngoài 1.066 MHz (266 MHz x 4) hoặc 1.333 MHz (333 MHz x 4).
- Cache nhớ hợp nhất L2 2 MB, 4 MB hoặc 8 MAILBOX.
- Công nghệ ảo của Intel (ngoại trừ Core 2 Duo E4300)
- Công nghệ Intel EM64T.
- Tập chỉ lệnh SSE3.
- Execute Disable Bit
- Khả năng xử lý công suất tiêu thụ thông minh
- Công nghệ Enhanced SpeedStep.

Trên hình 2.7.1 ta có thể thấy được một bức tranh về chân đế của Core 2 Duo CPU.



Hình 2.7.1: Chân đế của bộ vi xử lý Core 2

Các model

Trong bảng dưới đây chúng tôi sẽ liệt kê tất cả các model của Core 2 Duo đã được phát hành cho tới gần thời điểm này.

Chi tiết kỹ thuật	Model	Clock trong	Clock ngoài	L2 Cache	Số lượng transistor	Kích thước chân	TDP	Nhiệt độ tối đa (°C)	Điện áp
SLA9U	E6850	3 GHz	1,333 MHz	4 MB	291 million	143 mm2	65 W	72	0.962V-1.35V
SLA9V	E6750	2.66 GHz	1,333 MHz	4 MB	291 million	143 mm2	65 W	72	0.962V-1.35V
SL9ZF	E6700	2.66 GHz	1,066 MHz	4 MB	291 million	143 mm2	65 W	60.1	-
SL9S7	E6700	2.66 GHz	1,066 MHz	4 MB	291 million	143 mm2	65 W	60.1	0.85V-1.35V
SL9ZL	E6600	2.40 GHz	1,066 MHz	4 MB	291 million	143 mm2	65 W	60.1	1.18V-1.32V
SL9S8	E6600	2.40 GHz	1,066 MHz	4 MB	291 million	143 mm2	65 W	60.1	0.85V-1.35V
SLA9X	E6550	2.33 GHz	1,333 MHz	4 MB	291 million	143 mm2	65 W	72	0.962V-1.35V
SLAAX	E6540	2.33 GHz	1,333 MHz	4 MB	291 million	143 mm2	65 W	72	0.962V-1.35V
SL94T	E6420	2.13 GHz	1,066 MHz	4 MB	291 million	143 mm2	65 W	60.1	-

SL9T9	E6400	2.13 GHz	1,066 MHz	2 MB	167 million	111 mm2	65 W	61.4	1.22V-1.32V
SL9S9	E6400	2.13 GHz	1,066 MHz	2 MB	167 million	111 mm2	65 W	61.4	0.85V-1.35V
SLA4U	E6320	1.86 GHz	1,066 MHz	4 MB	291 million	143 mm2	65 W	60.1	-
SL9TA	E6300	1.86 GHz	1,066 MHz	2 MB	167 million	111 mm2	65 W	61.4	1.22V-1.32V
SL9SA	E6300	1.86 GHz	1,066 MHz	2 MB	167 million	111 mm2	65 W	61.4	0.85V-1.35V
SLA95	E4500	2.20 GHz	800 MHz	2 MB	167 million	111 mm2	65 W	73.3	0.962 V-1.35V
SL93F	E4400	2 GHz	800 MHz	2 MB	167 million	111 mm2	65 W	61.4	1.16V-1.31V
SLA98	E4400	2 GHz	800 MHz	2 MB	167 million	111 mm2	65 W	73.3	1.16V-1.31V
SL9TB	E4300	1.8 GHz	800 MHz	2 MB	167 million	111 mm2	65 W	61.4	0.85V-1.35V

Trong bảng dưới đây là liệt kê của các model Core 2 Quad.

Chi tiết kỹ thuật	Model	Clock trong	Clock ngoài	L2 Cache	TDP	Nhiệt độ tối đa (°C)	Điện áp	Số nhân
SLACQ	Q6700	2.66 GHz	1,066 MHz	8 MB	95	71	1.10V-1.37V	4
SL9UM	Q6600	2.4 GHz	1,066 MHz	8 MB	105 W	62,2	1.10V-1.37V	4

SL9UM	Q6600	2.4 GHz	1,066 MHz	8 MB	105 W	62,2	1.10V-1.37V	4
-------	-------	---------	-----------	------	-------	------	-------------	---

Trong bảng dưới đây là các model Core 2 Extreme đã được phát hành.

Chi tiết kỹ thuật	Model	Clock trong	Clock ngoài	L2 Cache	TDP	Nhiệt độ tối đa (°C)	Điện áp	Số nhân
SLAFN	QX6850	3 GHz	1,333 MHz	8 MB	130 W	64.5	1.10V-1.37V	4
SL9UK	QX6800	2.93 GHz	1,066 MHz	8 MB	130 W	64.5	1.10V-1.37V	4
SL9S5	X6800	2.93 GHz	1,066 MHz	4 MB	75 W	60.4	0.85V-1.35V	2
SLACP	QX6800	2.93 GHz	1,066 MHz	8 MB	130 W	64.5	1.10V-1.37V	4
SLA33	X7900	2.80 GHz	800 MHz	4 MB	44 W	100	-	2
SLAF4	X7900	2.80 GHz	800 MHz	4 MB	44 W	100	1.125V-1.325V	2
SL9UL	QX6700	2.66 GHz	1,066 MHz	8 MB	130 W	65	1.10V-1.37V	4
SLA6Z	X7800	2.60 GHz	800 MHz	4 MB	44 W	100	-	2

Bảng dưới đây là những gì chúng tôi tổng hợp từ các chi tiết kỹ thuật của các thế hệ Centrino:

Nền tảng	Centrino Pro	Centrino Duo	Centrino Duo	Centrino	Centrino	Centrino
Tên mã	Santa Rosa	Santa Rosa	Napa	Napa	Sonoma	Carmel
Bộ vi xử lý	Core 2 Duo	Core 2 Duo	Core Duo (Yonah)	Core Solo	Pentium M	Pentium M

			Core 2 Duo (Meron)		(Dothan)	(Banias)
Chipset	Intel 965 Express	Intel 965 Express	Intel 945 Express	Intel 945 Express	Intel 915 Express	Intel 855
Wireless	Network Intel PRO / Wireless 4965AGN	Intel PRO / Wireless 4965AGN	Intel PRO / Wireless 3945ABG	Intel PRO / Wireless 3945ABG	Intel PRO / Wireless 2200BG Intel PRO / Wireless 2915ABG	Intel PRO / Wireless 2100

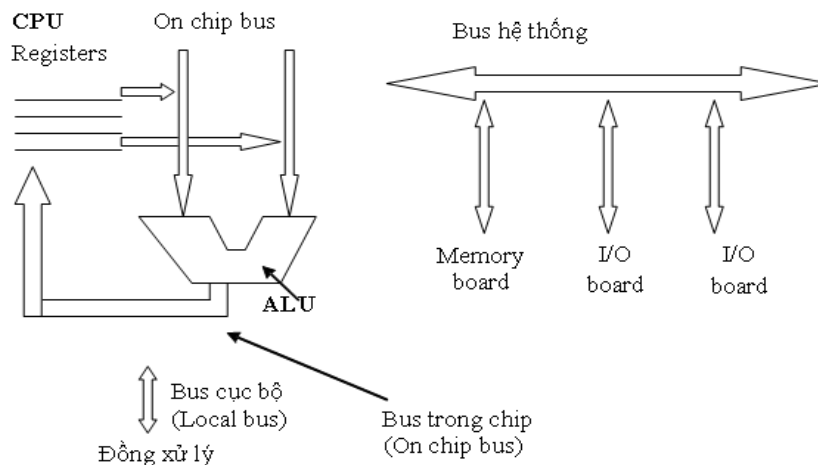
CHƯƠNG 2:

BUS VÀ TRUYỀN THÔNG TIN TRONG MÁY TÍNH

2.1.1. Định nghĩa BUS:

Bus là đường truyền tín hiệu điện nối các thiết bị khác nhau trong một hệ thống máy tính. Bus có nhiều dây dẫn được gắn trên mainboard, trên các dây này có các đầu nối đưa ra, các đầu này được sắp xếp và cách nhau những khoảng quy định để có thể cắm vào đó những I/O board hay board bộ nhớ (bus hệ thống – system bus).

2.1.2. Phân loại các Bus trong hệ thống:



Các Bus trong hệ thống máy tính

Cũng có những bus dùng cho mục đích chuyên biệt, thí dụ nối 1 vi xử lý với 1 hay nhiều vi xử lý khác hoặc nối với bộ nhớ cục bộ (local bus).

Trong vi xử lý cũng có một số bus để nối các thành phần bên trong của bộ vi xử lý với nhau. Người thiết kế chip vi xử lý có thể tùy ý lựa chọn loại bus bên trong nó, còn với các bus liên hệ bên ngoài cần phải xác định rõ các quy tắc làm việc cũng như các đặc điểm kỹ thuật về điện và cơ khí của bus để người thiết kế mainboard có thể ghép nối chip vi xử lý với các thiết bị khác. Một số bus được sử dụng phổ biến:

Tên Bus	Lĩnh vực áp dụng
Camac	Vật lý hạt nhân
EISA	Một số hệ thống có chip 80386
IBM PC, PC/AT	Máy IBM/PC, IBM/PC/AT
Massbus	Máy PDP – 11, và VAX
Microchannel	Máy PS/2
Multibus I	Một số hệ thống 8086

Multibus II	Một số hệ thống có chip 80386
Versabus	Một số hệ thống có chip xử lý của Motorola
VME	Một số hệ thống có chip xử lý họ 68x0 của Motorola

Nói cách khác, các bus này phải tuân theo 1 chuẩn nào đó. Tập các quy tắc của chuẩn còn được gọi là giao thức bus (bus protocol)

Ngoài ra, có rất nhiều loại bus khác nhau được sử dụng, các bus này nói chung là không tương thích với nhau.

Bus thường phân loại theo 3 cách sau:

- Theo tổ chức phân cứng (như trên)
- Theo giao thức truyền thông (bus đồng bộ và không đồng bộ)
- Theo loại tín hiệu truyền trên bus (bus địa chỉ, bus dữ liệu,...)

2.1.3. Bus hệ thống:

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

Ví dụ: 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, bộ điều khiển đĩa ra lệnh cho bộ nhớ nhận dữ liệu thì nó lại giữ vai trò master.

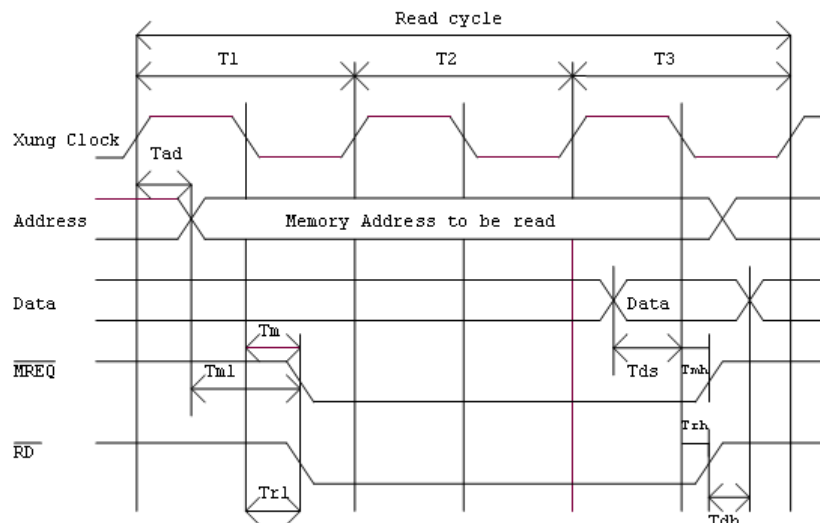
2.1.3 Bus Driver và Bus Receiver:

Tín hiệu điện trong máy tính phát ra thường không đủ để điều khiển bus, nhất là khi bus khá dài và có nhiều thiết bị nối với nó. Chính vì thế mà hầu hết các bus master được nối với bus thông qua 1 chip gọi là *bus driver*, về cơ bản nó là một bộ khuếch đại tín hiệu số. Tương tự như vậy, hầu hết các slave được nối với bus thông qua *bus receiver*. Đối với các thiết bị khi thì đóng vai trò master, khi thì đóng vai trò slave, người ta sử dụng 1 chip kết hợp gọi là *transceiver*. Các chip này đóng vai trò ghép nối và là các thiết bị 3 trạng thái, cho phép nó có thể ở trạng thái thứ 3 – hở mạch (thả nổi). Giống như vi xử lý, bus có các đường địa chỉ, đường số liệu và đường điều khiển. Tuy nhiên, không nhất thiết có ánh xạ 1 – 1 giữa các tín hiệu ở các chân ra của vi xử lý và các đường dây của bus. Thí dụ: một số chip vi xử lý có 3 chân ra, truyền ra các tín hiệu báo chip vi xử lý đang thực hiện các thao tác MEMR, MEMW, IOR, IOW hay thao tác khác, một số Bus điển hình có các đường trên. Các vấn đề quan trọng nhất liên quan đến thiết kế bus là: xung clock bus (sự phân chia thời gian, hay còn gọi là bus blocking), cơ chế phân xử bus (bus arbitration), xử lý ngắt và xử lý lỗi.

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

2.1.4 Bus đồng bộ (Synchronous bus):

Bus đồng bộ có một đường 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 $5\text{MHz} \div 50\text{MHz}$. Mọi hoạt động bus xảy ra trong một số nguyên lần chu kỳ này và được gọi là chu kỳ bus.



Chu kỳ đọc trong Bus đồng bộ

Hình trên là giản đồ thời gian của một bus đồng bộ với tần số xung clock là 4MHz , như vậy chu kỳ bus là 250ns .

Giả sử đọc 1 byte từ bộ nhớ chiếm 3 chu kỳ bus (750ns), tương ứng với T1, T2, T3 như hình vẽ. Vì tất cả các tín hiệu điện thay đổi mức không phải là tức thời, nên trên hình vẽ có các sườn xung, ta giả sử các sườn xung kéo dài 10ns .

- T1 bắt đầu bằng cạnh dương của xung clock, trong một phần thời gian của T1, vi xử lý đặt địa chỉ byte cần đọc lên bus địa chỉ. Sau khi tín hiệu địa chỉ được xác lập, vi xử lý đặt các tín hiệu MREQ và RD tích cực mức thấp, tín hiệu MREQ (Memory Request) - xác định truy xuất bộ nhớ chứ không phải thiết bị I/O, còn tín hiệu RD - chọn đọc chứ không phải ghi dữ liệu.
- T2: thời gian cần thiết để bộ nhớ giải mã địa chỉ và đưa dữ liệu lên bus dữ liệu.
- T3: tại cạnh âm của T3, vi xử lý nhận dữ liệu trên bus dữ liệu, chứa vào thanh ghi bên trong vi xử lý và chốt dữ liệu. Sau đó vi xử lý đảo các tín hiệu MREQ và RD.

Như vậy thao tác đọc đã hoàn thành, tại chu kỳ máy tiếp theo vi xử lý có thể thực hiện thao tác khác. Các giá trị cụ thể về thời gian của hình vẽ trên có thể được giải thích chi tiết như sau:

- T_{AD} : $T_{AD} < 110\text{ns}$, nghĩa là nhà sản xuất vi xử lý đảm bảo rằng trong mọi chu kỳ đọc toán hạng từ bộ nhớ, vi xử lý sẽ đưa ra tín hiệu địa chỉ không nhiều hơn 110 ns tính từ thời điểm cạnh dương của T1.
- T_{DS} : giá trị nhỏ nhất là 50ns, có nghĩa là nhà sản xuất bộ nhớ phải đảm bảo rằng dữ liệu đã ổn định trên bus dữ liệu ít nhất là 50ns trước điểm giữa cạnh âm của T3. Yêu cầu này đảm bảo cho vi xử lý đọc dữ liệu tin cậy.

Khoảng thời gian bắt buộc đối với T_{AD} và T_{DS} xác định rằng trong trường hợp xấu nhất, bộ nhớ chỉ có $250 + 250 + 125 - 110 - 50 = 465$ ns tính từ thời điểm có tín hiệu địa chỉ cho tới khi tạo ra dữ liệu trên bus dữ liệu. Nếu bộ nhớ không có khả năng đáp ứng đủ nhanh, nó phát tín hiệu WAIT trước cạnh âm của T2. Thao tác này đưa thêm các trạng thái chờ – wait state (tức là đưa thêm vào 1 chu kỳ bus), khi bộ nhớ đã đưa ra tín hiệu ổn định, nó sẽ đảo WAIT thành WAIT

- T_{ML} : đảm bảo tín hiệu địa chỉ sẽ được xác lập trước tín hiệu MREQ ít nhất 60ns. Khoảng thời gian này sẽ quan trọng nếu tín hiệu MREQ điều khiển quá trình tạo tín hiệu chọn chip CS hay CE do 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 75ns.
- T_M, T_{RL} cho phép hai tín hiệu MREQ và RD tích cực trong khoảng thời gian 85ns tính từ thời điểm xuống của xung clock T1. Trong trường hợp xấu nhất, chip nhớ chỉ có $250 + 250 - 85 - 50 = 365\text{ns}$ sau khi 2 tín hiệu trên tích cực để đưa dữ liệu ra bus dữ liệu. Sự bắt buộc về thời gian này bổ sung thêm sự bắt buộc thời gian với tín hiệu clock.
- T_{MH}, T_{RH} : thời gian để các tín hiệu MREQ và RD được đảo sau khi dữ liệu đã được vi xử lý nhận vào.
- T_{DH} : Thời gian bộ nhớ cần giữ data trên bus sau khi tín hiệu RD đã đảo.

Giản đồ thời gian một chu kỳ đọc trên bus đồng bộ đã được đơn giản hoá so với thực tế, trong đó các tín hiệu cần sử dụng lớn hơn nhiều. Giá trị tới hạn của các thông số cho trong bảng sau:

Ký hiệu	Tham số	Min (ns)	Max (ns)
T_{AD}	Thời gian trễ của địa chỉ		100
T_{ML}	Thời gian địa chỉ ổn định trước MREQ bù	60	

T_M	Thời gian trễ của MREQ so với cạnh âm của T1		85
T_{RL}	Thời gian trễ của RD bù so sườn xuống của tín hiệu đồng bộ T1		85
T_{DS}	Thời gian thiết lập dữ liệu trước sườn xuống của tín hiệu xung clock(tín hiệu đồng hồ)	50	
T_{MH}	Thời gian trễ của MREQ bù so sườn xuống của tín hiệu đồng hồ T3		85
T_{RH}	Thời gian trễ của RD bù so sườn xuống của tín hiệu đồng hồ T3		85
T_{DH}	Thời gian lưu trữ dữ liệu từ lúc đảo tín hiệu RD bù	0	

Truyền theo khối:

Ngoài các chu kỳ đọc/ghi, một số bus truyền dữ liệu đồng bộ còn hỗ trợ truyền dữ liệu theo khối. Khi bắt đầu thao tác đọc khối, bus master báo cho slave biết số byte cần được truyền đi, thí dụ truyền con số này đi trong chu kỳ T1, sau đó đáng lẽ truyền đi 1 byte, slave đưa ra trong mỗi chu kỳ 1 byte cho tới khi đủ số byte được thông báo. Như vậy, khi đọc dữ liệu theo khối, n byte dữ liệu cần n+2 chu kỳ clock chứ không phải 3n chu kỳ.

Một cách khác để cho truyền dữ liệu nhanh hơn là giảm chu kỳ. Ở ví dụ trên: 1 byte được truyền đi trong 750ns, vậy bus có tốc độ truyền 1.33MBps. Nếu xung clock có tần số 8MHz, thời gian 1 chu kỳ chỉ còn một nửa, tốc độ sẽ là 2.67MBps. Tuy nhiên, giảm chu kỳ bus dẫn đến khó khăn về mặt kỹ thuật, các tín hiệu truyền trên các đường khác nhau không phải luôn có cùng tốc độ, dẫn đến hiệu ứng *bus skew*. Điều quan trọng là thời gian chu kỳ phải dài hơn so với skew để tránh việc những khoảng thời gian được số hoá lại trở thành các đại lượng biến thiên liên tục.

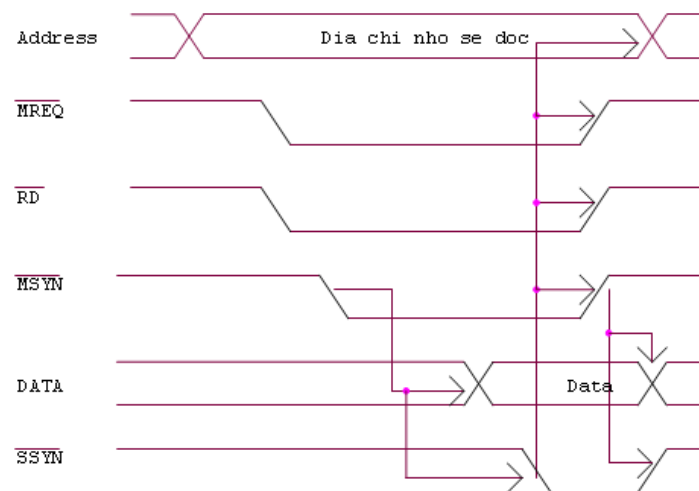
2.1.5 Bus bất đồng bộ(Asynchronous bus)

Bus bất đồng bộ không sử dụng xung clock đồng bộ, chu kỳ của nó có thể kéo dài tùy ý và có thể khác nhau đối với các cặp thiết bị khác nhau. Làm việc với các 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. Mọi công việc được tiến hành trong khoảng thời gian là bội số của xung clock, nếu 1 thao tác nào đó của vi xử lý hay bộ nhớ hoàn thành trong 3.1 chu kỳ thì nó cũng sẽ phải kéo dài trong 4 chu kỳ. Khi đã chọn chu kỳ bus và đã xây dựng bộ nhớ, I/O card cho bus này thì khó có thể tận dụng những tiến bộ của công nghệ. Chẳng hạn sau khi đã xây bus với sự định thời như trên,

công nghệ mới đưa ra các vi xử lý và bộ nhớ có thời gian chu kỳ là 100ns chứ không còn là 750ns như cũ, thì chúng vẫn chạy với tốc độ thấp như các vi xử lý, bộ nhớ loại cũ, bởi vì giao thức bus đòi hỏi bộ nhớ phải đưa được dữ liệu ra và ổn định trước thời điểm cạnh âm của T3. Nếu có nhiều thiết bị khác nhau cùng nối với 1 bus, trong đó có thể có một số thiết bị hoạt động nhanh hơn các thiết bị khác thì cần phải đặt bus hoạt động phù hợp với thiết bị có tốc độ thấp nhất.

Bus bất đồng bộ ra đời nhằm khắc phục những nhược điểm của bus đồng bộ. Trước hết master phát ra địa chỉ nhớ mà nó muốn truy cập, sau đó phát tín hiệu MREQ bù tích cực để xác định cần truy xuất bộ nhớ. Tín hiệu này cần thiết khi bộ nhớ và các cổng I/O sử dụng chung miền địa chỉ. Sau khi phát địa chỉ, bên master cũng phải phát tín hiệu RD tích cực để bên slave biết rằng master sẽ thực hiện thao tác đọc chứ không phải ghi.

Các tín hiệu MREQ bù và RD bù được đưa ra sau tín hiệu địa chỉ một khoảng thời gian phụ thuộc tốc độ hoạt động của master. Sau khi 2 tín hiệu này đã ổn định, master sẽ phát ra tín hiệu 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 được tín hiệu này, nó sẽ thực hiện công việc với tốc độ nhanh nhất có thể được, đưa dữ liệu của ô nhớ được yêu cầu lên bus dữ liệu. Khi hoàn thành slave sẽ phát tín hiệu SSYN (slave synchronization) tích cực.



Chu kỳ đọc của Bus bất đồng bộ

Master nhận được tín hiệu SSYN tích cực thì xác định được dữ liệu của slave đã sẵn sàng nên thực hiện việc chốt dữ liệu, sau đó đảo các đường địa chỉ cũng như các tín hiệu MREQ, RD, và SSYN. Khi slave nhận được tín hiệu MSYN không tích cực, nó xác định kết thúc chu kỳ và đảo tín hiệu SSYN làm bus trở lại trạng thái ban đầu, mọi tín hiệu đều không tích cực, chờ bus master mới. Trên giản đồ thời gian của bus bất đồng bộ, ta sử dụng mũi tên để thể hiện nguyên nhân và kết quả MSYN tích cực dẫn đến việc

truyền dữ liệu ra bus dữ liệu và đồng thời cũng dẫn đến việc slave phát ra tín hiệu SSYN tích cực, đến lượt mình tín hiệu SSYN lại gây ra sự đảo mức của các đường địa chỉ, MREQ bù, RD bù, và SSYN. Cuối cùng sự đảo mức của MSYN lại gây ra sự đảo mức tín hiệu SSYN và kết thúc chu kỳ.

Tập các tín hiệu phối hợp với nhau như vậy được gọi là bắt tay toàn phần (full handshake), chủ yếu gồm 4 tín hiệu sau:

- MSYN tích cực.
- SSYN bù tích cực để đáp lại tín hiệu MSYN.
- MSYN được đảo để đáp lại tín hiệu SSYN bù (tích cực).
- SSYN bù được đảo để đáp lại tín hiệu MSYN không tích cực.

Ta có thể nhận thấy bắt tay toàn phần là độc lập thời gian, mỗi sự kiện được gây ra bởi 1 sự kiện trước đó chứ không phải bởi xung clock. Nếu 1 cặp master-slave nào đó hoạt động chậm thì cặp master-slave kế tiếp không hề bị ảnh hưởng. Tuy ưu điểm của bus bất đồng bộ rất rõ ràng, nhưng trong thực tế phần lớn các bus đang sử dụng là loại đồng bộ. Nguyên nhân là các hệ thống sử dụng bus đồng bộ dễ thiết kế hơn. Vì xử lý chỉ cần chuyển các mức tín hiệu cần thiết sang trạng thái tích cực là bộ nhớ đáp ứng ngay, không cần tín hiệu phản hồi. Chỉ cần các chọn phù hợp thì mọi hoạt động đều trôi chảy, không cần phải bắt tay.

2.1.6 Phân xử bus (bus arbitration)

Trong hệ thống máy tính không phải chỉ có CPU làm bus master, các chip I/O cũng có lúc làm bus master để có thể đọc hay ghi bộ nhớ và gọi ngắt. Các bộ đồng xử lý cũng có thể làm bus master. Như vậy nảy sinh ra vấn đề: điều gì sẽ xảy ra khi 2 thiết bị trở lên đồng thời cần làm bus master? Từ đó cần có một cơ chế phân xử để tránh sự hỗn loạn của hệ thống. Cơ chế phân xử có thể là tập trung hay không tập trung.

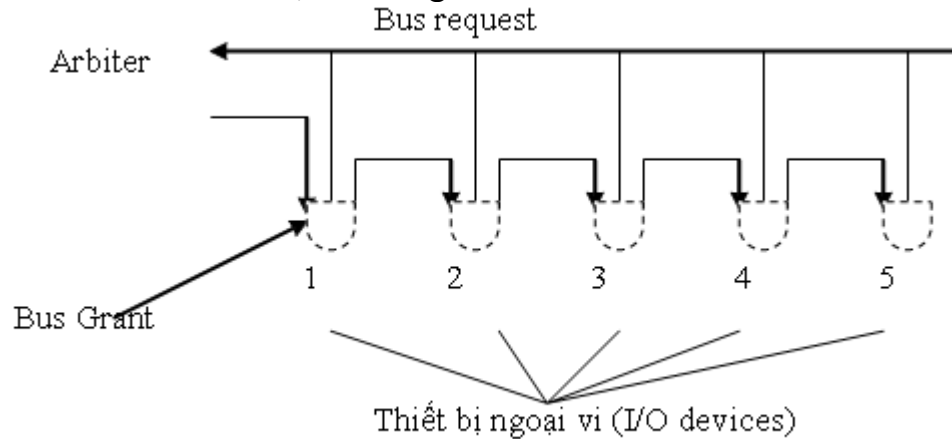
2.1.7 Phân xử bus tập trung

Nhiều vi xử lý có đơn vị phân xử được chế tạo nằm ngay trong chip CPU, trong một số máy tính mini, đơn vị này nằm ngoài chip CPU. Theo cơ chế này thì bộ phân xử (arbiter) chỉ có thể biết có yêu cầu chiếm dụng bus hay không mà không biết có bao nhiêu đơn vị muốn chiếm dụng bus. Khi arbiter nhận được yêu cầu, nó sẽ phát ra 1 tín hiệu cho phép trên đường dây (bus grant: cho phép sử dụng bus). Đường dây này nối qua tất cả các thiết bị I/O theo kiểu nối tiếp.

Khi thiết bị nằm gần arbiter nhất nhận được tín hiệu cho phép, nó kiểm tra xem có phải chính nó đã phát ra yêu cầu hay không. Nếu có thì nó sẽ chiếm lấy bus và không truyền tiếp tín hiệu cho phép trên đường dây. Nếu không thì nó sẽ truyền tín hiệu cho phép tới thiết bị kế tiếp trên đường dây, với thiết bị này sự việc xảy ra giống thiết bị trước nó, quá trình

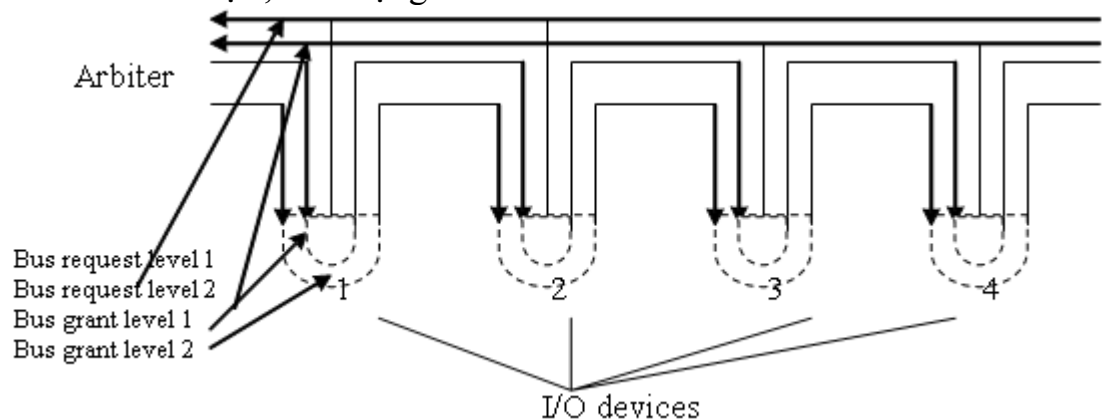
cứ tiếp diễn cho đến khi có một thiết bị chiếm lấy bus.

Sơ đồ xử lý như vậy có tên gọi là daisy chaining (chuỗi cánh hoa). Điểm nổi bật của sơ đồ này là các thiết bị được gán thứ tự ưu tiên tùy thuộc vào vị trí của nó so với arbiter, thiết bị gần hơn thì mức ưu tiên cao hơn.



Phân sử Bus tập trung một mức nối tiếp

Một số loại bus có nhiều mức độ ưu tiên, với mỗi mức độ ưu tiên có đường yêu cầu bus (bus request) và đường dây cho phép bus (bus grant). Ví dụ: giả sử 1 bus có 2 mức ưu tiên 1 và 2 (các bus thực tế có 4, 8 hay 16 mức). Mỗi thiết bị trong hệ thống máy tính nối với 1 trong các mức yêu cầu bus, các đường thường được sử dụng nhiều hơn được gán với đường dây có mức ưu tiên cao hơn. Ở ví dụ, các thiết bị 1, 2 sử dụng mức ưu tiên 1, còn các thiết bị 3, 4 sử dụng mức ưu tiên 2.



Phân sử Bus tập trung 2 mức

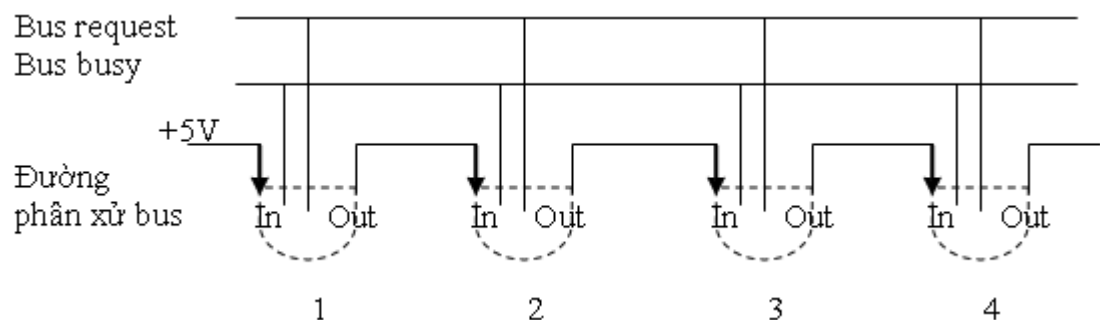
Nếu có một số thiết bị ở các mức ưu tiên khác nhau cùng yêu cầu, arbiter chỉ phát ra tín hiệu grant đối với yêu cầu có mức ưu tiên cao nhất. Trong số các thiết bị có cùng mức ưu tiên, thiết bị nào gần arbiter hơn sẽ ưu tiên hơn. Về mặt kỹ thuật, không cần nối đường grant level 2 giữa các thiết bị vì chúng không bao giờ đòi hỏi bus ở mức 2. Tuy nhiên, trong thực tế để thuận tiện cho việc lắp đặt người ta hay làm như sau: nối tất cả các đường grant thông qua tất cả các thiết bị, như vậy sẽ dễ dàng hơn là nối các đường

grant một cách riêng biệt, và từ đó căn cứ vào thiết bị nào có quyền ưu tiên cao hơn.

Một arbiter(phân xử) có đường dây thứ 3 nối tới các thiết bị để các thiết bị xác nhận đã nhận được tín hiệu grant và chiếm dụng bus – đường ACK (acknowledgement). Ngay sau 1 thiết bị phát tín hiệu tích cực trên đường dây ACK, có thể đảo tín hiệu trên các đường dây request và grant xuống mức không tích cực. Các thiết bị khác có thể yêu cầu bus khi thiết bị đầu tiên đang dùng bus. Khi sự truyền thông kết thúc, bus master kế tiếp sẽ được lựa chọn. Cách làm việc như vậy làm tăng hiệu quả sử dụng bus, nhưng cần thêm 1 đường truyền tín hiệu và cấu trúc thiết bị cũng phức tạp hơn. Các chip trong máy tính PDP-11 và các chip Motorola làm việc với các bus như vậy.

2.1.7 Phân xử bus không tập trung:

Trong Multibus, người ta cho phép có thể lựa chọn sử dụng phân xử bus tập trung hay không tập trung, cơ chế phân xử bus không tập trung được thực hiện như sau:



Phân xử Bus không tập trung trong multibus

Cơ chế sử dụng 3 đường dây, không phụ thuộc vào số lượng thiết bị nối với bus:

- Bus request: yêu cầu chiếm dụng bus.
- Bus busy: đường báo bận, được bus master đặt ở mức tích cực khi có thiết bị đang chiếm dụng bus
- Bus arbitration: được mắc nối tiếp thành 1 chuỗi xích qua tất cả các thiết bị ngoại vi. Đầu của chuỗi này được gắn với mức điện áp 5V của nguồn nuôi.

Khi không có đơn vị nào yêu cầu chiếm dụng bus, đường dây phân xử bus truyền mức tích cực tới tất cả các thiết bị trong chuỗi xích. Khi 1 đơn vị nào đó muốn chiếm dụng bus, đầu tiên nó kiểm tra xem bus có rảnh không và đầu vào In của đường trọng tải bus có mức tích cực hay không. Nếu không (not active) thì nó không trở thành bus master. Ngược lại, nó sẽ đảo đầu Out thành không tích cực, làm cho các thiết bị đứng sau nó trong chuỗi xích có đầu In không tích cực. Khi trạng thái có thể hiểu lầm (khoảng thời gian tín hiệu trên đầu In và Out đang thay đổi) qua đi, chỉ còn

lại duy nhất 1 thiết bị có đầu In tích cực và Out không tích cực. Thiết bị này trở thành bus master, nó sẽ đặt bus busy tích cực và bắt đầu truyền thông tin trên bus.

2.1.8 Xử lý ngắt:

Ở trên, ta chỉ khảo sát các chu kỳ bus thông thường, trong đó master nhận hay gửi thông tin từ / đến slave. Một ứng dụng quan trọng nữa của bus là dùng để xử lý ngắt. Khi CPU ra lệnh cho thiết bị I/O làm một việc gì đó, 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 sẽ đáp ứng ngay, có thể nhận dữ liệu do thiết bị I/O truyền về, hay gửi tiếp dữ liệu ra thiết bị I/O, hay CPU sẽ sử dụng bus cho một thao tác khác.... Như vậy chính ngắt phát ra tín hiệu yêu cầu sử dụng bus. Vì có thể nhiều thiết bị ngoại vi cùng phát ra ngắt, cho nên cần có 1 cơ chế phân xử 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 1 arbiter tập trung để trao quyền ưu tiên cho các thiết bị quan trọng thường xuyên được sử dụng.

§ 2.2 Bus mở rộng (Expansion bus)

Bus mở rộng cho phép PC liên lạc được với các thiết bị ngoại vi, các thiết bị này được cài đặt qua các khe cắm mở rộng (expansion slot). Các thông số chính của bus mở rộng: tốc độ truyền tối đa giữa các thiết bị với nhau và giữa các thiết bị với bộ nhớ chính, số đường địa chỉ (số lượng ô nhớ có thể được truy xuất bởi 1 thiết bị), số đường ngắt cứng,

2.2.1 Bus ISA (Industry Standard Architecture)

Bus ISA dùng cho hệ thống chỉ được điều khiển bởi 1 CPU trên bảng mạch chính, tức là tất cả các chương trình và thiết bị đều chỉ được điều khiển bởi CPU đó. Tần số làm việc cực đại là 8.33 MHz (tốc độ chuyển tải cực đại là 16.66 MBps với số liệu 2 bytes). Bề rộng dữ liệu là 8 hay 16 bits. ISA có 24 đường địa chỉ nên quản lý được 16 MB bộ nhớ. Bus ISA tương thích 90% với bus AT.

2.2.2 Bus EISA và MCA:

Sử dụng cho các CPU 32 bits (số liệu và đường địa chỉ) từ 80386 trở đi.

2.2.3 Bus EISA (Extended ISA):

Đây là chuẩn mở rộng của ISA để bố trí các dữ liệu 32 bits nhưng vẫn giữ được sự tương thích với mạch nối ghép cũ. Bus EISA có 2 nấc, các tín hiệu ISA được gửi qua nấc trên và các tín hiệu phụ trợ EISA qua nấc dưới. Các đặc trưng của EISA như sau:

- Về mặt cơ khí: có nhiều chân cắm hơn nhưng vẫn tương thích với ISA

- Độ rộng dữ liệu: có thể truy xuất 2 đường 8 bits (tương thích với ISA), hay 2 đường 16 bits Do. đó, đơn vị quản lý bus 32 bits có thể chuyển tải 4 byte với bộ nhớ hoặc thiết bị ngoại vi. Điều này góp phần tăng tốc độ truyền tải lên khoảng 33 MBps so với 16.66 MBps của ISA.
- Độ rộng địa chỉ: ngoài 24 đường như ISA còn thêm 8 đường bổ sung nữa, do đó có thể định địa chỉ trong 4 GB bộ nhớ.
- Phần cứng được thiết kế theo hệ thống EISA phức tạp hơn ISA vì nó cũng phải thực hiện các chu kỳ bus tương thích với ISA. EISA có thể thực hiện phân xử bus, nó cho phép vi xử lý nằm ngoài bảng mạch chính có thể điều khiển toàn bộ bus. Điều này rất hiệu quả trong các hệ thống đa xử lý (multiprocessor). Hãng Intel đã phát triển 4 chip điện tử phục vụ cho bus EISA như sau:
 - ISP (Intergrated system peripheral)
 - BMIC (Bus master interface controller)
 - EBC (EISA bus controller)
 - EBB (EISA bus buffer)

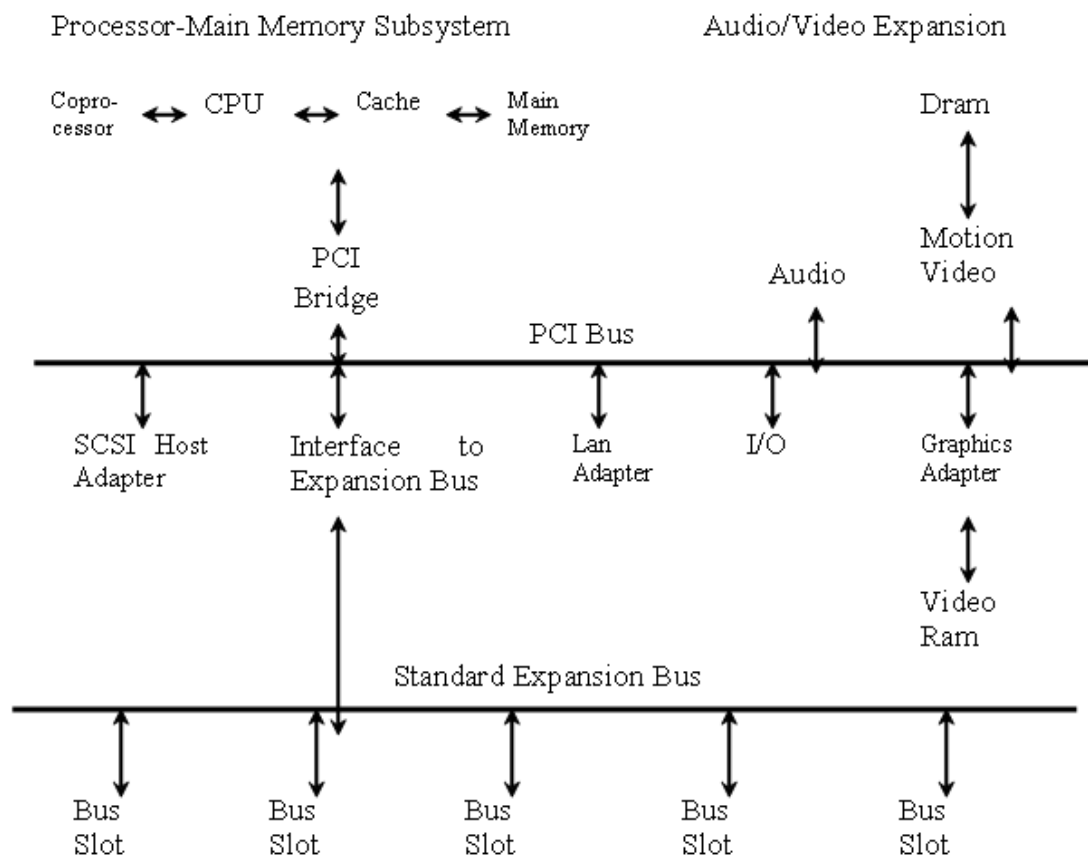
2.2.3 Bus MCA (Micro Channel Architecture)

Phục vụ cho hệ thống IBM PS/2 không tương thích với bus ISA, có thể hoạt động với 16 hay 32 bits dữ liệu. Nó có nhiều đường dẫn hơn ISA, thiết kế phức tạp cho phép giảm bớt các nhiễu cao tần của PC tới các thiết bị xung quanh. Tốc độ truyền dữ liệu có thể lên tới 160 MBps.

2.2.4 Bus cục bộ (Local Bus)

Nhược điểm của các bus chuẩn trên là mặc dù xung clock của CPU rất cao nhưng cũng chỉ làm việc với các ngoại vi với tốc độ truyền tải không quá 33MBps. Điều này không thể đáp ứng được tốc độ của các card đồ hoạ cắm vào khe cắm của bus mở rộng trong chế độ đồ hoạ. Chuẩn các *bus cục bộ* tạo thêm các *khe cắm mở rộng* nối trực tiếp vào bus cục bộ (bus nối giữa CPU và các bộ đệm). Do vậy, bus mở rộng loại này cho phép truy xuất lên trên 32 bit cũng như tận dụng được tốc độ xung clock của chính CPU, tránh được rào cản 8.33MHz của bus hệ thống. Theo hướng giải quyết này, Intel đã phát triển bus PCI và Ủy ban VESA (Video Electronics Standards Association) đã phát triển bus VL.

2.2.5 Bus PCI (Peripheral Component Interconnect)



Sơ đồ Bus PCI

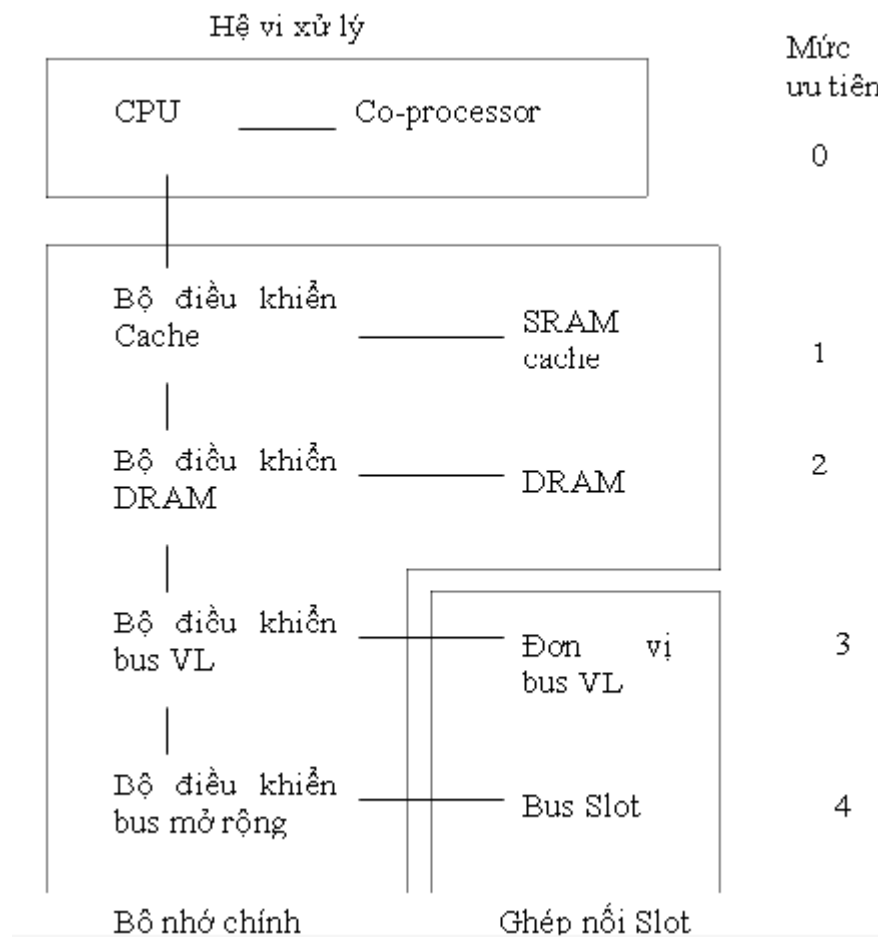
Bus PCI là bus của i486 trong đó dữ liệu và địa chỉ được gởi đi theo cách thức dồn kênh (multiplexing), các đường địa chỉ và dữ liệu được dồn chung trên các đường của PCI. Cách này tiết kiệm được số chân PCI nhưng lại hạn chế tốc độ vì phải cần 2 xung clock cho một quá trình truyền dữ liệu (1 cho địa chỉ và 1 cho dữ liệu). Việc nối giữa CPU, bộ nhớ chính, và bus PCI được thực hiện bằng *cầu PCI* (PCI bridge), qua đó bus PC sẽ phục vụ cho tất cả các đơn vị của bus PCI. Tối đa là 10 thiết bị có thể được nối tới bus PCI, trong đó cầu PCI được coi là một. Chu kỳ bus của PCI đạt gần bằng tốc độ chu kỳ bus của i486. Nó có thể hoạt động với độ rộng 32 bits dữ liệu và tốc độ 33MHz (có thể đạt 64 bits với tốc độ 66 MHz). Một điểm mạnh của PCI là dữ liệu được truyền tải theo kiểu cụm (burst), trong đó địa chỉ chỉ truyền đi 1 lần, sau đó nó sẽ được hiểu ngầm bằng cách cho các đơn vị phát hoặc thu đếm lên trong mỗi xung clock. Do đó, bus PCI hầu như được lấp đầy bởi dữ liệu. Tốc độ truyền tối đa trong kiểu burst có thể lên đến 120MBps.

2.4.6 Bus VL (VESA local bus)

Giống như PCI, bus VL cũng phân cách giữa hệ CPU, bộ nhớ chính, và bus mở rộng chuẩn. Thông qua bus cục bộ trên board mạch chính, nó có thể để điều khiển tối đa 3 thiết bị ngoại vi. Khe cắm VL có 116 tiếp điểm. Bus VL chạy với xung clock bên ngoài CPU, như vậy trong các máy DX2 thì

tần số này chỉ bằng một nửa clock CPU. Về mặt logic, mỗi một thiết bị có thể ở một trong hai vị trí: LMB (Local bus master) hoặc LBT (Local bus target). Bộ phận điều khiển bus cục bộ LBC (local bus controller) trên main board sẽ quyết định thiết bị nào sẽ trở thành LMB, tức là được nắm quyền điều khiển bus và cho phép nhường quyền đó cho thiết bị có quyền ưu tiên cao hơn. Thường có 3 cấp ưu tiên được sắp xếp theo thứ tự giảm dần như sau: DMA/làm tươi, CPU/đơn vị làm chủ bus (bus master) và các đơn vị làm chủ bus khác. Thiết bị nào ở vị trí LBT thì không có khả năng làm các việc liên quan đến chuyển tải dữ liệu.

Bus VL chỉ làm việc với 32 bits, trong tương lai sẽ được mở rộng đến 64 bits.



CHƯƠNG 3:

BỘ NHỚ

§ 3.1. Các đặc trưng của bộ nhớ.

Memory: Memory đơn giản là một thiết bị nhớ nó có thể ghi và chứa thông tin. ROM, RAM, Cache, Hard disk, Floppy disk, CD.... đều có thể gọi là memory cả (vì nó vẫn lưu thông tin). Dù là loại memory nào ta cũng có các thông số sau đây:

3.1.1. TÊN GỌI

Khái niệm RAM (Random Access Memory) là bộ nhớ truy xuất ngẫu nhiên, mất dữ liệu trong RAM khi mất điện. DRAM hay SDRAM là khái niệm mở rộng hơn (Synchronous Dynamic Random Access Memory - RAM đồng bộ). SDRAM là tên gọi chung của một dòng bộ nhớ máy tính, nó được phân ra SDR (Single Data Rate) và DDR (Double Data Rate). Do đó nếu gọi một cách chính xác, chúng ta sẽ có hai loại RAM chính là SDR SDRAM và DDR SDRAM. Cấu trúc của hai loại RAM này tương đối giống nhau, nhưng DDR có khả năng truyền dữ liệu ở cả hai điểm lên và xuống của tín hiệu nên tốc độ nhanh gấp đôi.

3.1.2. TỐC ĐỘ (SPEED)

Đây có lẽ là khái niệm được người dùng quan tâm nhất, tuy nhiên có người thắc mắc về cách gọi tên, đối với DDR thì có hai cách gọi theo tốc độ MHz hoặc theo băng thông. Ví dụ, khi nói DDR333 tức là thanh RAM đó mặc định hoạt động ở tốc độ 333MHz nhưng cách gọi PC2700 thì lại nói về băng thông RAM, tức là khi chạy ở tốc độ 333MHz thì nó sẽ đạt băng thông là 2700MB/s (trên lý thuyết).

3.1.3. ĐỘ TRỄ (LATENCY)

Là khoảng thời gian từ khi ra lệnh đến khi nhận được sự phản hồi. CAS là viết tắt của 'Column Address Strobe' (địa chỉ cột). RAS (Row Address Strobe) là địa chỉ hàng. khi chipset sẽ truy cập vào hàng ngang (ROW) của ma trận bộ nhớ thông qua việc đưa địa chỉ vào chân nhớ (chân RAM) rồi kích hoạt tín hiệu RAS. Chúng ta sẽ phải chờ khoảng vài xung nhịp hệ thống (RAS to CAS Delay) trước khi địa chỉ cột được đặt vào chân nhớ và tín hiệu CAS phát ra. Sau khi tín hiệu CAS phát đi, chúng ta tiếp tục phải chờ một khoảng thời gian nữa (đây chính là CAS Latency) thì dữ liệu sẽ được tìm thấy.

3.1.4. TẦN SỐ LÀM TƯỚI

Module DRAM được tạo nên bởi nhiều tế bào điện tử, mỗi tế bào này phải

được nạp lại điện hàng nghìn lần mỗi giây vì nếu không dữ liệu chứa trong chúng sẽ bị mất. Một số loại DRAM có khả năng tự làm tươi dữ liệu độc lập với bộ xử lý thường được sử dụng trong những thiết bị di động để tiết kiệm điện năng.

3.1.5. SDRAM ACCESS TIME

Việc cho ra đời cách đọc dữ liệu theo từng chuỗi (Burst Mode) đã giúp khắc phục nhiều nhược điểm và tăng hiệu năng cho RAM, chu kỳ của chuỗi ngắn hơn rất nhiều chu kỳ trang của RAM loại cũ. Chu kỳ của chuỗi cũng được coi như là chu kỳ xung nhịp của SDRAM và chính vì thế nó được coi như thang xác định cho tốc độ của RAM bởi đó là khoảng thời gian cần thiết giữa các lần truy xuất dữ liệu theo chuỗi của RAM. Những con số -12, -10, -8... ghi trên các chip RAM cho biết khoảng thời gian tối thiểu giữa mỗi lần truy xuất dữ liệu: nhân -12 xác định chu kỳ truy cập dữ liệu của RAM là 12ns (nano-giây).

3.1.6 Các loại bộ nhớ

+ **ROM (Read Only Memory)**

Có đặc tính là thông tin lưu trữ trong ROM không thể xoá được và không sửa được, thông tin sẽ được lưu trữ mãi mãi. Nhưng ngược lại ROM có bất lợi là một khi đã cài đặt thông tin vào rồi thì ROM sẽ không còn tính đa dụng

+ **PROM (Programmable ROM)**

Mặc dù ROM nguyên thủy là không xoá/ghi được, nhưng do sự tiến bộ trong khoa học, Thông tin có thể được "cài" vào chip và nó sẽ lưu lại mãi trong chip. Một đặc điểm lớn nhất của loại PROM là thông tin chỉ cài đặt một lần mà thôi.

+ **EPROM (Erasable Programmable ROM)**

Một dạng cao hơn PROM là EPROM, tức là ROM nhưng chúng ta có thể xoá và viết lại được. Dạng "CD-Erasable" là một điển hình. EPROM khác PROM ở chỗ là thông tin có thể được viết và xoá nhiều lần theo ý người sử dụng, và phương pháp xoá là hardware (dùng tia cực tím xoá) cho nên khá là tốn kém và không phải ai cũng trang bị được.

+ **EEPROM (Electronic Erasable Programmable ROM)**

Đây là một dạng cao hơn EPROM, đặt điểm khác biệt duy nhất so với EPROM là có thể ghi và xoá thông tin lại nhiều lần bằng software thay vì hardware. Ứng dụng của EEPROM cụ thể nhất là "flash BIOS". BIOS vốn là ROM và flash BIOS tức là tái cài đặt thông tin (upgrade) cho BIOS.

+**RAM (Random Access Memory)** thông tin có thể được truy cập không cần theo thứ tự.

+ **SRAM (Static RAM) và DRAM (Dynamic RAM)**

SRAM là loại RAM lưu giữ data mà không cần cập nhật thường xuyên (static).

Trên thực tế, chế tạo SRAM tốn kém hơn DRAM và SRAM thường có kích cỡ lớn hơn DRAM, nhưng tốc độ nhanh hơn DRAM vì không phải tốn thời gian refresh nhiều lần.

+ **BEDO-DRAM (Burst Extended Data Out DRAM)**

Là thế hệ sau của EDO DRAM, dùng kỹ thuật "pipeline technology" để rút ngắn thời gian dò địa chỉ của data. Nếu các ta để ý những mẫu RAM tôi giới thiệu trên theo trình tự kỹ thuật thì thấy là hầu hết các nhà chế tạo tìm cách nâng cao tốc độ truy cập thông tin của RAM bằng cách cải tiến cách dò địa chỉ hoạt cách chế tạo hardware.

+ **SDRAM (Synchronous DRAM)**

Đây là một loại RAM có nguyên lý chế tạo khác hẳn với các loại RAM trước. Như tên gọi của nó là "synchronous" DRAM, synchronous có nghĩa là đồng bộ, tốc độ xung đồng hồ của RAM đồng bộ với dữ liệu.

+ **DDR SDRAM (Double Data Rate SDRAM)**

Đây là loại memory cải tiến từ SDRAM. Nó nhân đôi tốc độ truy cập của SDRAM bằng cách dùng cả hai quá trình đồng bộ khi clock chuyển từ 0 sang 1 và từ 1 sang 0. Ngay khi clock của memory chuyển từ 0 sang 1 hoặc từ 1 sang 0 thì thông tin trong memory được truy cập.

+ **DRDRAM (Direct Rambus DRAM)**

Đây lại là một bước ngoặt mới trong lĩnh vực chế tạo memory, hệ thống Rambus (cũng là tên của một hãng chế tạo nó) có nguyên lý và cấu trúc chế tạo hoàn toàn khác loại SDRAM truyền thống. Memory sẽ được vận hành bởi một hệ thống phụ gọi là Direct Rambus Channel có độ rộng 16 bit và một clock 400MHz điều khiển. (có thể lên 800MHz)

+ **SLDRAM (Synchronous-Link DRAM)**

Là thế hệ sau của DRDRAM, thay vì dùng Direct Rambus Channel với chiều rộng 16bit và tốc độ 400MHz, SLDRAM dùng bus 64bit chạy với tốc độ 200MHz. Theo lý thuyết thì hệ thống mới có thể đạt được tốc độ $400\text{MHz} \times 64\text{ bits} = 400\text{MHz} \times 8\text{ bytes} = 3.2\text{Gb/giây}$, tức là gấp đôi DRDRAM. Điều thuận tiện là SLDRAM được phát triển bởi một nhóm 20 công ty hàng đầu về vi tính cho nên nó rất đa dụng và phù hợp nhiều hệ thống khác nhau.

+ **PC66, PC100, PC133, PC1600, PC2100, PC2400....**

$\text{PC xxx} * 2 * 8 = \text{bảng thông}$. Chiều rộng của DDR SDRAM: $\text{PC200} * 8 = \text{PC1600}$. Tương tự $\text{PC133} * 2 * 8\text{bytes} = \text{PC2100}$ và $\text{PC150} * 2 * 8 = \text{PC2400}$.

+ **Cache memory**

Là loại memory có dung lượng rất nhỏ và chạy rất nhanh (gần như tốc độ của CPU). Thông thường thì Cache memory nằm gần CPU và có nhiệm vụ cung cấp những data thường (đang) dùng cho CPU. Sự hình thành của Cache là một cách nâng cao hiệu quả truy cập thông tin của máy tính mà thôi. Những thông

tin ta thường dùng (hoặc đang dùng) thường được chứa trong Cache, mỗi khi xử lý hay thay đổi thông tin, CPU sẽ dò trong Cache memory trước xem có tồn tại hay không, nếu có nó sẽ lấy ra dùng lại còn không thì sẽ tìm tiếp vào RAM hoặc các bộ phận khác.

Lý do Cache memory nhỏ là vì nó rất đắt tiền và chế tạo rất khó khăn bởi nó gần như là CPU (về cấu thành và tốc độ). Thông thường Cache memory nằm gần CPU, trong nhiều trường hợp Cache memory nằm trong con CPU luôn. Người ta gọi Cache Level 1 (L1), Cache level 2 (L2)...là do vị trí của nó gần hay xa CPU. Cache L1 gần CPU nhất, sau đó là Cache L2...

+ **Interleave**

Là một kỹ thuật làm tăng tốc độ truy cập thông tin bằng giảm bớt thời gian nhàn rỗi của CPU. Ví dụ, CPU cần đọc thông tin thông từ hai nơi A và B khác nhau, vì CPU chạy quá lẹ cho nên A chưa kịp lấy đồ ra CPU phải chờ rồi! A thấy CPU chờ thì phiền quá mới bảo CPU sang B đòi luôn sau đó trở lại A lấy cũng chưa muộn! Bởi thế CPU có thể rút bớt thời gian mà lấy được đồ ở cả A và B. Toàn bộ nghĩa interleave là vậy.

+ **Bursting**

Cũng là một kỹ thuật khác để giảm thời gian truyền tải thông tin trong máy tính. Thay vì CPU lấy thông tin từng byte một, bursting sẽ giúp CPU lấy thông tin mỗi lần là một block.

3.1.7. Cách Tính Dung Lượng Của Memory (RAM)

Thông thường RAM có hai chỉ số, ví dụ, 32Mx4. Thông số đầu biểu thị số hàng (chiều sâu) của RAM trong đơn vị Mega Bit, thông số thứ nhì biểu thị số cột (chiều ngang) của RAM. $32 \times 4 = 32 \text{ MegaBit} \times 4 \text{ cột} = 128 \text{ Mega Bit} = 128/8 \text{ Mega Bytes} = 16 \text{ MB}$.

- **Số Pin của RAM**

Khi chọn RAM, ngoài việc chú ý tốc độ, sức chứa, ta phải coi số Pin của nó. Thông thường số Pin của RAM là (tùy vào loại RAM): 30, 72, 144, 160, 168, 184 pins.

- **SIMM (Single In-Line Memory Module)** Đây là loại ra đời sớm và có hai loại hoặc là 30 pins hoặc là 72 pins. Người ta hay gọi rõ là 30-pin SIMM hoặc 72-pin SIMM. Loại RAM (có cấu hình SIMM) này thường tải thông tin mỗi lần 8bits, sau đó phát triển lên 32bits. Ta cũng không cần quan tâm lắm đến cách vận hành của nó, nếu ra ngoài thị trường ta chỉ cần nhận dạng SIMM khi nó có 30 hoặc 72 pins. Loại 72-pin SIMM có chiều rộng 4 1/2" trong khi loại 30-pin SIMM có chiều rộng 3 1/2" (xem hình)

- DIMM (Dual In-line Memory Modules)

Cũng gần giống như loại SIMM mà thôi nhưng có số pins là 72 hoặc 168. Một đặc điểm khác để phân biệt DIMM với SIMM là cái chân (pins) của SIMM dính lại với nhau tạo thành một mảng để tiếp xúc với memory slot trong khi DIMM có các chân hoàn toàn cách rời độc lập với nhau. Một đặc điểm phụ nữa là DIMM được cài đặt thẳng đứng (ấn miếng RAM thẳng đứng vào memory slot) trong khi SIMM thì ấn vào nghiêng khoảng 45 độ. Thông thường loại 30 pins tải data 16bits, loại 72 pins tải data 32bits, loại 144 (cho notebook) hay 168 pins tải data 64bits.

- SO DIMM (Small Outline DIMM)

Đây là loại memory dùng cho notebook, có hai loại pin là 72 hoặc 144. Nếu ta để ý một tý thì thấy chúng có khổ hình nhỏ phù hợp cho notebook. Loại 72pins vận hành với 32bits, loại 144pins vận hành với 64bits.

- RIMM (Rambus In-line Memory Modules) và SO RIMM (RIMM dùng cho notebook)

Là technology của hãng Rambus, có 184 pins (RIMM) và 160 pins (SO RIMM) và truyền data mỗi lần 16bit (thế hệ cũ chỉ có 8bits mà thôi) cho nên chạy nhanh hơn các loại cũ. Tuy nhiên do chạy với tốc độ cao, RIMM memory tự nhiệt rất cao thành ra lối chế tạo nó cũng phải khác so với các loại RAM truyền thống.

3.1.8 Điện thế làm việc

Có một chỉ số về điện thế cung cấp cho RAM (DRAM Voltage). Thường thì DDR sử dụng mức điện thế 2,5v và DDR-II là 1,8v. Một số loại RAM DDR tốc độ cao có thể yêu cầu tới 2,8v hoặc 2,85v, đối với những loại này ta phải tham khảo tài liệu hướng dẫn đi kèm để có được thông tin.

Tuy nhiên cần tuân theo một nguyên tắc an toàn là: Không nên kéo điện thế lên quá 2,9v nếu không có giải pháp tản nhiệt hữu hiệu vì RAM có thể sẽ bị cháy hoặc hỏng IC sau một thời gian sử dụng.

§ 3.2. Sự phân cấp bộ nhớ

3.2.1 Xác định loại bộ nhớ

Hiện có 3 công nghệ bộ nhớ phổ biến là SDRAM, DDR-SDRAM và RDRAM nên ta cần xác định loại bộ nhớ dựa theo tài liệu hướng dẫn của bo mạch chủ.

SDRAM: Phổ biến trong các hệ thống Pentium, Pentium II, và Pentium III,

SDRAM có 3 loại: PC66, PC100 và PC133; tương ứng với tần số làm việc 66MHz, 100MHz và 133 MHz.

DDR SDRAM: Phổ biến trong hệ thống Pentium IV hay AMD. Cũng giống như SDRAM, DDR SDRAM cũng có nhiều loại tốc độ khác nhau như PC2100, PC2700, PC3200, PC3500 và PC3700 (xung làm việc tương ứng là 266MHz, 333MHz, 400MHz, 433MHz, và 466MHz)

RDRAM: Là công nghệ bộ nhớ tốt nhất, RDRAM sử dụng cho các hệ thống Xeon và Pentium IV cao cấp.

3.2.2 Video Ram (Vram)

VRAM được phát triển dựa trên công nghệ FPM (fast page mode), có hai cổng giao tiếp thay vì một cổng như thông thường: một cổng dành cho chức năng làm tươi màn hình) cổng còn lại xuất ảnh ra màn hình. Nhờ thiết kế này, VRAM hoạt động hiệu quả hơn DRAM trong những ứng dụng video. Tuy nhiên, do sản lượng tiêu thụ chip video ít hơn chip nhớ chính nên giá còn cao. Vì thế, trong một số hệ thống card video ít tiền, người ta có thể dùng DRAM thông thường để giảm giá thành.

3.2.3 Graphic ddr (gddr)

GDDR (DDR đồ họa) được phát triển dựa trên công nghệ DDR SDRAM dành riêng cho đồ họa. Sau phiên bản GDDR-2 thiết kế dựa trên DDR-II, ATI và NVIDIA đã kết hợp chặt chẽ với các nhà sản xuất bộ nhớ để đưa ra phiên bản GDDR-3 có điện áp làm việc thấp hơn GDDR-2, làm việc từ tần số 500MHz đến 800MHz với mục tiêu giảm điện năng tiêu thụ, tăng mật độ chip nhớ và đơn giản hóa giải pháp tản nhiệt.

3.2.4 Window ram (wram)

WRAM là một dạng bộ nhớ hai cổng khác, được dùng trong những hệ thống chuyên xử lý đồ họa. Hơi khác VRAM, WRAM có cổng hiển thị nhỏ hơn và hỗ trợ tính năng EDO (Extended Data Out).

3.2.5 Synchronous Graphic Ram (SGRAM)

SGRAM là loại SDRAM thiết kế riêng cho video với chức năng đọc/ghi đặc biệt. SGRAM cho phép truy xuất và chỉnh sửa dữ liệu theo khối thay vì từng đơn vị nên giảm bớt số lượt đọc/ghi bộ nhớ và tăng hiệu năng của bộ điều khiển đồ họa.

3.2.6 Base Rambus VÀ Concurrent Rambus

Trước khi trở thành công nghệ bộ nhớ chính, công nghệ Rambus được dùng làm bộ nhớ video. Công nghệ bộ nhớ Rambus dùng làm bộ nhớ chính hiện tại được gọi là Direct Rambus. Còn hai dạng Rambus sơ khai là Base Rambus và Concurrent Rambus được dùng cho ứng dụng video trong máy trạm và hệ thống game video như Nintendo 64

3.2.7 Bộ nhớ cải tiến

+ Enhanced Sdram (Esdrum)

Để tăng tốc độ và hiệu năng, thanh nhớ chuẩn có thể được tích hợp thêm bộ đệm SRAM (Static RAM) trực tiếp trên chip. ESDRAM là SDRAM có thêm bộ đệm SRAM để có khả năng làm việc với tần số 200MHz. Cũng tương tự nguyên lý bộ nhớ đệm ngoài, DRAM cũng dùng một bộ đệm SRAM để lưu dữ liệu thường dùng, nhằm rút ngắn thời gian truy xuất DRAM. Ưu điểm của SRAM trên chip là tạo lập tuyến bus rộng hơn giữa SRAM và DRAM, tăng cường băng thông và tốc độ DRAM một cách hiệu quả.

+ Fast Cycle Ram (Fcrum)

FCRAM được Toshiba và Fujitsu đồng phát triển nhằm phục vụ máy chủ, máy in cao cấp và hệ thống chuyên mạch viễn thông. Bộ nhớ được phân thành nhiều mảng và có thiết kế hàng đợi nên tăng được tốc độ truy xuất ngẫu nhiên và giảm điện năng tiêu thụ.

+ Synclink Dram (Sldram)

Hiện tại tuy đã lỗi thời nhưng SLDRAM từng được cộng đồng chế tạo DRAM phát triển nhằm cạnh tranh với Rambus vào cuối thập niên 1990.

+ Virtual Channel Memory (Vcm)

Do NEC phát triển, VCM cho phép các 'khối' bộ nhớ khác nhau giao tiếp độc lập với bộ điều khiển nhớ và có đệm riêng. Cách này cho phép mỗi tác vụ hệ thống thành một khối riêng, không chia sẻ hay dùng chung với các tác vụ cùng chạy khác

§ 3.3. Xây dựng bộ nhớ từ các chip nhớ.

3.3.1 Lưu trữ từ tính

kiểu lưu trữ từ tính dùng để lưu dữ liệu trên một trục các đĩa mỏng. Các đĩa này được chế tạo từ aluminum, thủy tinh hoặc ceramic và được bọc bên ngoài với một lớp vật liệu sắt từ, thường là hợp kim coban. Lớp vật liệu sắt từ này bao phủ sẽ cho phép đầu đọc/ghi có thể từ hóa các vùng nhỏ của đĩa để thể hiện dữ liệu số trên đó. Có nhiều đĩa mỏng bên trong một ổ cứng, chúng được cách ly với nhau bởi các miếng đệm trên một trục đơn. Trục này được điều khiển bởi một mô tơ có thể quay tròn các đĩa. Tốc độ của mô tơ này là không đổi và là tốc



độ của ổ cứng.

Các đầu đọc ghi trên từng mặt đĩa được gắn với một cánh tay chuyển động riêng. Cánh tay chuyển động này được điều khiển bằng một mô tơ servo có thể chuyển đầu đọc vào gần hoặc ra xa với trục. Có hai cách để ghi dữ liệu lên đĩa: theo chiều ngang và chiều vuông góc. Chiều ngang là cách truyền thống để ghi dữ liệu lên đĩa.

Công nghệ ghi vuông góc có thể cho phép các thiết kế đóng gói được nhiều dữ liệu hơn trên cùng một vùng của đĩa mà không phải buồn phiền về hiệu quả của siêu thuận từ cho các vùng có kích thước giống nhau. Nó hơi khó hiểu trong vấn đề tại sao chúng ta không phải lo lắng về hiệu quả của siêu thuận từ. Về bản chất vấn đề này là hướng của trường từ tính đã thay đổi, và vì vậy chúng tương tác với các thành phần bên cạnh của chúng khác nhau. Sự tương tác này rất quan trọng trong việc xác định xem siêu thuận từ có ảnh hưởng hay không. Tương tự với ổ cứng, các bit được lưu trên một dải băng bằng cách đảo cực một vùng từ tính nhỏ. Có hai kiểu cơ bản của dải băng đó là: tuyến tính và xoắn ốc. Các ổ băng tuyến tính có các rãnh tuyến tính. Trên băng có một số rãnh mở rộng từ băng này đến băng khác. Mỗi rãnh gồm có nhiều vùng từ tính nhỏ, các vùng này có thể được sử dụng để thể hiện bit dữ liệu '1' hoặc '0'. Các băng xoắn ốc có các rãnh có thể chạy theo đường chéo lên xuống theo băng. Điều đó có nghĩa rằng các rãnh sẽ chồng lên nhau. Bình thường điều đó là không tốt, tuy nhiên kiểu băng này sẽ dùng hai đầu ghi, mỗi đầu ghi lại sử dụng một trạng thái phân cực đối nhau, điều đó cho phép đầu đọc có thể phân biệt được các rãnh. Do vậy nó có thể cho dung lượng cao hơn khi sử dụng băng.

3.3.2 Bộ nhớ bán dẫn

Một trong những loại bộ nhớ bán dẫn thông thường đó là RAM, bạn có thể xem hình 3.3.2. Có hai loại RAM chung đó là động và tĩnh. Ram tĩnh (SRAM) lưu dữ liệu trong một bộ gồm có 6 transistor (6 transistor này tạo thành một khối được biết đến đó là một flip-flop (FF)). Ram động (DRAM) lưu dữ liệu bằng các tụ điện, cần phải làm tươi liên tục, đây là lý do tại sao DRAM không lưu được dữ liệu khi mất điện. Ưu điểm của DRAM là chỉ cần đến một transistor và mỗi tụ điện cho mỗi bit. Điều này làm cho nó có dung lượng nhớ cao hơn SRAM. Còn ưu điểm của SRAM là các transistor không cần đến việc làm tươi và tương tác nhanh hơn các tụ điện.



Hình 3.3.2: RAM

Một kiểu bộ nhớ bán dẫn khác đang được sử dụng nhiều hiện nay đó là bộ nhớ Flash. Trong loại bộ nhớ này cũng được chia thành hai kiểu: NOR và NAND. NOR (Not OR) thường sử dụng các cổng logic NOR trong khi đó NAND (Not AND) lại sử dụng các cổng logic NAND. Cả hai cổng logic NAND và NOR đều được cấu tạo từ các transistor và không chứa tụ điện trong đó. Điều này nghĩa là khi mất điện dữ liệu được lưu bên trong chúng sẽ không bị mất.

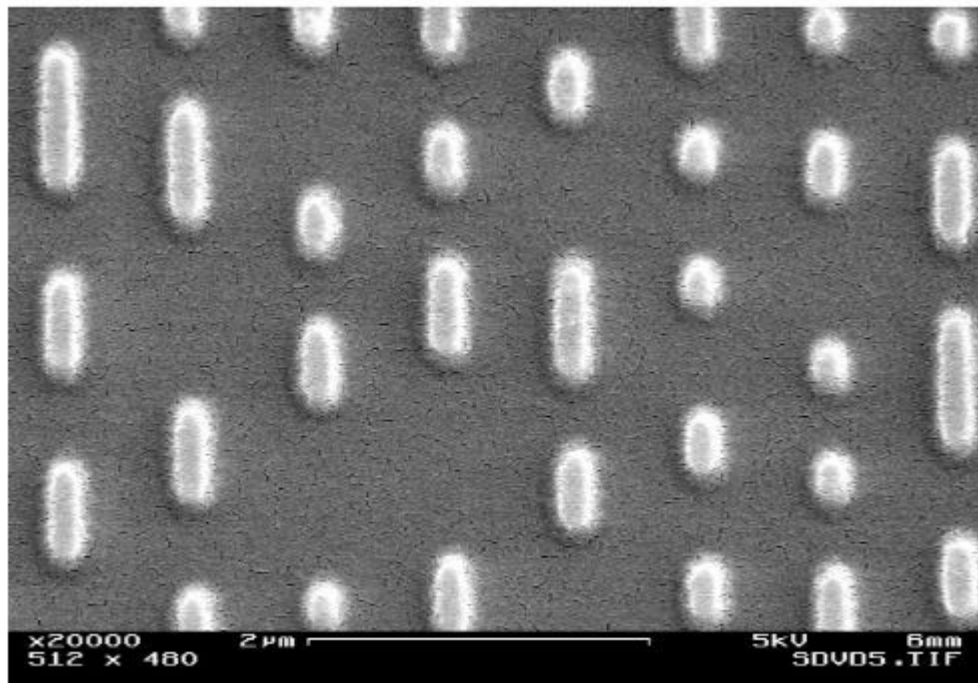
Mặc dù cả NAND và NOR Flash đều tương tự nhau nhưng chúng cũng có một số điểm khác nhau. NAND flash sử dụng công nghệ truy cập tuần tự phù hợp hơn cho việc lưu trữ dữ liệu. NOR flash là một công nghệ truy cập ngẫu nhiên, điều này làm cho nó tốt hơn trong việc lưu trữ các chương trình sử dụng tốn ít bộ nhớ. NOR flash thường được sử dụng trong các ứng dụng như chạy một hệ điều hành của điện thoại di động. NAND flash được sử dụng điển hình trong các ứng dụng như các thẻ nhớ USB.

3.3.3 Bộ nhớ quang

Một kiểu lưu trữ quang học thường được sử dụng nhất đó là CD. Các CD được sản xuất từ polycarbonat plastic có các lỗ nhỏ, các lỗ nhỏ này được sắp xếp theo hình xoắn ốc xung quanh đĩa, đó là các lỗ dùng để biểu thị dữ liệu. Trên polycarbonat này là một lớp vật liệu phản chiếu mỏng, thường là aluminum hay vàng và trên đó là một lớp acrylic để bảo vệ đĩa.

Khi một CD đang đọc, tia laser sẽ đập vào lớp polycarbonat và được phản chiếu vào lớp vật liệu phản chiếu. Tia laser phản chiếu quay trở lại được phát hiện bằng một cảm biến quang, dùng để chuyển đổi tín hiệu tia laser nhận được thành tín hiệu điện. Phụ thuộc vào tia laser tập trung vào các lỗ hổng khác nhau thì tín hiệu điện thu được cũng sẽ khác nhau vì tia phản chiếu khác nhau. Sự khác nhau trong các tín hiệu điện là cách một máy tính có thể đọc dữ liệu trên đĩa CD như thế nào. Đó là trường hợp đối với các CD thông thường, nhưng việc ghi dữ liệu lên các đĩa CD như thế nào? Một CD-R cũng tương tự như một CD trong cấu trúc chung của nó ngoại trừ có hai khía cạnh chính. Đầu tiên đó là nó không có các lỗ. Thứ hai, giữa polycarbonat và aluminum phản chiếu có một lớp thuốc nhuộm trong suốt.

Để lưu dữ liệu vào CD-R, tia laser dùng để ghi được tập trung vào phần xoắn ốc muốn ghi (đường xoắn ốc này không tồn tại cho tới khi bạn tạo nó bằng việc ghi dữ liệu lên) và nung nóng lớp thuốc nhuộm. Các thuộc tính hóa học của lớp thuốc nhuộm thay đổi tương ứng ở độ mờ của nó với nhiệt độ đốt nóng. Vì vậy tia laser dùng để ghi có thể chuyển động dọc theo đường xoắn ốc và thay đổi độ mờ của các vùng nhỏ, sự khác biệt trong độ mờ thể hiện ra dữ liệu đó là các con số '1' và '0'. Dữ liệu sau đó được đọc từ CD-R theo cách như một CD. CD-R chỉ có thể được ghi một lần. Điều này là bởi vì khi đã làm lớp thuốc nhuộm thay đổi thì bạn không thể làm cho nó trong suốt lại được lần nữa. Vậy sau CD-R là gì? CD-RW sử dụng lớp thuốc nhuộm khác, lúc đầu có màu đục, sau khi được đốt nóng có màu trong suốt. Thuốc nhuộm này cũng có thuộc tính khá thú vị đó là có thể trở lại trạng thái ban đầu nếu được đốt nóng đến một nhiệt độ cao hơn. Điều này cho phép bạn dễ dàng xóa các dữ liệu đã được lưu trên đĩa trước đó.



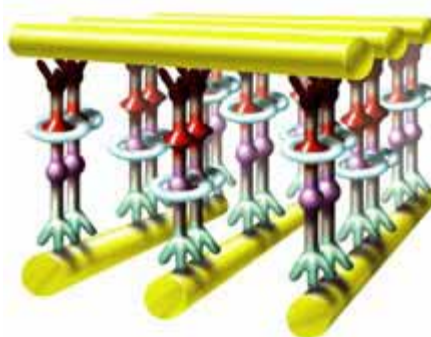
Hình 3.3.3. Ảnh của các lỗ trên một đĩa DVD

Các đĩa DVD làm việc cũng giống như đĩa CD. Các đĩa này có thể lưu nhiều dữ liệu hơn đĩa CD vì về bản chất chúng gồm nhiều đĩa CD mỏng được cất trữ trên một DVD. Điều đó được tạo từ một số lớp polycarbonat và vật liệu phản chiếu. Các tia laser và cảm biến quang cũng có nhiều cải tiến hơn, trong đó tia laser có khả năng xuyên qua các lớp khác nhau và cảm biến quang cũng có thể phát hiện được tất cả các lớp khác nhau đó.

3.3.4. Bộ nhớ phân tử

Động cơ cho việc phát triển công nghệ lưu trữ mới là chúng ta muốn nhanh chóng vượt đến được giới hạn nhỏ và nhanh trong các thiết bị, trong khi đó người dùng luôn yêu cầu dung lượng và hiệu suất tốt hơn. Chính vì vậy các công nghệ mới càng cần phải được nghiên cứu và sớm đưa ra hơn. Trong phần này chúng tôi có giới thiệu đến công nghệ nhớ phân tử. Vậy công nghệ nhớ phân tử là gì? Điều gì làm cho bộ nhớ phân tử hấp dẫn đến vậy, câu trả lời là các phân tử rất nhỏ và có thể cung cấp một mật độ nhớ lớn hơn gấp nhiều lần so với các công nghệ hiện tại. Để giữ một bit trong một phân tử, theo lý thuyết điều này khá đơn giản. Ta chỉ cần thêm hoặc bớt các electron trong mỗi phân tử đó. Điều khó khăn ở đây là việc đọc và ghi các bit dữ liệu đó như thế nào.

Để truy cập vào các phân tử để đọc và ghi, một số nhà nghiên cứu đã sắp xếp một mảng phân tử xung quanh các ống nano nhỏ có khả năng tích điện. Phương pháp này được thể hiện như trong hình 3.3.4. Một số chuyên gia nghiên cứu khác lại muốn gia công các bit dữ liệu thông qua sóng vô tuyến. Họ thực hiện điều đó bằng cách tạo một xung điện từ ở một tần số nào đó, xung này sau đó có thể thay đổi để nạp cho phân tử. Để đọc các bit dữ liệu, một xung tần số khác sẽ được tạo ra sau đó. Kết quả phân tử có xung thứ hai này có thể cho bạn biết rằng xung đầu tiên đã tương tác với phân tử, do vậy cho phép bạn lưu và sau đó đọc bit dữ liệu đó.



Hình 3.3.4. Sơ đồ thiết bị nhớ phân tử

Như những gì bạn có thể nhìn thấy ở trên là công nghệ bộ nhớ phân tử, công nghệ này có thể hứa hẹn sẽ cung cấp cho người dùng một mật độ nhớ lớn. Tuy nhiên, hiện nay bộ nhớ phân tử vẫn nằm trong các phòng thí nghiệm, vì vậy có lẽ chúng ta sẽ phải đợi đến vài năm tiếp theo để có thể thấy được công nghệ mới này sẽ mang đến những thuận lợi trong ứng dụng cho chúng ta như thế nào.

3.3.5. Bộ nhớ thay đổi pha

Không giống như bộ nhớ phân tử, bộ nhớ thay đổi pha hiện đã được đưa vào

ứng dụng. Trong thực tế, công nghệ bộ nhớ thay đổi pha được đưa ra cách đây khoảng vài thập kỷ. Vào năm những năm 60, Stanford Ovshinsky đã phát minh ra cách để kết tinh các vật liệu vô định hình, các vật liệu không có cấu trúc cụ thể. Các CD-R và CD-RW làm việc bởi tia laser thay đổi độ mờ của một vùng nhỏ trên mỗi đĩa. Sự thay đổi tính mờ của vật liệu từ vô định sang kết tinh, và ngược lại. Đây cũng là công nghệ được phát minh bởi Ovshinsky. Ovshinsky là người đầu tiên chế tạo CD-RW vào năm 1970. Sự khác nhau giữa công nghệ CD-R và công nghệ thay đổi pha là với bộ nhớ của công nghệ thay đổi pha, trạng thái kết tinh của một vùng nhỏ được thay đổi bằng một dòng điện chứ không phải tia laser. Khi không sử dụng tia laser để đọc và ghi dữ liệu thì chúng ta sẽ không làm mờ vùng nhưng lại xuất hiện điện trở suất ở vùng đó. Khi vùng đó thay đổi sang kết tinh hoặc vô định hình thì điện trở suất của vùng có thể đo được và dựa vào số điện trở suất người ta có thể phân biệt được đó là '1' hay '0'.

Bây giờ ta có thể thấy được điện trở xuất khá giống với tính mờ đục. Một vật liệu có điện trở không cho phép nhiều điện tích để lưu thông qua nó và vật liệu mờ không cho phép nhiều ánh sáng xuyên qua nó. Cũng nên biết rằng các vật liệu mờ trong thực tế có sự phản chiếu ánh sáng. Ta có thể không nhận ra rằng các vật liệu có điện trở cũng phản chiếu. Đúng hơn, nó là trở kháng (impedance) của vật liệu sẽ phản chiếu điện. Điện trở là một khía cạnh của những gì tạo nên trở kháng; các thành phần khác là điện dung và điện cảm. Trong nhiều ứng dụng, việc hạn chế sự phản chiếu bởi trở kháng phù hợp là một vấn đề lớn trong thiết kế. Bộ nhớ thay đổi pha có tiềm năng thay thế được bộ nhớ flash trong một vài năm tới. Vậy làm thế nào để có thể so sánh được với ổ flash? Giống như ổ flash, bộ nhớ thay đổi pha là bộ nhớ truy cập ngẫu nhiên ổn định làm cho nó phù hợp với cả chạy mã và lưu trữ dữ liệu. Năm 2006, IBM cùng với Macronix và Qimonda đã tuyên bố các kết quả nghiên cứu rằng họ đã thiết kế, xây dựng và minh chứng được thiết bị nhớ thay đổi pha đầu tiên. Thiết bị này nhanh hơn gấp 500 lần so với ổ flash trong khi sử dụng ít hơn một nửa công suất tiêu thụ. Thiết bị đầu tiên này cũng nhỏ gọn hơn các bộ nhớ flash.

3.3.6. Bộ nhớ Holographic

Nhiều người nghĩ rằng công nghệ holographic chỉ là mang tính lý thuyết, nhưng ngày nay nó đang dần trở thành một công nghệ hiện thực. Rõ ràng nó chưa được sử dụng rộng rãi và khá đắt đỏ. Tuy nhiên tình trạng này sẽ sớm được thay đổi bởi vì có rất nhiều ưu điểm để lưu trữ dữ liệu của bạn trên bộ nhớ công nghệ holographic này.



Hình 3.3.6. Thiết bị nhớ công nghệ holographic

Bộ nhớ holographic làm việc bằng cách chiếu hai chùm sáng dính liền vào một môi trường nhạy cảm với ánh sáng, một chùm dữ liệu và một chùm tham chiếu. Mẫu giao thoa kích thước ba chiều được tạo bởi hai chùm sáng này được lưu như một kỹ thuật tạo ảnh ba chiều. Mẫu giao thoa này có thể được đọc bằng cách chỉ chiếu sáng chùm tham chiếu vào mẫu giao thoa; chùm thu được sẽ giống với chùm dữ liệu gốc.

Kiểu bộ nhớ ba chiều này nghĩa là chúng ta có thể lưu và truy cập các trang bộ nhớ cùng một thời điểm. Nó cũng có nghĩa là các thiết bị nhớ holographic sẽ có một khả năng nhớ với số lượng lớn không tưởng.

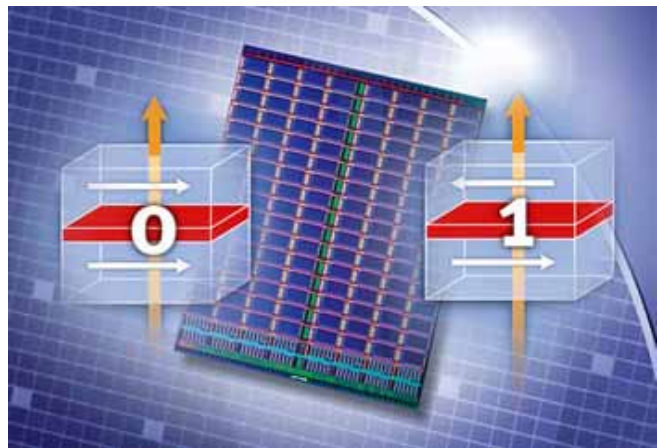
Với các ưu điểm của nó, holographic sẽ sớm trở thành một công nghệ lưu trữ được sử dụng trong thị trường lưu trữ thứ ba. Mặc dù vậy các thiết bị holographic sẽ ít phổ biến như các đĩa CD và DVD ngày nay.

3.3.7 MRAM_ Hiệu ứng Từ Điện Trở Của Ram

Bộ nhớ truy nhập ngẫu nhiên (RAM) là phần tử nhớ không thể thiếu trong các máy tính hiện nay. Điểm kém của bộ nhớ RAM hiện nay là dữ liệu bị xóa mất sau khi ngắt nguồn điện và tốc độ truy nhập còn hạn chế. Bộ nhớ RAM từ điện trở (MRAM) đang được nghiên cứu mạnh mẽ và sẽ là một thay thế hữu hiệu cho RAM truyền thống.

a. Sơ lược về hiệu ứng từ điện trở và MRAM

Bộ nhớ MRAM được mở đầu từ năm 1984 bởi 2 tiến sĩ Arthur Pohm và Jim Daughton lúc đó đang làm việc cho Honeywell, đưa ý tưởng về một loại bộ nhớ sử dụng hiệu ứng từ điện trở (Magnetoresistance Effect) cho phép tạo ra các bộ nhớ với mật độ lưu trữ thông tin cao, truy nhập ngẫu nhiên, và không tự xóa. Năm 1989, Daughton rời Honeywell và ý tưởng này bắt đầu được phát triển thành thương phẩm, đồng thời với sự nhảy vọt của hiệu ứng từ trở là sự phát hiện ra hiệu ứng từ điện trở khổng lồ (Giant Magnetoresistance - GMR) và từ điện trở chui hầm (Tunelling Magnetoresistance - TMR).

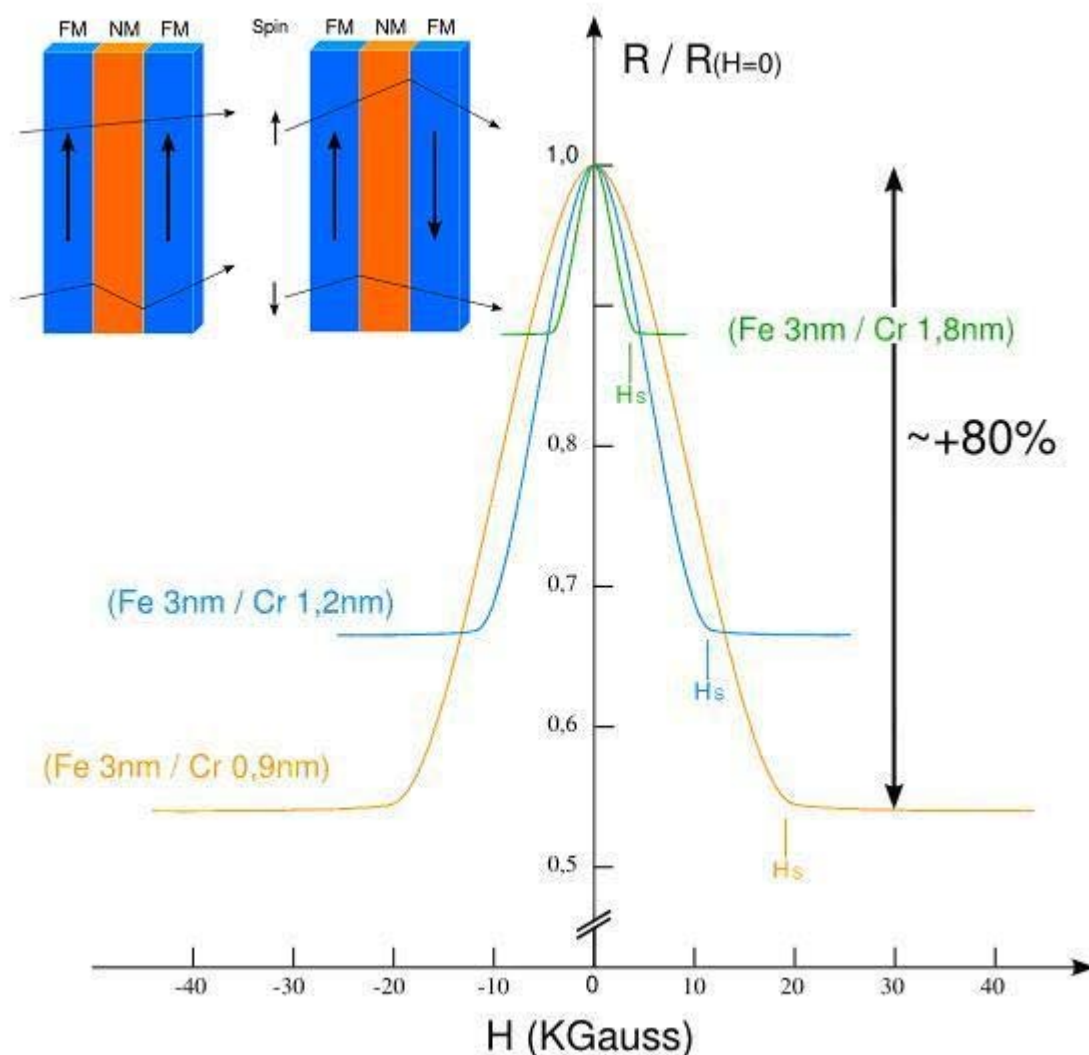


Thực chất hiệu ứng từ điện trở đã được nghiên cứu và sử dụng trước đó khá lâu, hiểu đơn giản là sự thay đổi điện trở suất trong chất rắn dưới sự tác dụng của từ trường. Trước đó, người ta thường ứng dụng hiệu ứng từ điện trở dị hướng (Anisotropic Magnetoresistance - AMR) trong các màng hợp kim Permalloy NiFe cho đầu đọc bộ nhớ và sensor từ. Để định nghĩa hiệu ứng từ trở, người ta đưa ra tỉ số từ trở:

$$MR(\%) = \frac{\rho(H) - \rho(H=0)}{\rho(H=0)} = \frac{R(H) - R(H=0)}{R(H=0)}$$

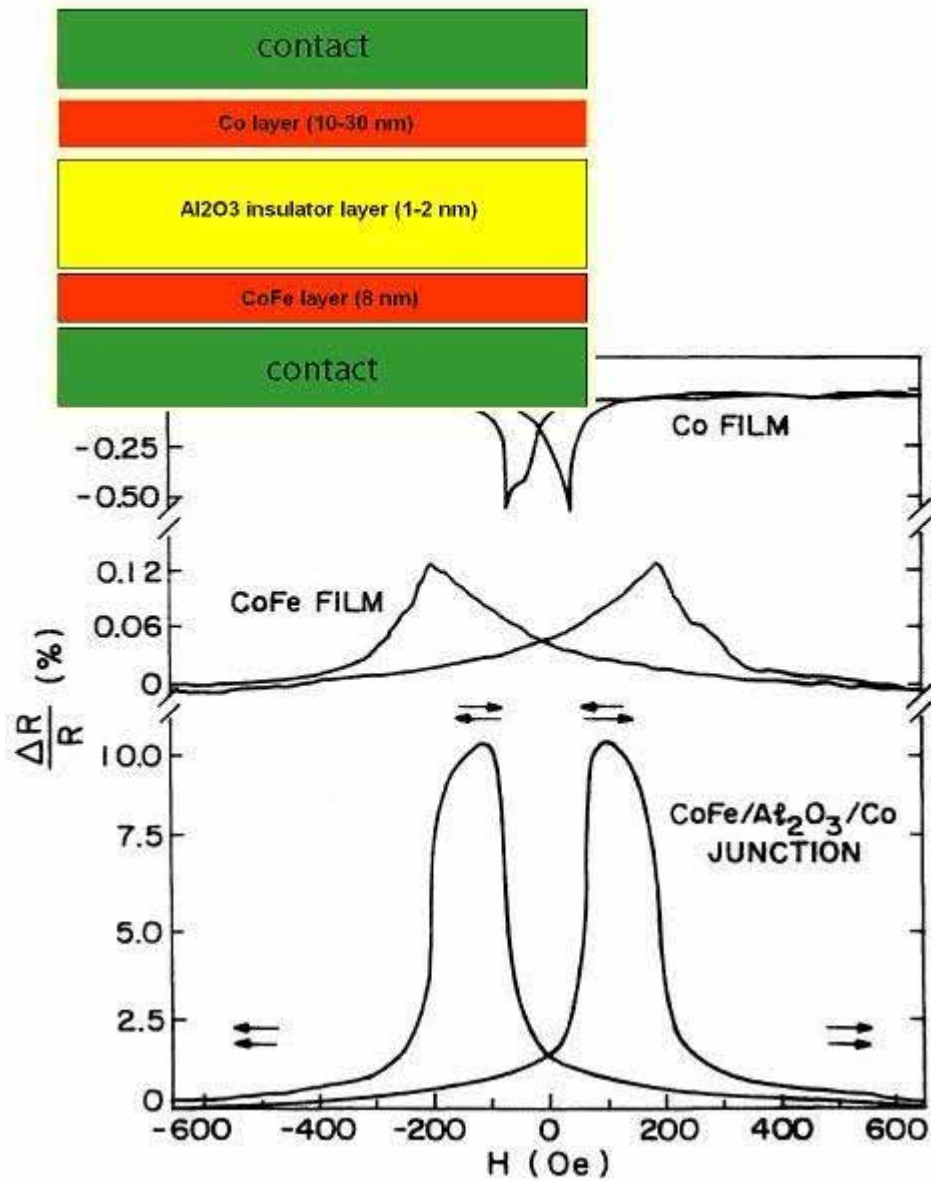
Hiệu ứng GMR lần đầu tiên được phát hiện bởi nhóm của Fert năm 1988 trên các màng đa lớp siêu mạng Fe/Cr. Với các màng này, có thể cho tỉ số MR tới vài chục % và còn có thể lớn hơn, và được gọi là hiệu ứng từ điện trở khổng lồ (xem hình 1). Các nghiên cứu sau này đã chỉ ra rằng tên gọi "khổng lồ" không xuất phát từ giá trị lớn của MR mà xuất phát từ cơ chế tạo ra hiệu ứng GMR, đó là cơ chế tán xạ phụ thuộc spin của điện tử qua các lớp sắt từ. Như ta biết rằng, điện tử có spin, điện trở của một chất phụ thuộc vào sự tán xạ của điện tử trên: trên nút mạng tinh thể, trên các moment từ, và trên sai hỏng. Từ trường ngoài gây ra sự định hướng của các moment từ và

do đó làm thay đổi sự tán xạ của điện tử trên các spin và đó là các hiệu đơn giản về hiệu ứng GMR.



Hình 1. Hiệu ứng GMR: đường cong thay đổi điện trở theo từ trường ngoài trong cấu trúc siêu mạng Fe/Cr.

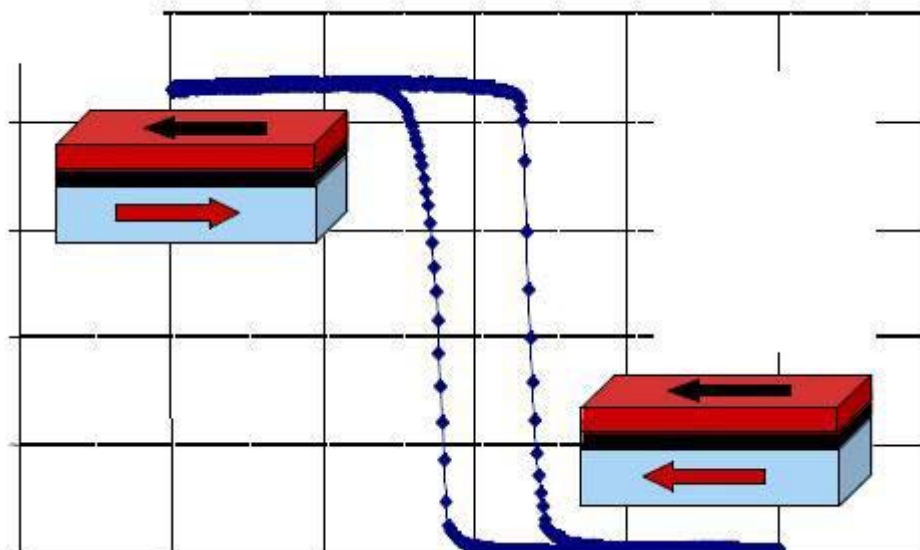
Hiệu ứng từ điện trở chui hầm (Tunneling Magnetoresistance - TMR) lần đầu tiên được phát hiện bởi nhóm của J.S. Moodera của MIT năm 1995 trên màng đa lớp. Lớp cách điện đóng vai trò là lớp cho điện tử chui hầm từ các lớp sắt từ sang nhau và tán xạ trên các lớp sắt từ. Có thể nói hai phát hiện này là đóng góp quan trọng cho sự nhảy vọt của MRAM biến ý tưởng của Pohm và Daughton thành hiện thực và MRAM bắt đầu được nghiên cứu chế tạo thành thương phẩm ở quy mô lớn tại hầu hết các phòng thí nghiệm lớn trên thế giới.



Hình 2. Đường cong MR trong các màng CoFe/Al₂O₃/Co - Hiệu ứng từ điện trở chui hầm

a. Cấu Trúc Của MRAM

- Ô nhớ cơ bản của RAM: Tiếp xúc chui hầm từ tính (Magnetic Tunnel Junction - MTJ)

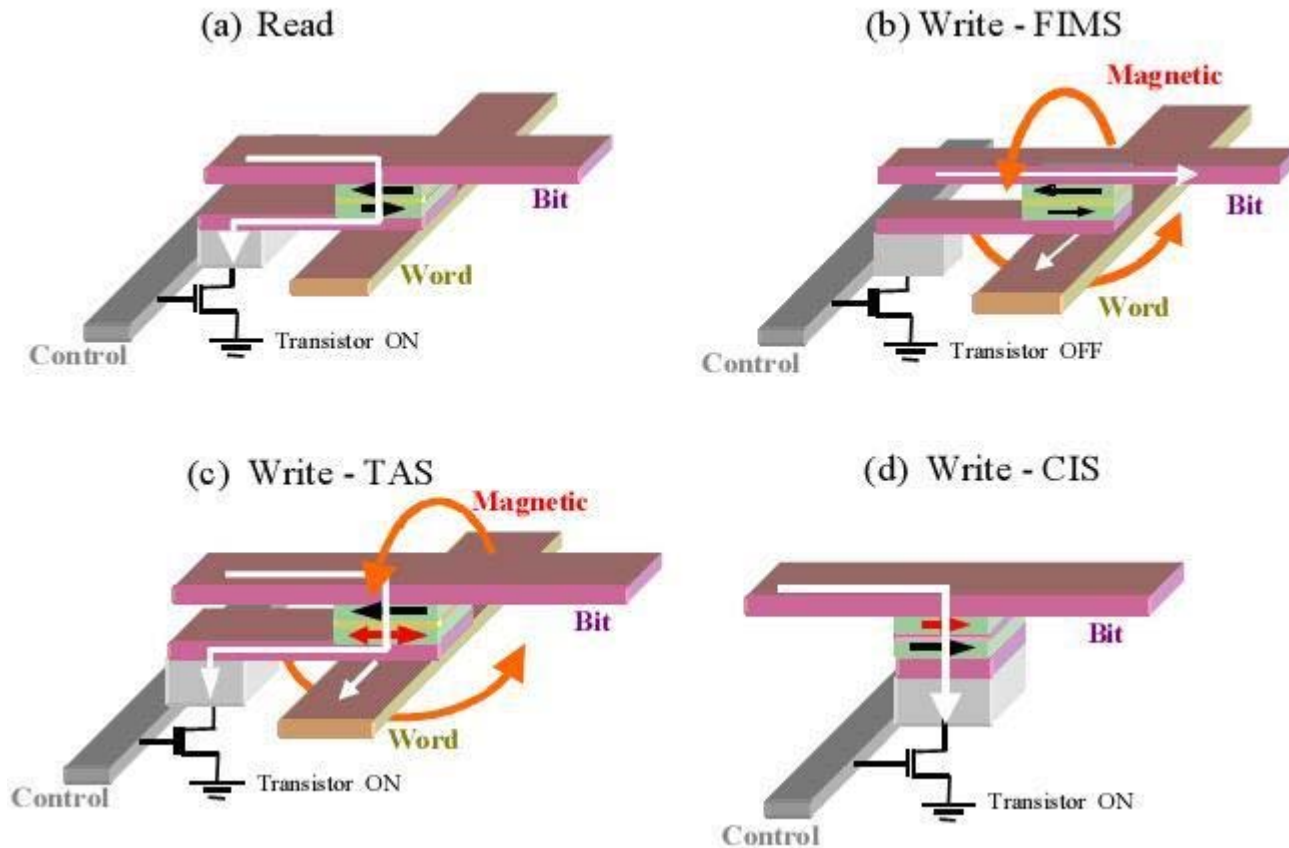


Hình 3. Ô nhớ của MRAM và các bit (0), (1) tương ứng với trạng thái điện trở thấp và cao (J.P. Nozieres, Spintech, CNRS).

Trong MRAM, thông tin không được lưu trữ bởi điện tích của điện tử như bộ nhớ bán dẫn mà được lưu trữ bởi spin của điện tử, mà cụ thể là theo sự định hướng của moment từ theo 2 chiều. Một ô nhớ cơ bản của MRAM được gọi là MTJ gồm 2 lớp từ tính kẹp giữa là một lớp cách điện mỏng (cỡ dưới nm) như hình 3. Moment từ của một lớp đóng vai trò lớp chuẩn, bị giữ cố định theo một chiều, còn moment từ của lớp còn lại như là lớp lưu trữ có thể đảo dưới tác dụng của từ trường từ song song đến phản song song với lớp chuẩn do đó dẫn đến sự thay đổi về điện trở của cấu hình (do sự tán xạ khác nhau của điện tử trong các trạng thái song song và phản song song). Các bit (0) và (1) được quy ước tương ứng với trạng thái điện trở thấp và cao. Trên thực tế, cấu trúc thực của một MTJ phức tạp hơn nhiều, mô hình trên chỉ là đơn giản hóa. Sự lưu trữ thông tin sau khi ngắt nguồn điện được xác lập nhờ sự giữ lại trạng thái của các moment từ (bản chất cố hữu của từ tính).

- Thế hệ đầu tiên: Đảo từ trường cảm ứng (Field-Induced Magnetic Switching - FIMS)

Một MRAM hoàn chỉnh gồm các dãy 2 chiều các ô nhớ riêng biệt có địa chỉ riêng. Trong cấu trúc ngày nay, mỗi ô nhớ là sự kết hợp của một transistor CMOS với một tiếp xúc chui hầm từ và 3 mức thẳng (line levels) như hình 4.



Hình 4. Phương pháp đọc và ghi ở MRAM trong các cấu trúc phổ biến (a): Nguyên tắc đọc, (b) cách ghi trong cấu trúc FIMS, (c) cách ghi trong cấu trúc TAS, (d) cách ghi trong cấu trúc CIS.

+ Khi đọc dữ liệu, một dòng xung công suất thấp sẽ đi vào qua cổng Control và mở transistor dẫn tới địa chỉ ô nhớ được chọn, điện trở của ô được xác định bằng cách điều khiển dòng qua "word line" qua tiếp xúc chui hầm từ và so sánh với ô lấy mẫu trong dây (hình 4a).

+ Dữ liệu được ghi theo nguyên lý từ trễ. Các "word line" và "bit line" sắp xếp qua 2 cực của tiếp xúc chui hầm từ và được hoạt động nhờ một dòng xung đồng bộ để tạo ra một từ trường tại địa chỉ ô nhớ. Cường độ dòng được chọn sao cho chỉ lớp nhớ của tiếp xúc từ có thể bị đảo từ còn các lớp lấy mẫu vẫn giữ nguyên trạng thái. Điều này có thể tạo được là do đặc tính của các cấu trúc nano.

Cấu trúc FIMS đã được sử dụng rất hiệu quả trong thế hệ MRAM đầu tiên. Tuy nhiên, nó có những hạn chế khi kích thước ô nhớ giảm xuống dưới 1 μm :

- + Công suất ghi sẽ tăng lên do trường đảo từ tỷ lệ nghịch với kích thước hạt.
- + Các lỗi lựa chọn ở chế độ ghi cũng tăng lên khi phân bố trường đảo từ SFD (Switching Field Distribution) có xu hướng tăng vọt lên.

+ Sự bền vững của dữ liệu trong thời gian dài có nguy cơ bị tác động do sự tăng các tác động của kích thích nhiệt.

- *Thế hệ sử dụng chế độ đảo từ nhờ nhiệt (Thermally Assisted Switching - TAS)*

Như vậy, giảm kích thước ô nhớ xuống dưới 1 μm là một thách thức cho MRAM. Một kỹ thuật ghi khác phát triển bởi SPINTECH (CNRS, France) có thể loại bỏ điều này là đảo từ nhờ nhiệt (TAS). Điều này thực hiện nhờ đặc tính phụ thuộc nhiệt độ của trường đảo từ trong các hạt nano. Trong chế độ TAS, các transistor CMOS sẽ mở ở chế độ ghi, và sẽ có một dòng xung ngắn chạy qua lớn tiếp xúc từ đồng thời với dòng xung tạo ra từ trường ghi và sinh ra nhiệt tại lớp rào thế chui hầm (như một điện trở) và nhanh chóng đốt nóng lớp kim loại của tiếp xúc từ. Kết quả là trường đảo từ bị giảm xuống tại lớp lưu trữ và cho phép ghi dễ dàng hơn

Cách này có nhiều ưu thế hơn so với phương pháp cũ:

+ Lỗi địa chỉ bị giảm xuống do quá trình lựa chọn ghi lúc này hầu như bị điều khiển bởi nhiệt độ.

+ Dù dòng đốt bổ sung, nhưng công suất ghi toàn thể có thể giảm giảm rất nhiều so với chế độ FIMS và hầu như không phụ thuộc vào kích thước ô nhớ.

+ Tốc độ ghi được tăng lên do khả năng địa chỉ đồng thời (song song) với xác suất lỗi thấp.

+ Sự bền nhiệt có thể cải tiến bằng cách thay thế các vật liệu có trường đảo từ lớn hơn tại nhiệt độ hoạt động.

- *Thế hệ sử dụng dòng phân cực spin cảm ứng (Spin Polarized Current Induced Switching - CIS)*

Đây là phương pháp mới được phát hiện gần đây sử dụng nguyên tắc kiểu bộ nhớ RAM truyền thống như các flash RAM. Bức tranh đơn giản của CIS là khi dòng chạy qua vật liệu từ, các spin bị phân cực, ví dụ như sự mất cân bằng giữa spin up và down. Khi dòng điện đi vào các lớp từ tính khác, sự mất cân bằng spin này sinh ra các mômen xoắn tại các từ độ địa phương và có thể gây ra một sự đảo từ. Cách thức này có rất nhiều ưu thế:

+ Không hề có lỗi địa chỉ

+ Có thể nâng gấp đôi mật độ bộ nhớ

+ Loại trừ sự ảnh hưởng của hiệu ứng kích thước ô nhớ

+ Giảm công suất đọc ghi

Tuy nhiên, phương pháp này mới đang được phát triển gần đây.

Có thể nói, MRAM sẽ là một tiến bộ thay thế cho các bộ nhớ RAM truyền thống (SRAM, DRAM) với các ưu điểm:

- Mật độ cao (tăng dung lượng)

- Dữ liệu không bị xóa mất

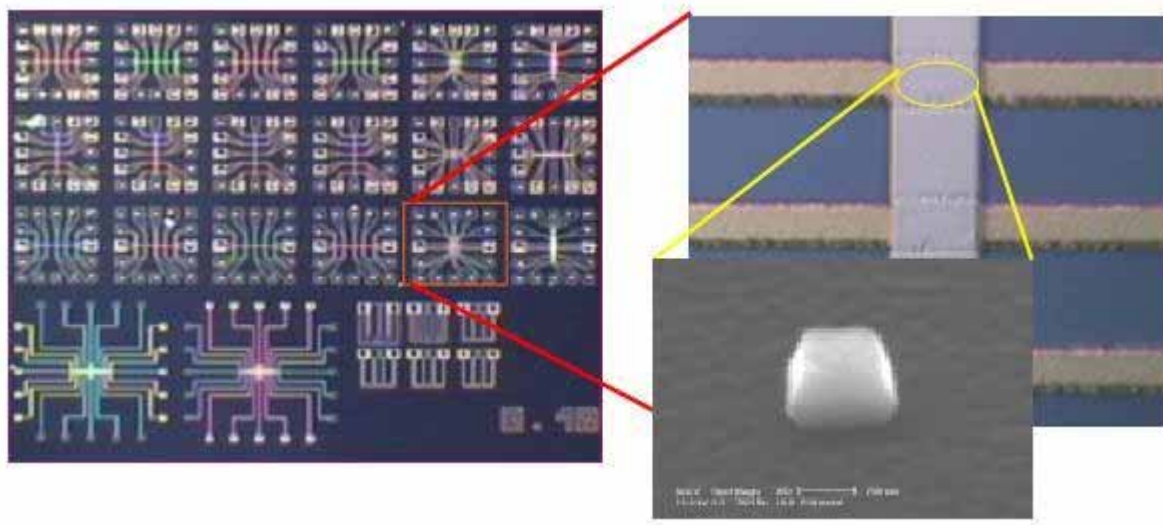
- Tốc độ truy xuất cao hơn

- Công suất tiêu tốn giảm

MRAM chính là một sản phẩm của công nghệ spintronics, điều khiển các spin của điện tử trong các linh kiện mới mà những thành tựu của nó được phát triển từ các kết quả nghiên cứu về vật liệu từ nano (hiệu ứng từ điện trở, từ trễ...). Trong một tương lai không xa, bộ nhớ MRAM sẽ trở thành thương phẩm phổ biến thay thế cho các bộ nhớ cũ. Bảng dưới đây so sánh MRAM với các loại RAM truyền thống.

	DRAM	SRAM	FLASH	FeRAM	OUM	MRAM
Write speed	Moderate	Fast	Slow	Moderate	Moderate	Fast
Read speed	Moderate	Fast	Fast	Moderate	Fast	Fast
Density	High	Low	High	Medium	High	High
Endurability	Good	Good	Poor	Poor	Good	Good
Power	High	Low	Low	Low	Low	Low
Refresh	Yes	No	No	No	No	No
Retention	No	No	Yes	Partially	Yes	Yes
Scalability	Bad	Good	Good	Medium	Good	Good
Write/Erase	Charge (Capacitance)	CMOS Logic	Charge (Tunneling)	Ferroelectric	Phase transition	Magnetization

Dưới đây là một hình chụp MRAM phát triển bởi SPINTEC



Hình 5. Ảnh chụp một MRAM phát triển bởi SPINTEC.

	7	6	5	4	3	2	1	0
Tín hiệu máy in	D7	D6	D5	D4	D3	D2	D1	D0
Chân số	9	8	7	6	5	4	3	2

Thanh ghi trạng thái máy in (chỉ đọc):

	7	6	5	4	3	2	1	0
Tín hiệu máy in	BSY	/ACK	PAP	OFON	/FEH	x	x	x
Số chân cắm	11	10	12	13	15	-	-	-

Thanh ghi điều khiển máy in:

	7	6	5	4	3	2	1	0
Tín hiệu máy in	x	x	x	IRQ	/DSL	/INI	/ALF	STR
Số chân cắm	-	-	-	-	17	16	14	1

x: không sử dụng

IRQ: yêu cầu ngắt cứng; 1 = cho phép; 0 = không cho phép

Bản mạch ghép nối chỉ có bus dữ liệu 8 bit do dữ liệu luôn đi qua máy in thành

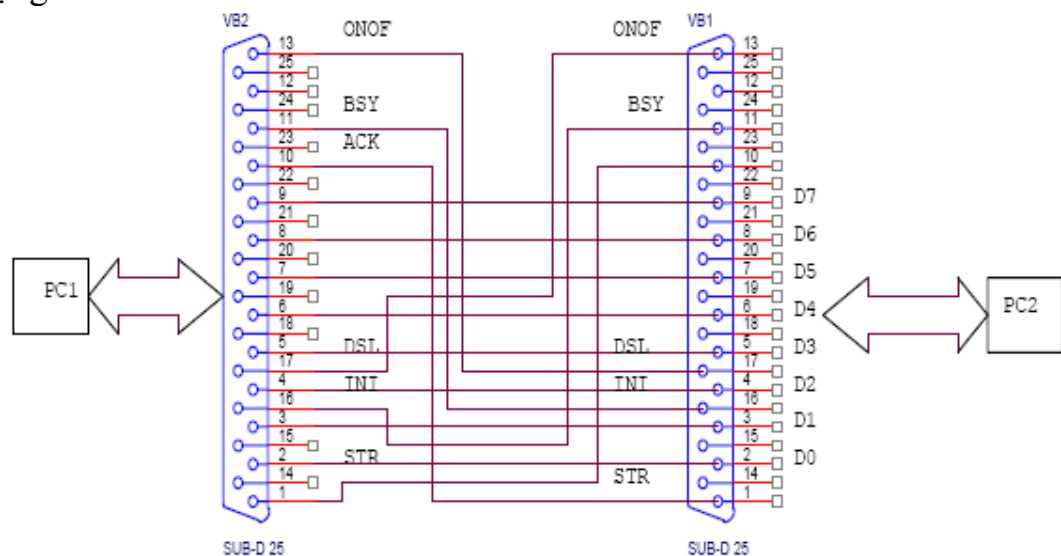
từng khối 8 bit. Các chân tín hiệu của đầu cắm 25 chân của cổng song song LPT như sau:

Chân	Tín hiệu	Mô tả
1	STR	Mức tín hiệu thấp, truyền dữ liệu tới máy in
2	D0	Bit dữ liệu 0
3	D1	Bit dữ liệu 1
4	D2	Bit dữ liệu 2
5	D3	Bit dữ liệu 3
6	D4	Bit dữ liệu 4
7	D5	Bit dữ liệu 5
8	D6	Bit dữ liệu 6
9	D7	Bit dữ liệu 7
10	ACK	Mức thấp: máy in đã nhận 1 ký tự và có khả năng nhận nữa
11	BSY	Mức cao: ký tự đã được nhận; bộ đệm máy in đầy; khởi động máy in; máy in ở trạng thái off-line.
12	PAP	Mức cao: hết giấy
13	OFON	Mức cao: máy in ở trạng thái online
14	ALF	Tự động xuống dòng; mức thấp: máy in xuống dòng tự động
15	FEH	Mức thấp: hết giấy; máy in ở offline; lỗi máy in
16	INI	Mức thấp: khởi động máy in
17	DSL	Mức thấp: chọn máy in
18-25	GROUND	0V

Thường tốc độ xử lý dữ liệu của các thiết bị ngoại vi như máy in chậm hơn PC

nhiều nên các đường ACK, BSY và STR được sử dụng cho kỹ thuật bắt tay. Khởi đầu, PC đặt dữ liệu lên bus sau đó kích hoạt đường STR xuống mức thấp để thông tin cho máy in biết rằng số liệu đã ổn định trên bus. Khi máy in xử lý xong dữ liệu, nó sẽ trả lại tín hiệu ACK xuống mức thấp để ghi nhận. PC đợi cho đến khi đường BSY từ máy in xuống thấp (máy in không bận) thì sẽ đưa tiếp dữ liệu lên bus.

Dữ liệu có thể trao đổi trực tiếp giữa 2 PC qua các cổng song song với nhau. Muốn vậy, các đường điều khiển bên này phải được kết nối với các đường trạng thái bên kia.



Hình 5.2 - Trao đổi dữ liệu qua cổng song song giữa 2 PC

Máy in có thể được truy xuất bằng chương trình qua DOS, BIOS hay trực tiếp qua

các cổng. Các lệnh như “copy tên_file << PRN” trong DOS cho phép in 1 file ra máy in. Ngắt 17h với hàm 01h khởi động máy in, 00h in 1 ký tự ra máy in, 02h trả về trạng thái của máy in,... có sẵn trong BIOS.

4.1.2. Cổng Nối tiếp (Serial port)

a. Truyền nối tiếp đồng bộ và bất đồng bộ

Ghép nối tiếp cho phép trao đổi giữa các thiết bị từng bit một. Dữ liệu thường được gửi theo các nhóm bit SDU (serial data unit) mà mỗi nhóm tạo thành 1 byte hay 1 word. Các thiết bị ngoại vi như: máy vẽ, modem, chuột có thể được nối với PC qua cổng nối tiếp COM. Sự khác nhau giữa truyền nối tiếp đồng bộ và bất đồng bộ là: trong kỹ thuật truyền đồng bộ, ngoài đường dây dữ liệu phải đưa thêm vào một hoặc vài đường tín hiệu đồng bộ để cho biết rằng khi nào bit tiếp theo ổn định trên đường truyền. Ngược lại

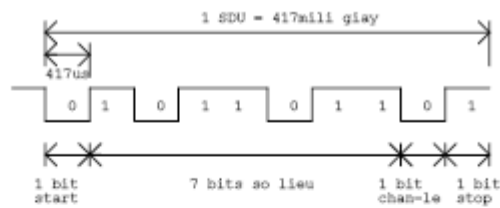
trong truyền bất đồng bộ, các bit dữ liệu tự nó chứa các thông tin để đồng bộ; phần phát và phần thu phải hoạt động với cùng 1 tần số xung clock. Thông tin đồng bộ (trong truyền bất đồng bộ) gồm có các bit start (cho biết bắt đầu của khối dữ liệu được truyền) và một bit stop (cho biết kết thúc khối dữ liệu).

b. Kiểm tra chẵn lẻ và tốc độ truyền

Bit chẵn lẻ (parity bit) được đưa vào khung SDU dùng để phát hiện lỗi trên đường truyền. Việc truyền bit chẵn lẻ chỉ kiểm soát được các lỗi trên đường truyền ngắn và các lỗi bit đơn nên trong một số ứng dụng đặc biệt người ta phải dùng mã CRC mặc dù phức tạp hơn. Tuy nhiên, gần như tất cả các chip hỗ trợ cho ghép nối nối tiếp ngày nay đều được thiết kế phần cứng kiểm tra chẵn lẻ. Một thông số khác liên quan tới truyền dữ liệu nối tiếp là tốc độ truyền dữ liệu được gọi là tốc độ baud. Trong việc truyền mã nhị phân, đó là số bit được truyền trong một giây (bps - bit per second).

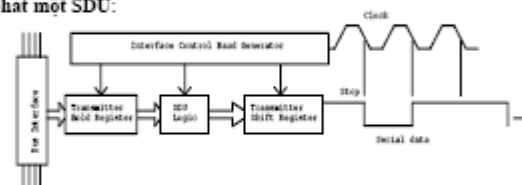
c. Nhóm dữ liệu nối tiếp SDU và nối tiếp hóa

Trước khi truyền chuỗi số liệu nối tiếp, máy phát và máy thu phải được khởi tạo để hoạt động với cùng một định dạng dữ liệu, cùng một tốc độ truyền. Một SDU với 1 bit start, 7 bits số liệu, 1 bit chẵn lẻ và 1 bit stop mô tả như hình vẽ. Lưu ý rằng: bit start luôn bằng 0 (space) và bit stop luôn bằng 1 (mark).

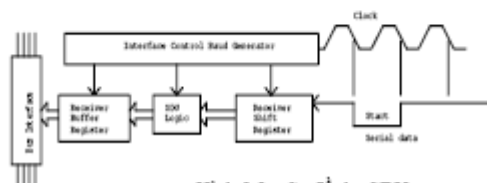


Hình 5.3 - Nhóm số liệu nối tiếp SDU.

Thu, phát một SDU:



Hình 5.4 - Sơ đồ phát SDU



Hình 5.5 - Sơ đồ thu SDU

Bus interface: ghép nối bus;

Serial data: dữ liệu nối tiếp;

Transmitter holder register: thanh ghi đệm giữ dữ liệu phát;

Transmitter shift register: thanh ghi dịch dữ liệu phát;

Receiver buffer register: thanh ghi đệm dữ liệu thu;

Receiver shift register: thanh ghi dịch dữ liệu thu;

SDU logic: mạch logic SDU;

Interface control baud generator: máy phát điều khiển tốc độ truyền dữ liệu baud;

Clock: xung clock;

4.1.3 Chuẩn ghép nối RS-232

Các ghép nối của PC cho trao đổi nối tiếp đều theo tiêu chuẩn RS-232 của EIA (Electronic Industries Association) hoặc của CCITT ở Châu Âu. Chuẩn này quy định ghép nối về cơ khí, điện, và logic giữa một thiết bị đầu cuối số liệu DTE (Data Terminal Equipment) và thiết bị thông tin số liệu DCE (Data Communication Equipment). Thí dụ, DTE là PC và DCE là MODEM. Có 25 đường với đầu cắm 25 chân D25 giữa DTE và DCE. Hầu hết việc truyền số liệu là bất đồng bộ. Có 11 tín hiệu trong chuẩn RS232C dùng cho PC, IBM còn quy định thêm đầu cắm 9 chân D9. Các chân tín hiệu và mối quan hệ giữa các đầu cắm 25 chân và 9 chân:

D25	D9	Tín hiệu	Hướng truyền	Mô tả
1	-	-	-	Protected ground: nối đất bảo vệ
2	3	TxD	DTE→DCE	Transmitted data: dữ liệu phát
3	2	RxD	DCE→DTE	Received data: dữ liệu thu
4	7	RTS	DTE→DCE	Request to send: DTE yêu cầu truyền dữ liệu
5	8	CTS	DCE→DTE	Clear to send: DCE sẵn sàng nhận dữ liệu
6	6	DSR	DCE→DTE	Data set ready: DCE sẵn sàng làm việc
7	5	GND	-	Ground: nối đất (0V)
8	1	DCD	DCE→DTE	Data carrier detect: DCE phát hiện sóng mang
20	4	DTR	DTE→DCE	Data terminal ready: DTE sẵn sàng làm việc
22	9	RI	DCE→DTE	Ring indicator: báo chuông
23	-	DSRD	DCE→DTE	Data signal rate detector: dò tốc độ truyền

Chuẩn RS-232C cho phép truyền tín hiệu với tốc độ đến 20.000 bps nhưng nếu cáp truyền đủ ngắn có thể lên đến 115.200 bps. Chiều dài cáp cực đại là 17-20m.

Các phương thức nối giữa DTE và DCE:

- Đơn công (simplex connection): dữ liệu chỉ được truyền theo 1 hướng.
- Bán song công (half-duplex): dữ liệu truyền theo 2 hướng, nhưng mỗi thời điểm chỉ được truyền theo 1 hướng.
 - Song công (full-duplex): số liệu được truyền đồng thời theo 2 hướng.

§ 4.2. Các phương pháp vào-ra dữ liệu

4.2.1. Truy xuất cổng nối tiếp dùng DOS và BIOS

Lệnh ngoại trú MODE của DOS có thể đặt các thông số cho cổng nối tiếp RS232.

Thí dụ: *MODE COM2:2400, E,8,1* chọn cổng COM2, tốc độ 2400 baud, parity chẵn, 8 bit dữ liệu và 1 bit stop. Cũng có thể dùng ngắt 21h của DOS để phát hoặc thu dữ liệu qua cổng nối tiếp bằng 4 hàm sau:

- Hàm 03h: đọc 1 ký tự
- Hàm 04h: phát 1 ký tự
- Hàm 3Fh: đọc 1 file
- Hàm 40h: ghi 1 file

BIOS cho phép truy xuất khối ghép nối nối tiếp qua ngắt 14h.

- Hàm 00h: khởi động khối ghép nối, định dạng dữ liệu, tốc độ truyền,....
- Hàm 01h, 02h: phát và thu 1 ký tự
- Hàm 03h: trạng thái của cổng nối tiếp
- Hàm 04h, 05h: mở rộng các điều kiện khởi động khối ghép nối, cho phép truy

xuất các thanh ghi điều khiển MODEM.

Byte trạng thái phát:

D7	D6	D5	D4	D3	D2	D1	D0
----	----	----	----	----	----	----	----

D7: lỗi quá thời gian (time-out); 1 = có lỗi; 0 = không lỗi;

D6: thanh ghi dịch phát; 1 = rỗng ; 0 = không rỗng

D5: thanh ghi đệm phát; 1 = rỗng; 0 = không rỗng

D4: ngắt đường truyền; 1= có ; 0 = không

D3: lỗi khung truyền SDU; 1 = có ; 0 = không

D2: lỗi chẵn lẻ; 1 = có ; 0 = không

D1: lỗi tràn; 1 = có ; 0 = không

D0: số liệu thu; 1 = có ; 0 = không

Byte trạng thái Modem

D7	D6	D5	D4	D3	D2	D1	D0
----	----	----	----	----	----	----	----

D7: phát hiện sóng mang; 1= phát hiện, 0 = không

D6: chỉ báo tín hiệu chuông; 1= có ; 0= không

D5: tín hiệu DTR; 1 = có , 0 = không

D4: tín hiệu CTS; 1 = có , 0 = không

D3: tín hiệu DDC, 1 = có , 0 = không

D2: tín hiệu delta RI; 1 = có, 0 = không

D1: tín hiệu delta DTR; 1 = có , 0 = không

D0: tín hiệu delta CTS; 1 = có , 0 = không

Thanh ghi DX chứa giá trị tương ứng với các cổng cần truy xuất (00h cho COM1,

01h cho COM2, 10h cho COM3, 11h cho COM4). Các thông số định dạng khung truyền

SDU được nạp vào thanh ghi AL theo nội dung như sau:

D7	D6	D5	D4	D3	D2	D1	D0
----	----	----	----	----	----	----	----

D7, D6, D5: tốc độ baud

000 = 110 baud

001 = 150 baud

010 = 300 baud

011 = 600 baud

100 = 1200 baud

101 = 2400 baud

110 = 4800 baud

111 = 9600 baud

D4-D3: bit parity

00= không có

01= lẻ

10 = không có

11= chẵn

D2: số bit stop

0 = 1 bit

1 = 2 bits

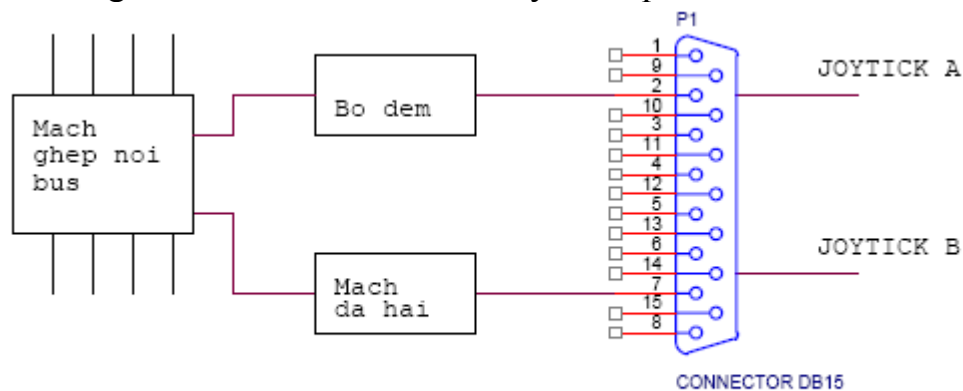
D1-D0: số bit số liệu

10 = 7 bits

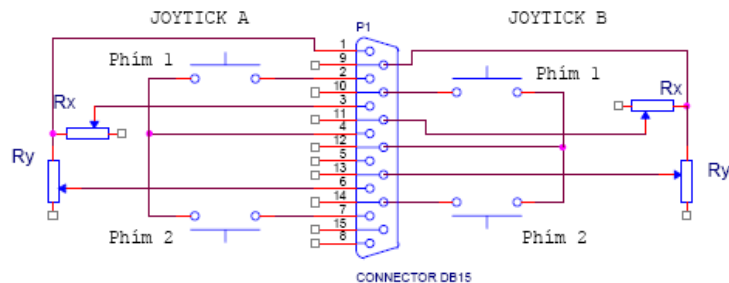
11= 8 bits

4.4.2. Giao tiếp PC Game

Cấu trúc và chức năng của board ghép nối trò chơi (PC game) như hình bên dưới. Bảng lệnh IN và OUT có thể truy xuất qua địa chỉ 201h.



Hình 5.6 - Cấu trúc và chức năng của board ghép nối trò chơi



Hình 5.7 - Cấu tạo của đầu nối 15 chân và 2 joystick A và B

Chân của đầu nối 15 chân	Sử dụng cho
2	Phím 1 của Joystick A (BA1)
3	Biến trở X của Joystick A
6	Biến trở Y của Joystick A
7	Phím 2 của Joystick A (BA2)
10	Phím 1 của Joystick B (BB1)
11	Biến trở X của Joystick B
13	Biến trở Y của Joystick B
14	Phím 2 của Joystick B (BB2)
1, 8, 9, 15	Vcc (+5V)
4, 5, 12	GND (0V)

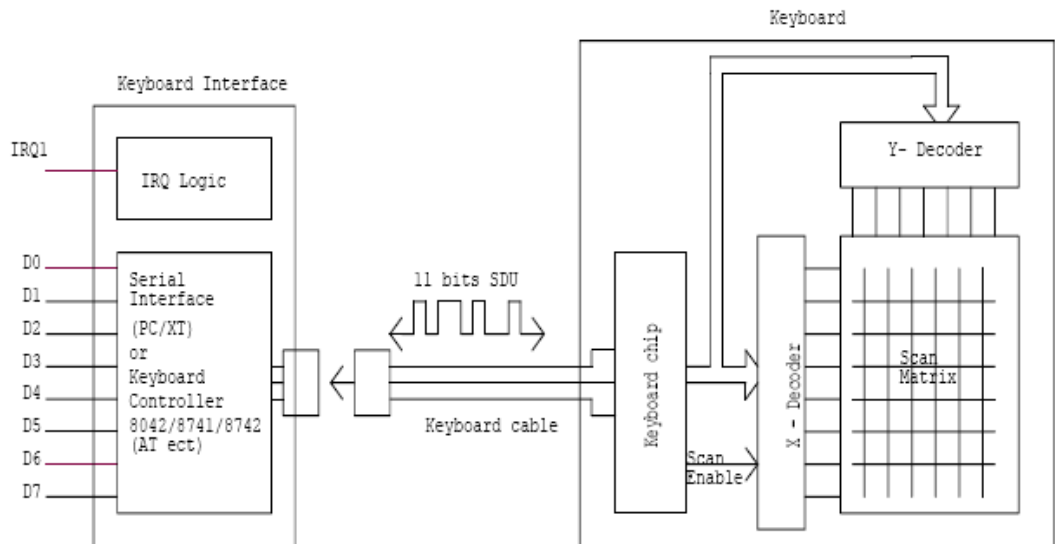
Board mạch được nối với bus hệ thống của PC chỉ qua 8 bits thấp của bus dữ liệu, 10 bits thấp của bus địa chỉ và các đường điều khiển IOR và IOW. Một đầu nối 15 chân được nối với board mạch cho phép nối cực đại hai thiết bị cho PC game gọi là joystick. Mỗi joystick có 2 biến trở có giá trị biến đổi từ 0 đến 100kΩ được đặt vuông góc với nhau đại diện cho vị trí x và y của joystick. Thêm nữa chúng có 2 phím bấm, thường là các công tắc thường hở phù hợp với các mức logic cao của các dây trên mạch. Có thể xác định được trạng thái nhấn hoặc nhả phím một cách dễ dàng bằng lệnh IN tới địa chỉ 201h. Nibble cao chỉ thị trạng thái của phím. Vì board không dùng đường IRQ do đó không có khả năng phát ra 1 ngắt, do vậy board chỉ hoạt động trong chế độ hỏi vòng (polling). Byte trạng thái của board game như sau:

D7	D6	D5	D4	D3	D2	D1	D0
BB ₂	BB ₁	BA ₂	BA ₁	BY	BX	AY	AX

BB₂, BB₁, BA₂, BA₁: Trạng thái của các phím B₂, B₁, A₂, A₁; 1 = nhả; 0 = nhấn
 BY, BX, AY, AX: Trạng thái của mạch đa hài tùy thuộc vào biến trở tương ứng.

4.4.3. Giao tiếp với bàn phím

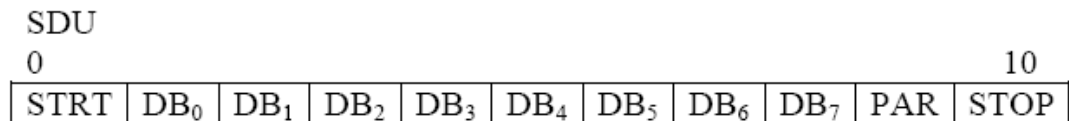
4.4.3.1. Bàn phím _ Cấu trúc và chức năng:



Hình 5.8 - Sơ đồ nguyên lý và các ghép nối của bàn phím

Chip xử lý bàn phím liên tục kiểm tra trạng thái của ma trận quét (scan matrix) để

xác định công tắc tại các tọa độ X,Y đang được đóng hay mở và ghi một mã tương ứng vào bộ đệm bên trong bàn phím. Sau đó mã này sẽ được truyền nối tiếp tới mạch ghép nối bàn phím trong PC. Cấu trúc của SDU cho việc truyền số liệu này và các chân cắm của đầu nối bàn phím.



STRT: bit start (luôn bằng 0)

DB0 - DB7: bit số liệu từ 0 đến 7.

PAR: bit parity (luôn lẻ)

STOP: bit stop (luôn bằng 1).

Tín hiệu xung nhịp dùng cho việc trao đổi dữ liệu thông tin nối tiếp đồng bộ với mạch ghép nối bàn phím (keyboard interface) trên main board được truyền qua chân số 1. Một bộ điều khiển bàn phím đã được lắp đặt trên cơ sở các chip 8042, hoặc 8742,8741.

Nó có thể được chương trình hóa (thí dụ khóa bàn phím) hơn nữa số liệu có

Chân 1: dữ liệu

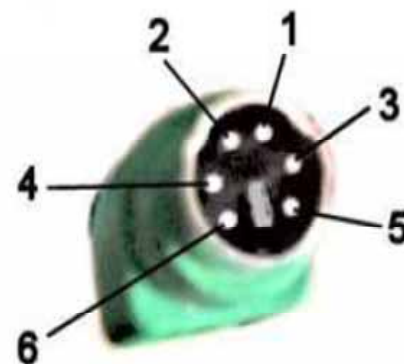
Chân 2: không dùng

Chân 3: GND

Chân 4: Vcc

Chân 5: clock

Chân 6: không dùng



Hình 5.10 – Đầu cắm bàn phím PS/2

thể truyền theo 2 hướng từ bàn phím và mạch ghép nối, do vậy vi mã của chip bàn phím có thể giúp cho việc nhận lệnh điều khiển từ PC, thí dụ như đặt tốc độ lặp lại của nhấn bàn phím,....

4.4.3.2. Mã quét bàn phím:

Mỗi phím nhấn sẽ được gán cho 1 mã quét (scan code) gồm 1 byte. Nếu 1 phím được nhấn thì bàn phím phát ra 1 mã make code tương ứng với mã quét truyền tới mạch ghép nối bàn phím của PC. Ngắt cứng INT 09h được phát ra qua IRQ1. Chương trình xử lý ngắt sẽ xử lý mã này tùy theo phím SHIFT có được nhấn hay không. Ví dụ: nhấn phím SHIFT trước, không rời tay và sau đó nhấn 'C': make code được truyền - 42 (SHIFT) - 46 ('C').

Nếu rời tay nhấn phím SHIFT thì bàn phím sẽ phát ra break code và mã này được

truyền như make code. Mã này giống như mã quét nhưng bit 7 được đặt lên 1, do vậy nó tương đương với make code cộng với 128. Tùy theo break code, chương trình con xử lý ngắt sẽ xác định trạng thái nhấn hay rời của các phím. Thí dụ, phím SHIFT và 'C' được rời theo thứ tự ngược lại với thí dụ trên: break code được truyền 174 (bằng 46 cộng 128 tương ứng với 'C') và 170 (bằng 42 cộng 128 tương ứng với SHIFT). Phần cứng và phần mềm xử lý bàn phím còn giải quyết các vấn đề vật lý sau:

- Nhấn và thả phím nhưng không được phát hiện.
- Khử nhiễu rung cơ khí và phân biệt 1 phím được nhấn nhiều lần hay được nhấn chỉ 1 lần nhưng được giữ trong một khoảng thời gian dài.

4.4.3.3. Truy xuất bàn phím qua BIOS

BIOS ghi các ký tự do việc nhấn các phím vào bộ đệm tạm thời được gọi là bộ đệm bàn phím (keyboard buffer), có địa chỉ 40:1E, gồm 32 byte và do vậy kết thúc ở địa chỉ 40:3D. Mỗi ký tự được lưu trữ bằng 2 bytes, byte cao là mã quét, và byte thấp là mã ASCII. Như vậy, bộ đệm có thể lưu trữ tạm thời 16 ký tự. Chương trình xử lý ngắt sẽ xác định mã ASCII từ mã quét bằng bảng biến đổi và ghi cả 2 mã vào bộ đệm bàn phím. Bộ đệm bàn phím được tổ chức như bộ đệm vòng (ring buffer) và được quản lý bởi 2 con trỏ. Các giá trị con trỏ được lưu trữ trong vùng số liệu của BIOS ở địa chỉ 40:1A và 40:1C. Ngắt INT 16h trong BIOS cung cấp 8 hàm cho bàn phím. Thường các hàm BIOS trả về một giá trị 0 của ASCII nếu phím điều khiển hoặc chức năng được nhấn..

Các thí dụ:

- Giả sử phím 'a' đã được nhấn.

MOV AH,00h ; chạy hàm 00h, đọc ký tự

INT 16h ; phát một interrupt

Kết quả: AH = 30 (mã quét cho phím 'a'); AL = 97 (ASCII cho 'a')

- Giả sử phím '.HOME' đã được nhấn.

MOV AH,00h ; chạy hàm 00h, đọc ký tự

INT 16h ; phát một interrupt

Kết quả: AH = 71 (mã quét cho phím 'HOME')

AL = 00 (các phím chức năng và điều khiển không có mã ASCII)

- Giả sử phím 'HOME' đã được nhấn.

MOV AH,10h ; chạy hàm 10h, đọc ký tự

INT 16h ; phát một interrupt

Kết quả: AH = 71 (mã quét cho phím 'HOME')

AL = E0h

4.4.3.4. Chương trình với bàn phím qua các cổng:

Bàn phím cũng là một thiết bị ngoại vi nên về nguyên tắc có thể truy xuất nó qua các cổng vào ra.

Các thanh ghi và các port:

Sử dụng 2 địa chỉ port 60h và 64h có thể truy xuất bộ đệm vào, bộ đệm ra và thanh ghi điều khiển của bàn phím.

Port	Thanh ghi	R/W
60h	Đệm ngõ ra	R
60h	Đệm ngõ vào	W
64h	Thanh ghi điều khiển	W
64h	Thanh ghi trạng thái	R

Thanh ghi trạng thái xác định trạng thái hiện tại của bộ điều khiển bàn phím. Thanh ghi này chỉ đọc (read only). Có thể đọc nó bằng lệnh IN tại port 64h.

7							0
PARE	TIM	AUXB	KEYL	C/D	SYSF	INPB	OUTB

PARE: Lỗi chặn lẻ của byte cuối cùng được vào từ bàn phím; 1 = có lỗi chặn lẻ, 0 = không có.

TIM: Lỗi quá thời gian (time-out); 1 = có lỗi, 0 = không có.

AUXB: Đệm ra cho thiết bị phụ (chỉ có ở máy PS/2); 1 = giữ số liệu cho thiết

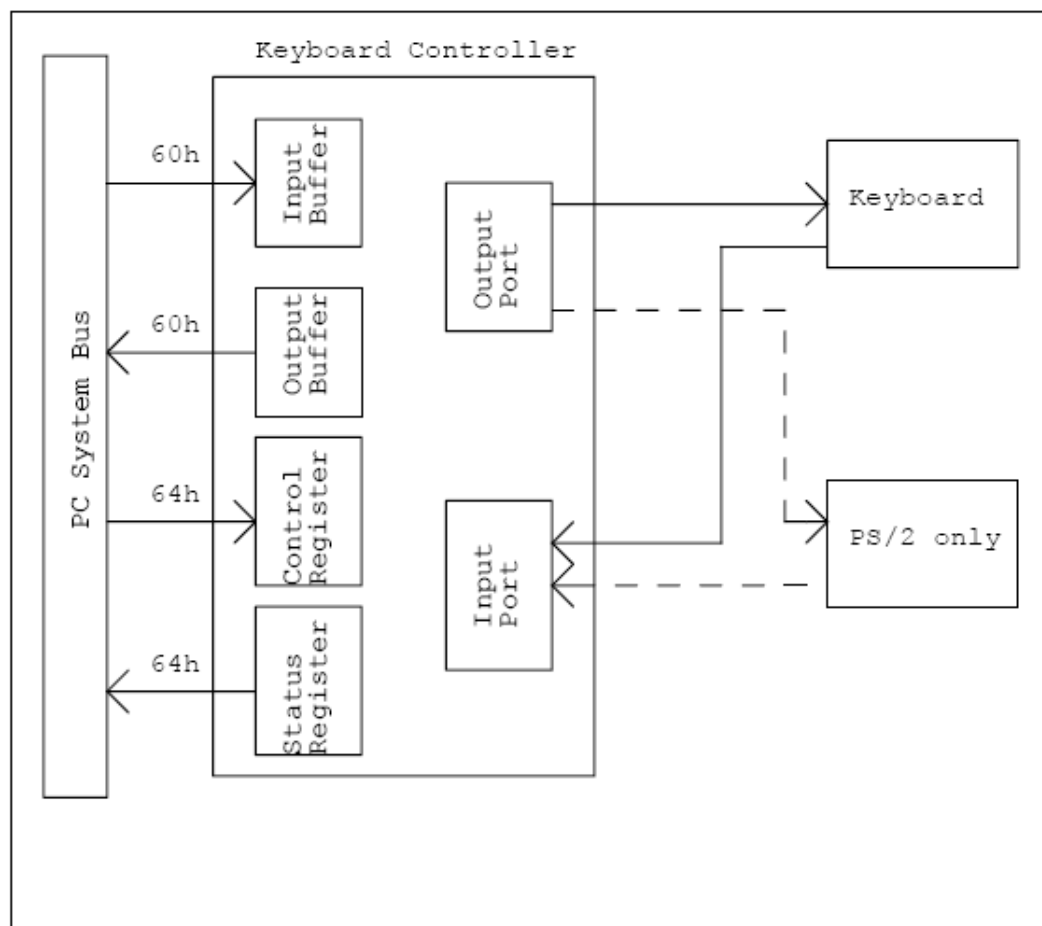
bị, 0 = giữ số liệu cho bàn phím.

KEYL: Trạng thái khóa bàn phím; 1 = không khóa, 0 = khóa.

C/D: Lệnh/số liệu; 1 = Ghi qua port 64h, 0 = Ghi qua port 60h.

INPB: Trạng thái đệm vào; 1 = số liệu CPU trong bộ đệm vào, 0 = đệm vào rỗng.

OUTB: Trạng thái đệm ra; 1 = số liệu bộ điều khiển bàn phím trong bộ đệm ra, 0 = đệm ra rỗng.



Hình 5.11 - Bộ điều khiển bàn phím

Thanh ghi điều khiển (64h)

7							0
C ₇	C ₆	C ₅	C ₄	C ₃	C ₂	C ₁	C ₀

Các lệnh cho bộ điều khiển bàn phím:

Mã Lệnh

A7h Cấm thiết bị phụ, A8h Cho phép thiết bị phụ, A9h Kiểm tra ghép nối tới thiết bị phụ

AAh Tự kiểm tra, ABh Kiểm tra ghép nối bàn phím, ADh Cấm bàn phím, AEh Cho phép bàn phím

C0h Đọc cổng vào

C1h Đọc cổng vào ra (byte thấp)

C2h Đọc cổng vào ra (byte cao)

D0h Đọc cổng ra

D1h Ghi cổng ra

D2h Ghi đệm ra bàn phím

D3h Ghi đệm ra thiết bị phụ

D4h Ghi thiết bị phụ

E0h Kiểm tra đọc cổng vào

F0h Gửi 1 xung tới lỗi ra

FFh Công

Khóa bàn phím:

Start:

IN AL, 64h ; đọc byte trạng thái

TEST AL, 02h ; kiểm tra bộ đệm có đầy hay không

JNZ start ; một vài byte vẫn còn trong bộ đệm vào

OUT 64h, 0ADh ; khóa bàn phím

Các lệnh cho bàn phím:

Tóm tắt các lệnh bàn phím:

Mã	Lệnh	Mô tả
EDh	Bật ON/OFF LED	Bật/tắt các đèn led của bàn phím
EEh	Echo	Trả về byte eeh
F0h	Đặt/nhận diện	Đặt 1 trong 3 mã quét và nhận diện các mã quét tập mã quét hiện tại.
F2h	Nhận diện bàn phím	Nhận diện ACK = AT, ACK+abh+41h=MF II.
F3h	Đặt tốc độ lặp lại/trễ	Đặt tốc độ lặp lại và thời gian trễ của bàn phím
F4h	Enable	Cho phép bàn phím hoạt động
F5h	Chuẩn/không cho phép	Đặt giá trị chuẩn và cấm bàn phím.
F6h	Chuẩn/cho phép	Đặt giá trị chuẩn và cho phép bàn phím.
FEh	Resend	Bàn phím truyền ký tự cuối cùng một lần nữa tới bộ điều khiển bàn phím
FFh	Reset	Chạy reset bên trong bàn phím

Thí dụ: lệnh bật đèn led cho phím NUMCLOCK, tắt tất cả các đèn khác.

OUT 60H, EDH ; ra lệnh cho bật tắt các đèn led

WAIT:

IN AL, 64H ; đọc thanh ghi trạng thái

JNZ WAIT ; bộ đệm vào đầy

OUT 60H, 02H ; bật đèn cho numclock

Cấu trúc của byte chỉ thị như sau:

Cấu trúc của byte chỉ thị như sau:

7	6	5	4	3	2	1	0
0	0	0	0	0	CPL	NUM	SCR
CPL:					1 = bật đèn Caps Lock;	0 = tắt	
NUM:					1 = bật đèn Num Lock;	0 = tắt	
SCR:					1 = bật đèn Scr Lock;	0 = tắt	

4.4.4 AGP - Accelerated Graphics Port

4.4.4.1 Nguyên lý chung

Các hình ảnh mà chúng ta thấy được trên màn hình máy tính được tạo bởi rất

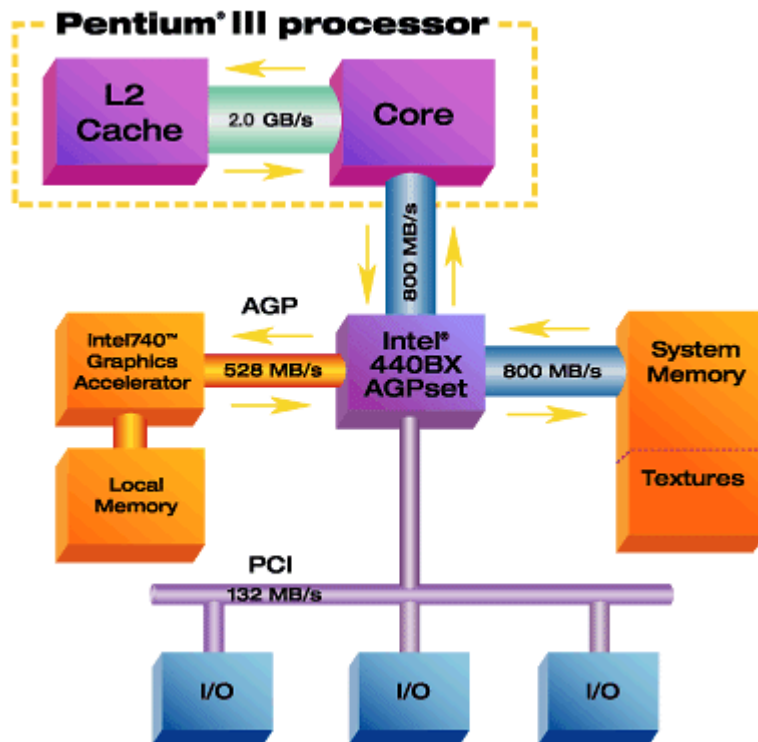
nhiều điểm ảnh gọi là pixel. Trong hầu hết các thiết lập cho độ phân giải thì màn hình thường hiển thị khoảng hơn 1 triệu điểm ảnh. Máy tính sẽ quyết định cần phải làm gì theo thứ tự đối với từng điểm ảnh để tạo ra một hình ảnh. Để có thể làm được việc này, nó sử dụng một bộ chuyển đổi, lấy các dữ liệu nhị phân từ CPU và chuyển chúng thành hình ảnh hiển thị trên màn hình. Khi CPU nhận được yêu cầu xem một hình ảnh từ phía người sử dụng, nó sẽ chuyển yêu cầu này tới card đồ họa để quyết định sẽ dùng những pixel nào hiển thị hình ảnh. Sau đó nó sẽ gửi những thông tin để màn hình hiển thị thông qua dây cáp.

Quá trình tạo ra những hình ảnh không phải là dữ liệu nhị phân thường đòi hỏi quá trình xử lý phức tạp hơn rất nhiều. Để có thể vẽ ra một hình ảnh 3D, card đồ họa phải tạo ra một khung điện từ, sau đó quét hình ảnh và thêm vào đó ánh sáng, màu. Đối với trò chơi có nhiều hình ảnh 3D, máy tính phải lặp lại quá trình này khoảng 60 lần mỗi giây.

Như các thành phần khác của máy tính, Graphic Card AGP được ưu tiên kết nối với CPU qua Bus. Về cơ bản, Bus được hiểu như kênh truyền hay đường nối giữa các thành phần trong máy tính. Do AGP được xây dựng dựa trên các chuẩn PCI Bus và được coi như một AGP Bus nên nó là một dạng kết nối điểm (Point to Point). Nói cách khác chỉ có một thiết bị kết nối giữa AGP với CPU và bộ nhớ, đó là Graphic Card và do vậy nó thực sự nó không phải là một Bus. AGP có hai cải tiến so với PCI là tốc độ nhanh hơn và truy xuất trực tiếp tới bộ nhớ hệ thống. AGP sử dụng các công nghệ sau để đạt được tốc độ nhanh hơn:

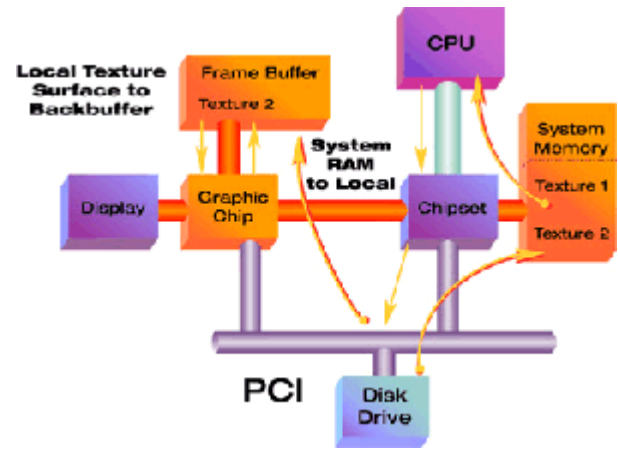
- AGP là một Bus 32 bit với xung nhịp 66 MHz. Điều đó có nghĩa là trong một giây nó có thể truyền tải một lượng thông tin có độ lớn 32 Bit (4 Byte) đến 66 triệu lần. Tốc độ truyền tải sẽ tăng lên khi nó hoạt động ở chế độ 2X và 4X.
- Không có thiết bị nào khác trên máy tính sử dụng AGP Bus, do vậy Graphic Card sẽ không phải chia sẻ Bus với các thiết bị khác và luôn hoạt động với khả năng kết nối tối đa.
- AGP sử dụng Pipelining để tăng tốc. Pipelining tổ chức việc thu hồi dữ liệu theo trình tự và Graphic Card nhận được các đoạn dữ liệu hoàn trả lại các yêu cầu đơn lẻ.

AGP sử dụng Sideband Addressing cho phép Graphic Card đưa ra các yêu cầu và phân bổ các thông tin địa chỉ sử dụng 8 Bit trong số 32 Bit dùng để truyền dữ liệu.

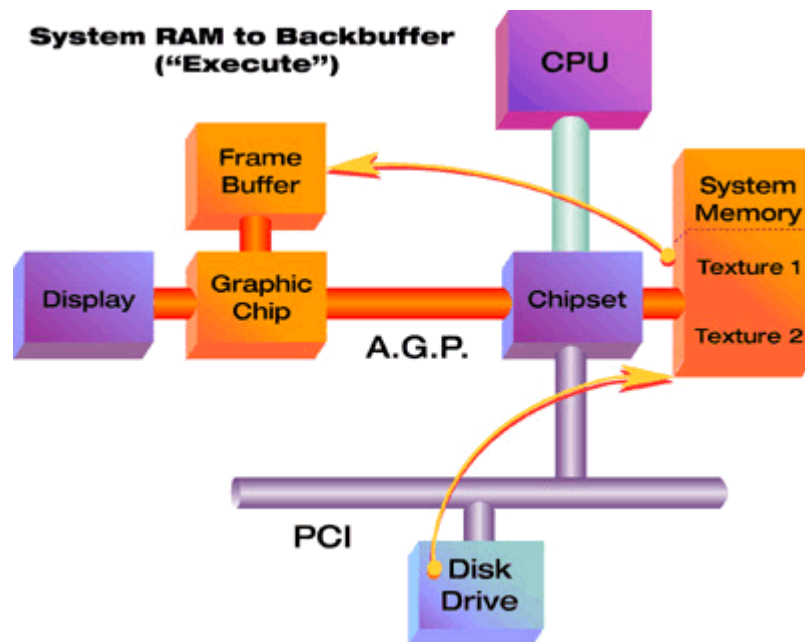


Bên cạnh cải tiến về tốc độ, một cải tiến nữa của AGP-based Graphic Card so với PCI là khả năng truy xuất trực tiếp tới bộ nhớ hệ thống qua AGP Bus với tốc độ tối đa. Đây là một thành phần rất quan trọng của AGP. Bảng lưu kết cấu (Texture Map) là chìa khoá quan trọng trong đồ hoạ máy tính, nó chiếm một lượng tương đối lớn bộ nhớ ở các Graphic Card thông thường. Do Video RAM thường đòi hỏi tương đối lớn trong khi lại bị hạn chế bởi dung lượng Graphic Card nên số lượng và độ lớn của Texture Map cũng bị giới hạn gần bằng dung lượng Graphic Card. Hệ thống AGP-based thuận lợi hơn ở chỗ có thể sử dụng bộ nhớ hệ thống để lưu trữ các Texture Map và các dữ liệu khác mà vẫn thường phải lưu ở Video RAM trên Card.

Trong các hệ thống không hỗ trợ AGP chẳng hạn như PCI-based Graphic Card, mọi Texture Map đều được lưu hai lần. Lần thứ nhất nó được nạp từ đĩa cứng lên bộ nhớ hệ thống. Sau đó nó được đọc từ bộ nhớ hệ thống ra để CPU xử lý rồi được gửi trả lại qua PCI Bus và lưu trên Framebuffer của Graphic Card. Kết quả là mọi Texture Map đều được xử lý và lưu hai lần, một lần bởi hệ thống và một lần bởi Graphic Card.



AGP chỉ lưu các Texture Map một lần với Chip GART (Graphic Address Remapping Table). GART sẽ phân bổ các phần bộ nhớ hệ thống để lưu giữ các Texture Map nhưng luôn làm CPU và Graphic Card lầm tưởng rằng các Texture Map được lưu trên Framebuffer của Card. GART có thể lưu kiểm soát các Bit của Texture Map cho dù



chúng được lưu ở những vùng khác nhau trên bộ nhớ hệ thống nhưng lại được thể hiện như một đoạn bộ nhớ liên tục trên Graphic Card. Trong trường hợp sử dụng non-AGP Card, mỗi Texture Map đều bị lưu thành hai lần dẫn đến CPU phải làm việc nhiều hơn. Đây chính là những hạn chế của non-AGP Card so với các AGP-based Card. AGP chỉ lưu các Texture Map một lần với Chip GART (Graphic Address Remapping Table). GART sẽ phân bổ các phần bộ nhớ hệ thống để lưu giữ các Texture Map nhưng luôn làm CPU và Graphic Card lầm tưởng rằng các Texture Map được lưu trên Framebuffer

của Card. GART có thể lưu kiểm soát các Bit của Texture Map cho dù chúng được lưu ở những vùng khác nhau trên bộ nhớ hệ thống nhưng lại được thể hiện như một đoạn bộ nhớ liên tục trên Graphic Card. Trong trường hợp sử dụng non-AGP Card, mỗi Texture Map đều bị lưu thành hai lần dẫn đến CPU phải làm việc nhiều hơn. Đây chính là những hạn chế của non-AGP Card so với các AGP-based Card.

Hiện tại có 3 thế hệ AGP 1.0, AGP 2.0 và AGP Pro. AGP 2.0 được xây dựng trên phiên bản AGP 1.0 cung cấp 3 chế độ hoạt động. Các chế độ này đều chạy với tốc độ 66 MHz qua AGP Bus. Đối với 2X AGP, Graphic Card gửi dữ liệu 2 lần sau mỗi xung nhịp còn ở chế độ 4X AGP nó sẽ gửi dữ liệu 4 lần sau mỗi xung nhịp.

Chế độ	Xung nhịp	Tốc độ truyền
1x	66 MHz	266 MBps
2x	133 MHz	533 MBps
4x	266 MHz	1,066 MBps

4.4.5. PCI EXPRESS

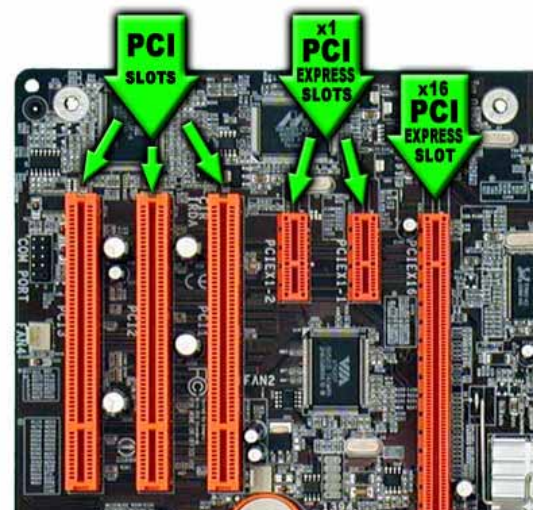
PCI Express, viết tắt là PCIe là một dạng giao diện bus hệ thống/card mở rộng của máy tính. Nó là một giao diện nhanh hơn nhiều và được thiết kế để thay thế giao diện PCI, PCI-X (PCI Extended) , và AGP cho các card mở rộng và card đồ họa. Khe cắm PCI Express (PCIe) hoàn toàn khác so với các chuẩn trước như PCI hay PCI Extended (PCI-X).

-Nhưng PCI có một vài hạn chế . Những CPU , Card màn hình , Card âm thanh và những Card mạng ngày càng nhanh hơn và mạnh hơn trong khi đó PCI cố định độ rộng dữ liệu 32-bit và chỉ có thể điều khiển 05 thiết bị trong cùng một lúc .

-Một giao thức mới gọi là PCI Express (PCIe) đã giải quyết được những hạn chế trên , cung cấp băng thông lớn hơn , tương thích với những hệ điều hành đang có .

4.4.5.1. Kết nối nối tiếp tốc độ cao :

Ngay từ khi ra đời của máy tính , việc cần thiết để trao đổi dữ liệu vô cùng lớn . Trong kết nối nối tiếp máy

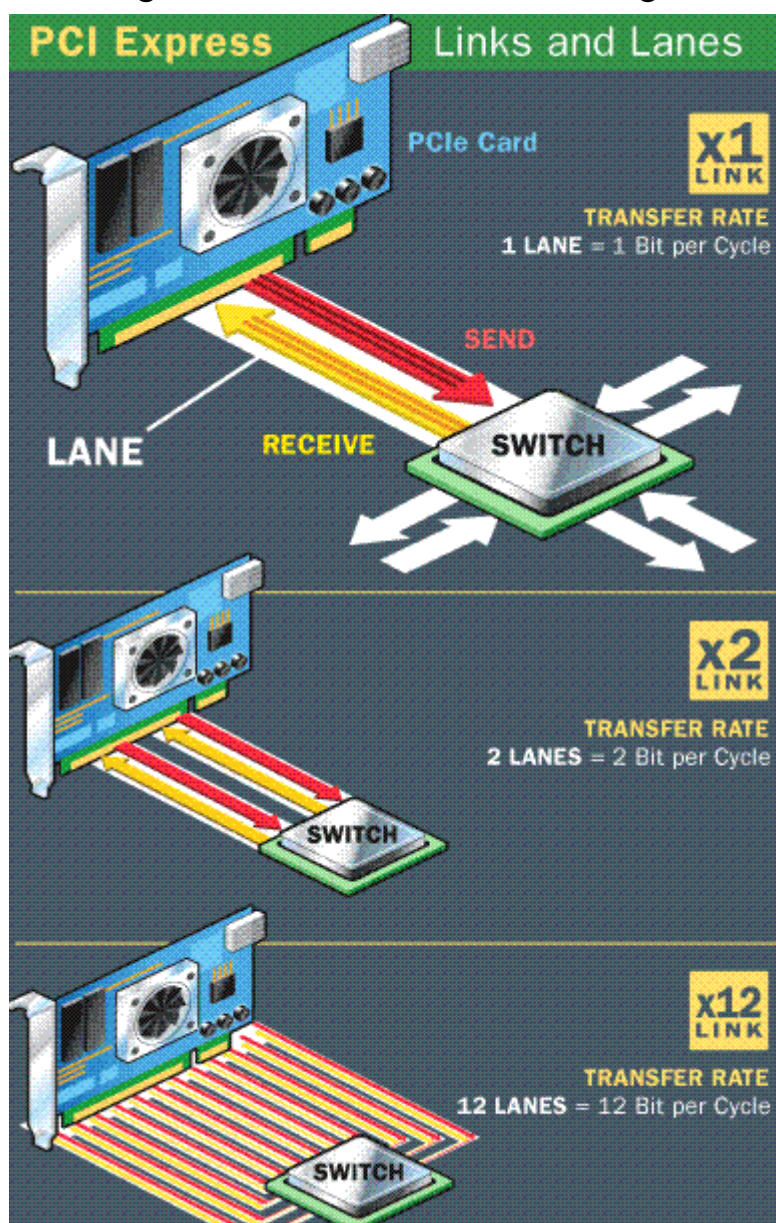


tính tách dữ liệu thành những nhóm và chuyển từng gói dữ liệu đi một, hết gói này rồi đến gói kia. Kết nối như thế trong thời điểm ban đầu của kỷ nguyên máy tính có tốc độ chậm, do đó nhiều nhà sản xuất bắt đầu chuyển sang dùng kết nối song song để gửi nhiều mẫu dữ liệu đi cùng một lúc.

Một vấn đề xảy ra khi những kết nối song song đạt tới tốc độ cao nào đó thì những dây dẫn cạnh nhau gây ảnh hưởng qua lại với nhau, do dòng điện đi qua dây dẫn tạo nên môi trường xung quanh nó một từ trường. Với cường độ từ trường một mức độ nào đó sẽ ảnh hưởng lên dây dẫn bên cạnh làm sai lệch tín hiệu bên trong một dây dẫn khác và ngược lại. Điều này đã xảy ra đối với Cable ATA 133. Do đó với tín hiệu truyền song song chỉ có thể đạt được một tốc độ cao nhất định. Để truyền tín hiệu song song với tốc độ cao không ảnh hưởng tới tín hiệu sang nhau đòi hỏi thiết kế lại hệ thống Bus có mức độ lọc nhiễu cao lúc đó lại ảnh hưởng tới giá thành của thiết bị.

PCIe là kết nối nối tiếp mà hoạt động như là mạng hơn là Bus. Thay vì một Bus mà điều khiển dữ liệu từ nhiều nguồn. PCIe có Switch điều khiển và kết nối Point-to-Point. Những kết nối này do Switch mang đến, hướng dữ liệu trực tiếp tới thiết bị cần đến. Mọi thiết bị có kết nối riêng của nó, do đó những thiết bị không mất thời gian chia sẻ băng thông như Bus bình thường khác.

Khi máy tính khởi động lên, PCIe xác định những thiết bị nào được cắm bên trong Mainboard. Sau đó nó nhận dạng



những liên kết giữa các thiết bị và tạo một bản đồ cho biết dữ liệu chuyển động ở đâu sẽ đi và phân chia độ rộng của mỗi liên kết . Sự nhận dạng của những thiết bị này và những kết nối là dùng cùng một giao thức PCI , do đó PCIe không cần thay đổi phần mềm hoặc những hệ điều hành.

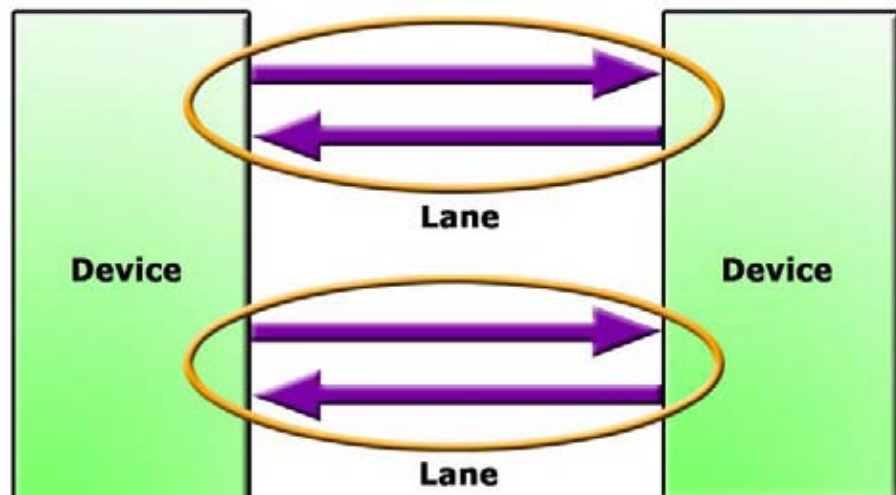
4.4.5.2. Vấn đề băng thông :

Hiện thời, PCI Express được chia làm nhiều loại ứng với từng tốc độ truyền tải dữ liệu khác nhau là: 1x, 2x, 4x, 8x, 12x, 16x (và cả 32x), tất cả đều có băng thông lớn hơn nhiều so với chuẩn PCI cũ. Trong đó loại 4x, 8x và 12x sử dụng trong thị trường máy chủ, còn 1x, 2x và 16x thì sử dụng cho người dùng thông thường. Bảng bên cạnh so sánh các loại này với nhau và với các chuẩn truyền tải dữ liệu khác:

PCI	32 MB/giây
AGP 8X	2,100 MB/giây
PCI Express 1x	250 [500]* MB/giây
PCI Express 2x	500 [1000]* MB/giây
PCI Express 4x	1000 [2000]* MB/giây
PCI Express 8x	2000 [4000]* MB/giây
PCI Express 12x	3000 [64000]* MB/giây
PCI Express 16x	4000 [8000]* MB/giây
PCI Express 32x	8000 [16000]* MB/giây
IDE (ATA100)	100 MB/giây
IDE (ATA133)	133 MB/giây
SATA	150 MB/giây
Gigabit Ethernet	125 MB/giây
IEEE1394B [firewire]	100 MB/giây





Lưu ý: vì PCI Express là công nghệ dựa trên nền tảng tương tự (serial) nên dữ liệu có thể truyền tải qua bus theo hai hướng, do đó con số trong bảng sau là băng thông tổng cộng theo cả hai hướng.

-Mỗi đường (lane) của kết nối PCIe gồm hai cặp dây, một để truyền dữ liệu và một để gửi dữ liệu. Những gói dữ liệu di chuyển trong Lane với tốc độ 1bit/chu kì. Và kết nối x1 là kết nối nhỏ nhất trong kết nối PCIe, như vậy một Lane có 04 dây dẫn, mang 1bit/chu kì theo mỗi hướng. Kết nối x2 gồm 08 dây dẫn và truyền 2 bit một lúc, kết nối x4 truyền 4 bit và cứ như thế. Những cấu hình khác là x12 , x16 và x32 .

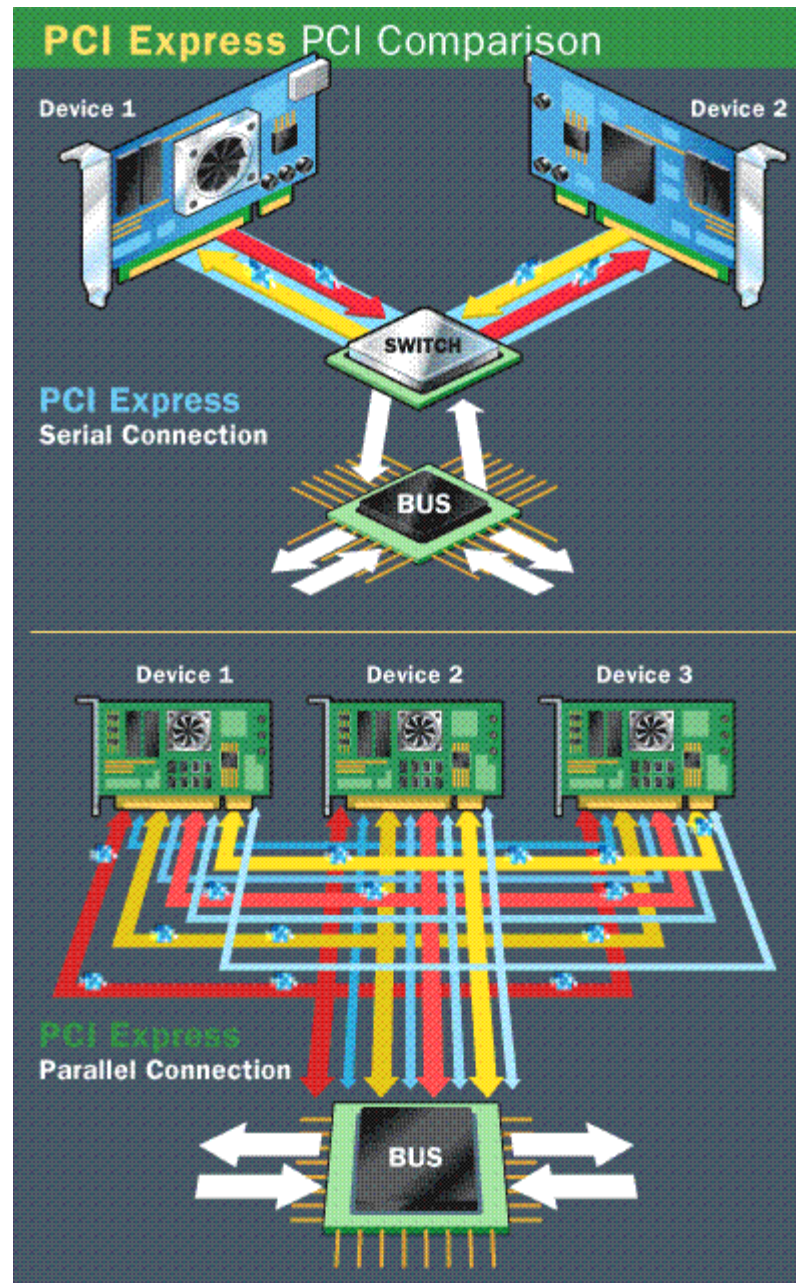


4.4.5.3. Tốc độ nhanh hơn

Bus PCI có độ rộng 32-bit , tốc độ xung nhịp đồng hồ cao nhất là 33MHz , cho phép dữ liệu cao nhất truyền 133MB/s . Bus PCI-X có độ rộng 64-bit , rộng gấp đôi so với Bus PCI . Những tính năng khác nhau của PCI-X cho

PCI Express Example Connectors	
x1	BANDWIDTH Single direction: 2.5 Gbps/200 MBps Dual Directions: 5 Gbps/400 MBps 
x4	BANDWIDTH Single direction: 10 Gbps/800 MBps Dual Directions: 20 Gbps/1.6 GBps 
x8	BANDWIDTH Single direction: 20 Gbps/1.6 GBps Dual Directions: 40 Gbps/3.2 GBps 
x16	BANDWIDTH Single direction: 40 Gbps/3.2 GBps Dual Directions: 80 Gbps/6.4 GBps 

phép tốc độ truyền dữ liệu lên tới từ 512MB tới 1GB/s



Một Lane trong kết nối PCIe có thể truyền dữ liệu lên tới 200MB/s cho mỗi hướng . PCIe 16x có thể gây kinh ngạc khi lên tới 64.GB/s cho mỗi hướng . Với tốc độ kết nối x1 có thể dễ dàng điều khiển kết nối Gigabit Ethernet , âm thanh và những ứng dụng lưu trữ . Kết nối x16 có thể dễ dàng điều khiển sức mạnh của Card màn hình .

Những điều kiện thuận lợi khi chuyển tốc độ kết nối nối tiếp :

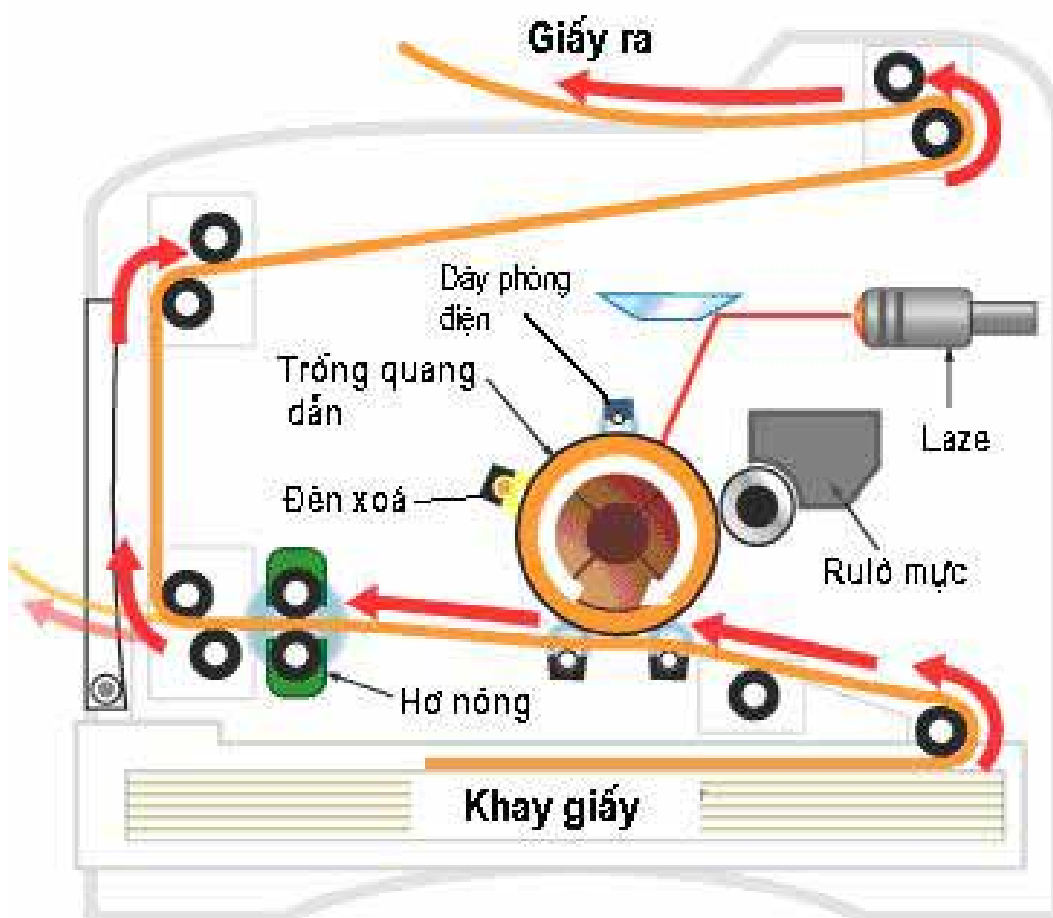
- Ưu tiên dữ liệu , điều này cho phép hệ thống di chuyển hầu hết những dữ liệu quan trọng đầu tiên và ngăn chặn hiện tượng kiểu nghẽn mạch cổ chai .
- Dữ liệu được truyền theo thời gian thực .
- Sử dụng ít chân cắm hơn do độ rộng dữ liệu nhỏ hơn Bus thông thường .
- Dễ dàng kết nối và dò tìm lỗi .
- Đơn giản hơn để ngắt dữ liệu thành những gói nhỏ và đặt những gói nhỏ cùng với nhau . Mỗi một thiết bị có những đường dữ liệu riêng do kết nối Point-to-Point từ Switch , tín hiệu từ nhiều nguồn không mất thời gian làm việc trên cùng một Bus .

CHƯƠNG 5: CÁC THIẾT BỊ NGOẠI VI

§ 5.1. Các thiết bị nhập, xuất dữ liệu

5.1.1. Cách hoạt động của một máy in laser:

Các máy in laser hoạt động bằng cách đặt mực toner (*toner*: chất mực dạng bột có khả năng tích điện) trên một trống quay (*drum*) được tích điện, rồi sau đó chuyển mực toner lên giấy in khi tờ giấy này dịch chuyển qua hệ thống ở cùng một tốc độ với trống quay. Hình 5.1.1 cho ta thấy sáu bước tuần tự của tiến trình in trong máy in laser. Bốn bước đầu tiên sẽ sử dụng các thành phần máy in vốn chịu đựng sự hao mòn nhiều nhất, tức các thành phần được chứa bên trong hộp tháo ra được (*cartridge*). Việc chứa đựng các thành phần này bên trong một hộp cartridge sẽ khiến máy in bền hơn. Hai bước sau cùng được thực hiện bên ngoài hộp cartridge. Các thủ tục in laser trong hình 5.1.1 như sau:



Hình 5.1.1 Sáu bước liên tục của việc in ấn trên máy in laser

1. Làm sạch : Mực toner còn sót lại và điện tích sẽ được lấy ra khỏi trống.

2. Chuẩn bị : Trống được nạp một điện tích cao.

3.Ghi : Một tia laser được sử dụng để giảm điện tích cao xuống một điện tích thấp hơn, chỉ ở những nơi mà mực toner sẽ bám vào.

4.Triển khai : Mực toner được đặt vào trống tại những nơi điện tích đã được giảm thấp xuống.

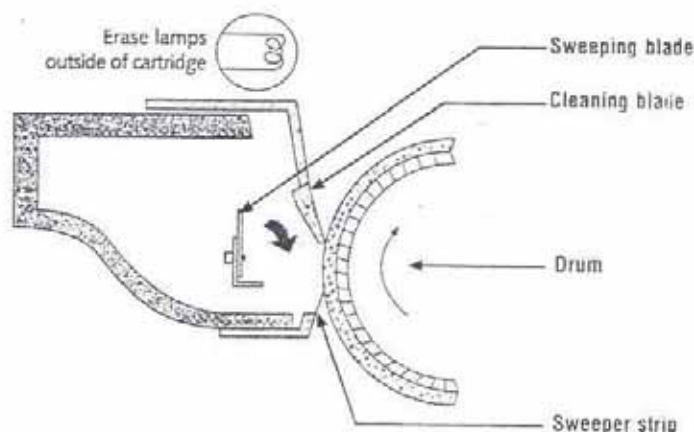
5.Chuyển giao: Một điện tích mạnh sẽ hút mực toner từ trống lên giấy. Đây là bước đầu tiên diễn ra bên ngoài hộp cartridge.

6.Nung chảy: Sức nóng và áp suất được sử dụng để nung chảy mực toner trên giấy.

Lưu ý rằng hình 5.1.1 chỉ cho ta thấy mặt cắt của trống, các cơ cấu và giấy in. Khi hình dung tiến trình này, cần nhớ rằng trống quay có chiều rộng bằng với chiều rộng của giấy in. Gương phản chiếu (*mirror*), thanh gạt mực và các trục lăn trong hình này cũng có chiều rộng bằng với chiều rộng của tờ giấy in. Trước hết bạn hãy để ý vị trí của hộp cartridge trong hình vẽ, trống cảm quang quay theo chiều kim đồng hồ nằm bên trong cartridge và đường đi của tờ giấy in, vốn di chuyển qua hình vẽ từ phải sang trái

Bước 1: Làm sạch.

Trước hết các thang gạt (*blade*) sẽ chùi sạch mực toner còn sót lại trên trống. Kế đó, các đèn xóa (*erase lamp*, được đặt bên ngoài hộp cartridge) sẽ khử điện tích cho trống bằng cách chiếu ánh sáng lên bề mặt của trống để trung hòa (*neutralize*) bất kỳ điện tích nào còn sót lại trên trống.



Bước làm sạch sẽ sạch mực toner và khử điện tích còn sót lại trên trống.

Bước 2: Chuẩn bị.

Bước chuẩn bị sẽ đặt một điện tích đồng nhất -600v lên trống. Điện tích này được đặt lên trống bởi một dây dẫn thiết bị điện hóa chính (*primary corona wire*) vốn được nạp điện bởi một bộ nguồn cung cấp điện thế cao. (Một *thiết bị điện hóa* (*corona*) là một thiết bị có khả năng tạo điện tích). Trong hình 5.1.1, ta có thể thấy thiết bị điện hóa chính (*primary corona*) nằm giữa dây dẫn thiết bị điện hóa chính và trống quay, nó sẽ điều hòa điện tích trên trống quay để đảm bảo rằng điện tích này đồng nhất ở mức -600v.

Bước 3: Ghi.

① Formatter

② DC Controller

③ Laser unit

Scanning mirror

Scanning motor

Optical fiber

Focusing lens

Beam detect mirror

Mirror

-100V charge where laser strikes

-600V charge where laser does not strikes

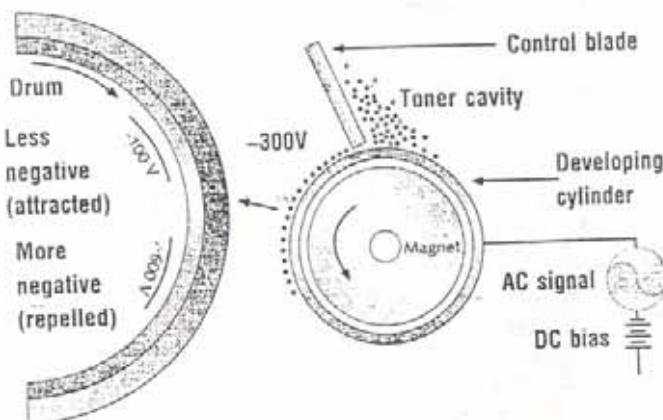
Tốc độ của mô-tơ quay trống và tốc độ của mô-tơ quét quay gương quét được đồng bộ hóa sao cho tia laser hoàn tất một đường quét (*scanline*) dọc theo trống rồi quay trở lại phần đầu của trống này để bắt đầu một đường quét mới, nhằm đạt được quét thích hợp cho mỗi inch của chu vi trống. Ví dụ đối với một máy in 300 dpi (dots per inch: số lượng điểm ảnh trên mỗi inch), tia laser sẽ quét 300 lượt cho mỗi inch của chu vi trống. Tia laser được bật và tắt liên tục khi nó thực hiện một quét đơn theo số chiều dài của trống, để các điểm (*dot*) được ghi dọc theo trống trên mỗi lượt quét. Đối với một máy in 300 dpi, 300 điểm sẽ được ghi dọc theo trống cho mỗi inch của chiều dài trống. 300 điểm trên mỗi inch chiều dài, cùng với 300 lượt quét cho mỗi inch của chu vi trống, hợp thành độ phân giải 300×300 điểm trên mỗi inch vuông của chiều máy in laser để bàn.

đầu quét dọc theo trống, gương phát hiện tia (*beam detect mirror*, xem hình 5.1.1) sẽ phát hiện sự hiện diện ban đầu của tia laser bằng cách phản xạ tia này vào một sợi quang (*optical fiber*). Tia sáng này sẽ đi dọc theo sợi quang để tới bộ kiểm soát DC (*DC controller*) và tại đó nó sẽ được chuyển đổi thành một tín hiệu điện được dùng để đồng bộ hóa kết xuất dữ liệu. Tín hiệu này được dùng để chẩn đoán các sự cố với tia laser hoặc mô-tơ quét.

Tia laser đã ghi một hình ảnh lên bề mặt trống ở dạng các vùng mang điện tích -100V . Điện tích -100V trên vùng hình ảnh này sẽ được sử dụng trong giai đoạn triển khai để chuyển mực toner sang bề mặt trống.

Bước 4: Triển khai.

Hình 5.1.3 cho ta thấy rõ hơn về bước triển khai, trong đó mực toner được trực lăn triển khai (*developing cylinder*) áp vào các vùng mang điện tích -100V trên bề mặt trống. Mực toner sẽ di chuyển từ trực lăn sang trống khi cả hai quay rất gần nhau. Trực lăn được bao phủ bởi một lớp mực toner, vốn được chế tạo từ nhựa thông đen liên kết với sắt, tương tự như loại mực toner được sử dụng trong các máy photocopy. Mực toner được giữ trên bề mặt của trực lăn bởi lực hấp dẫn của chính nó đối với một nam châm nằm bên trong trực lăn. Một thanh gạt kiểm soát (*control blade*) sẽ ngăn cản không cho mực toner bám vào bề mặt trực lăn. Mực toner này sẽ nhận một điện tích âm (giữa -200V và -500V) vì bề mặt này được nối tới một bộ nguồn DC được gọi là bộ thế dịch DC (*DC bias*).



Hình 5.1.3 bước triển khai, mực toner tích điện sẽ được đặt lên bề mặt của trống.

Mực toner mang điện tích âm nhiều hơn các vùng mang điện tích -100V trên bề mặt trống, nhưng ít hơn các vùng mang điện tích -600V trên bề mặt trống. Do đó, mực toner bị hút vào các vùng -100V trên bề mặt trống. Đồng thời, mực toner bị đẩy ra khỏi các vùng điện tích -600V của bề mặt trống, vì chúng mang điện tích âm tương đối đối với điện tích của mực toner. Kết quả là mực toner sẽ bám dính lên trống tại những nơi mà tia laser đã chiếu vào và bị đẩy ra khỏi những nơi mà tia laser chưa chiếu vào.

Hầu hết các máy in đều cung cấp một cách để bạn điều chỉnh mật độ in (*print density*). Với các máy in laser, khi bạn điều chỉnh mật độ in, bạn đang điều

chính điện tích bộ thể hiệu dịch DC (*DC bias*) trên trục lăn triển khai; điện tích này kiểm soát mực toner được hút vào trục lăn và do đó, khi điện tích này thay đổi, mật độ in cũng thay đổi, mật độ in cũng thay đổi theo.

Bước 5: Chuyển giao.

Trong bước chuyển giao, thiết bị điện hoa chuyển giao sẽ sinh ra một điện thế dương trên tờ giấy in khiến mực toner bị hút từ trống quay sang tờ giấy in khi nó qua giữa thiết này và trống quay. Bộ khử tĩnh điện(*static charge eliminator*) sẽ làm yếu điện tích dương trên tờ giấy in và điện tích âm trên trống quay để tờ giấy này không bám chặt vào trống quay do sự chênh lệch điện tích. Tính chất rít của tờ giấy in và bán kính nhỏ của trống quay khiến tờ giấy này tách rời khỏi trống in và đi tới trục nung chảy (*fusing roller*). Nếu sử dụng loại giấy mỏng trong một máy in laser, tờ giấy in có thể quấn tròn quanh trống quay và đây là lý do giải thích tại sao các tài liệu hướng dẫn sử dụng máy in laser đều chỉ dẫn bạn sử dụng chỉ những loại giấy được thiết kế dành cho máy in laser.

Bước 6: Nung chảy.

Bước nung chảy sẽ làm cho mực toner liên kết với giấy in. Cho tới thời điểm này, mực toner chỉ đơn thuần nằm trên giấy in. Các trục lăn nung chảy (*fusing roller*) sẽ áp dụng vừa áp suất lẫn nhiệt độ trên tờ giấy này. Mực toner sẽ lan chảy và các trục lăn sẽ ép mực toner vào tờ giấy in. Nhiệt độ của các trục lăn này được máy in giám sát. Nếu nhiệt độ này vượt quá giá trị tối đa cho phép (410F đối với một số máy in), máy in sẽ tự động tắt.

§ 5.2. Các thiết bị lưu trữ dữ liệu

Ổ đĩa cứng, hay còn gọi là ổ cứng *Hard Disk Drive*, viết tắt: 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 loại bộ nhớ "không thay đổi" (*non-volatile*), có nghĩa là chúng không bị mất dữ liệu khi ngừng cung cấp nguồn điện cho chúng..

5.2.1. Cấu tạo

Ổ đĩa cứng gồm các thành phần, bộ phận có thể liệt kê và giải thích như sau:

Bộ phận đĩa: Bao gồm toàn bộ các đĩa, trục quay và động cơ.

- **Đĩa từ.**
- **Trục quay:** truyền chuyển động của đĩa từ.
- **Động cơ:** Được gắn đồng trục với trục quay và các đĩa.

Bộ phận đầu đọc

- **Đầu đọc (*head*):** Đầu đọc/ghi dữ liệu
- **Cần di chuyển đầu đọc (*head arm* hoặc *actuator arm*).**

Bộ phận mạch điện



- Mạch điều khiển: có nhiệm vụ điều khiển **động cơ** đồng trục, điều khiển sự di chuyển của cần di chuyển đầu đọc để đảm bảo đến đúng vị trí trên bề mặt đĩa.
- Mạch xử lý dữ liệu: dùng để xử lý những **dữ liệu** đọc/ghi của ổ đĩa cứng.
- Bộ nhớ đệm (*cache* hoặc *buffer*): là nơi tạm lưu **dữ liệu** trong quá trình đọc/ghi dữ liệu. Dữ liệu trên bộ nhớ đệm sẽ mất đi khi ổ đĩa cứng ngừng được cấp **điện**.
- Đầu cắm nguồn cung cấp điện cho ổ đĩa cứng.
- Đầu kết nối giao tiếp với **máy tính**.
- Các cầu đầu thiết đặt (*jumper*) thiết đặt chế độ làm việc của ổ đĩa cứng: Lựa chọn chế độ làm việc của ổ đĩa cứng (SATA 150 hoặc SATA 300) hay thứ tự trên các kênh trên giao tiếp IDE (master hay slave hoặc tự lựa chọn), lựa chọn các thông số làm việc khác...

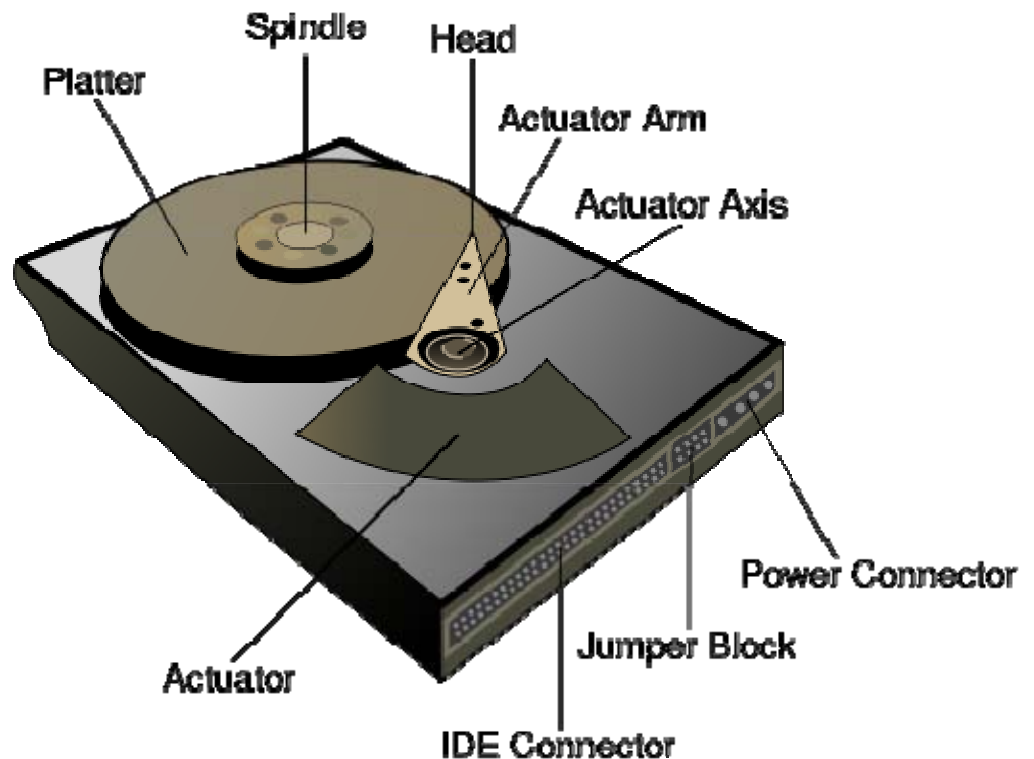
* Vỏ đĩa cứng:

Vỏ ổ đĩa cứng gồm các phần: Phần để chứa các **linh kiện** gắn trên nó, phần nắp đậy lại để bảo vệ các linh kiện bên trong.

Vỏ ổ đĩa cứng có chức năng chính nhằm định vị các linh kiện và đảm bảo độ kín khít để không cho phép bụi được lọt vào bên trong của ổ đĩa cứng.

Ngoài ra, vỏ đĩa cứng còn có tác dụng chịu đựng sự va chạm (ở mức độ thấp) để bảo vệ ổ đĩa cứng.

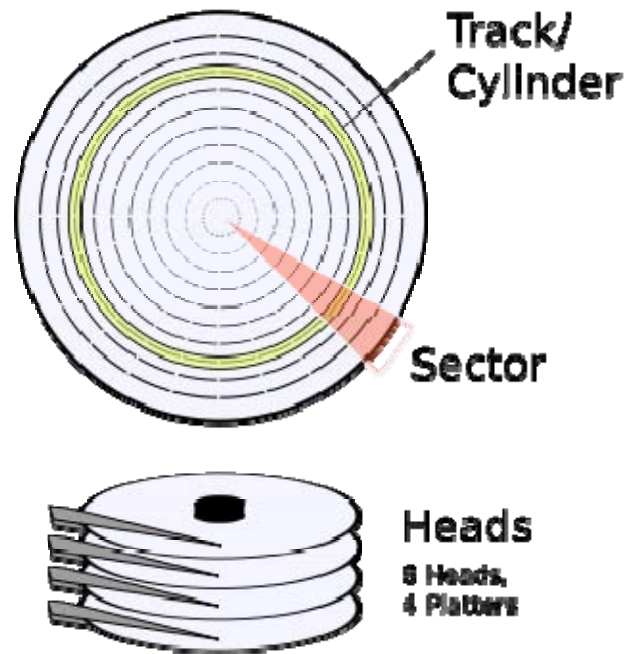
Do đầu từ chuyển động rất sát mặt đĩa nên nếu có bụi lọt vào trong ổ đĩa cứng cũng có thể làm xước bề mặt, mất lớp từ và hư hỏng từng phần (xuất hiện các khối hư hỏng (bad block))... Thành phần bên trong của ổ đĩa cứng là không khí có độ sạch cao, để đảm bảo áp suất cân bằng giữa môi trường bên trong và bên ngoài, trên vỏ bảo vệ có các hệ lỗ thoáng đảm bảo cản bụi và cân bằng áp suất.



*** Đĩa từ (platter):**

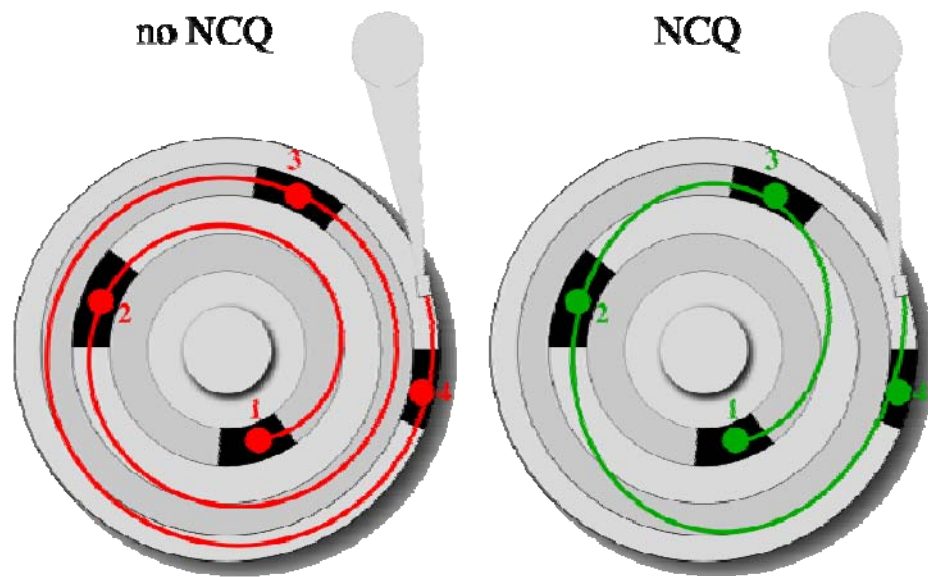
Đĩa thường cấu tạo bằng nhôm hoặc thủy tinh, trên bề mặt được phủ một lớp vật liệu từ tính là nơi chứa dữ liệu. Tùy theo hãng sản xuất mà các đĩa này được sử dụng một hoặc cả hai mặt trên và dưới. Số lượng đĩa có thể nhiều hơn một, phụ thuộc vào dung lượng và công nghệ của mỗi hãng sản xuất khác nhau. Mỗi đĩa từ có thể sử dụng hai mặt, đĩa cứng có thể có nhiều đĩa từ, chúng gắn song song, quay đồng trục, cùng tốc độ với nhau khi hoạt động.

* Track



Trên một mặt làm việc của đĩa từ chia ra nhiều vòng tròn đồng tâm thành các track.

Track có thể được hiểu đơn giản giống các rãnh ghi dữ liệu giống như các **đĩa nhựa** (ghi âm nhạc trước đây) nhưng sự cách biệt của các rãnh ghi này không có các gờ phân biệt và chúng là các vòng tròn đồng tâm chứ không nối tiếp nhau thành dạng xoắn tròn ốc như **đĩa nhựa**. Track trên ổ đĩa cứng không cố định từ khi sản xuất, chúng có thể thay đổi vị trí khi định dạng cấp thấp ổ đĩa (*low format*).



Khi một ổ đĩa cứng đã hoạt động quá nhiều năm liên tục, khi kết quả kiểm tra bằng các phần mềm cho thấy xuất hiện nhiều khối hư hỏng (bad block) thì có nghĩa là phần cơ của nó đã rơ rã và làm việc không chính xác như khi mới sản xuất, lúc này thích hợp nhất là **format cấp thấp** cho nó để tương thích hơn với chế độ làm việc của phần cơ

* **Sector**

Khu vực	Số sector/track	Số byte/track	Tốc độ truyền dữ liệu (MBps)
1	456	233.472	28,02
2	432	221.184	26,54
3	416	212.992	25,56
4	384	196.608	23,59
5	360	184.320	22,12

Trên *track* chia thành những phần nhỏ bằng các đoạn hướng tâm thành các sector. Các sector là phần nhỏ cuối cùng được chia ra để chứa dữ liệu. Theo chuẩn thông thường thì một sector chứa dung lượng 512 **byte**.

Số sector trên các track là khác nhau từ phần rìa đĩa vào đến vùng tâm đĩa, các ổ đĩa cứng đều chia ra hơn 10 vùng mà trong mỗi vùng có số sector/track bằng nhau.

* **Cylinder**

Tập hợp các *track* cùng bán kính (cùng số hiệu trên) ở các mặt đĩa khác nhau thành các cylinder. Nói một cách chính xác hơn thì: khi đầu đọc/ghi đầu tiên làm việc tại một track nào thì tập hợp toàn bộ các track trên các bề mặt đĩa còn lại mà các đầu đọc còn lại đang làm việc tại đó gọi là cylinder (*cách giải thích này chính xác hơn bởi có thể xảy ra thường hợp các đầu đọc khác nhau có khoảng cách đến tâm quay của đĩa khác nhau do quá trình chế tạo*). Trên một ổ đĩa cứng có nhiều cylinder bởi có nhiều track trên mỗi mặt đĩa từ.

* **Trục quay**

Trục quay là trục để gắn các đĩa từ lên nó, chúng được nối trực tiếp với **động cơ** quay đĩa cứng. Trục quay có nhiệm vụ truyền **chuyển động** quay từ **động cơ** đến các đĩa từ. Trục quay thường chế tạo bằng các **vật liệu** nhẹ (như **hợp kim nhôm**) và được chế tạo tuyệt đối chính xác để đảm bảo **trọng tâm** của chúng không được sai lệch - bởi chỉ một sự sai lệch nhỏ có thể gây lên sự rung lắc của toàn bộ đĩa cứng khi làm việc ở **tốc độ** cao, dẫn đến quá trình đọc/ghi không chính xác.

* **Đầu đọc/ghi**

Đầu đọc đơn giản được cấu tạo gồm lõi **ferit** (trước đây là lõi **sắt**) và cuộn dây (giống như **nam châm** điện). Gần đây các **công nghệ** mới hơn giúp cho ổ đĩa cứng hoạt động với mật độ xít chặt hơn như: chuyển các hạt từ sắp xếp theo phương **vuông góc** với bề mặt đĩa nên các đầu đọc được thiết kế nhỏ gọn và phát triển theo các ứng dụng công nghệ mới. Đầu đọc trong đĩa cứng có công dụng đọc **dữ liệu** dưới dạng từ hoá trên bề mặt đĩa từ hoặc từ hoá lên các mặt đĩa khi ghi dữ liệu. Số đầu đọc ghi luôn bằng số mặt hoạt động được của các đĩa cứng, có nghĩa chúng nhỏ hơn hoặc bằng hai lần số đĩa (nhỏ hơn trong trường hợp ví dụ hai đĩa nhưng chỉ sử dụng 3 mặt).

* **Cần di chuyển đầu đọc/ghi**

Cần di chuyển đầu đọc/ghi là các thiết bị mà đầu đọc/ghi gắn vào nó. Cần có nhiệm vụ di chuyển theo phương song song với các đĩa từ ở một khoảng cách nhất định, dịch chuyển và định vị chính xác đầu đọc tại các vị trí từ mép đĩa đến vùng phía trong của đĩa (phía trục quay). Các cần di chuyển đầu đọc được di chuyển đồng thời với nhau do chúng được gắn chung trên một trục quay (đồng trục), có nghĩa rằng khi việc đọc/ghi **dữ liệu** trên bề mặt (trên và dưới nếu là loại hai mặt) ở một vị trí nào thì chúng cũng hoạt động cùng vị trí tương ứng ở các bề mặt đĩa còn lại. Sự di chuyển cần có thể thực hiện theo hai phương thức:

- Sử dụng **động cơ bước** để truyền **chuyển động**.
- Sử dụng **cuộn cảm** để di chuyển cần bằng lực từ.

* **Hoạt động Giao tiếp với máy tính**

Cơ chế đọc và ghi dữ liệu ở ổ đĩa cứng không đơn thuần thực hiện từ theo tuần tự mà chúng có thể truy cập và ghi dữ liệu ngẫu nhiên tại bất kỳ điểm nào trên bề mặt đĩa từ, đó là đặc điểm khác biệt nổi bật của ổ đĩa cứng so với các hình thức lưu trữ truy cập tuần tự (như băng từ).

Thông qua giao tiếp với máy tính, khi giải quyết một tác vụ, CPU sẽ đòi hỏi dữ liệu (nó sẽ hỏi tuần tự các bộ nhớ khác trước khi đến đĩa cứng mà thứ tự thường là cache L1-> cache L2 ->RAM) và đĩa cứng cần truy cập đến các dữ liệu chứa trên nó. Không đơn thuần như vậy CPU có thể đòi hỏi nhiều hơn một tập tin dữ liệu tại một thời điểm, khi đó sẽ xảy ra các trường hợp:

1. Ổ đĩa cứng chỉ đáp ứng một yêu cầu truy cập **dữ liệu** trong một thời điểm, các yêu cầu được đáp ứng tuần tự.
2. Ổ đĩa cứng đồng thời đáp ứng các yêu cầu cung cấp **dữ liệu** theo phương thức riêng của nó.

Trước đây đa số các ổ đĩa cứng đều thực hiện theo phương thức 1, có nghĩa là chúng chỉ truy cập từng tập tin cho **CPU**. Ngày nay các ổ đĩa cứng đã được tích hợp các bộ nhớ đệm (*cache*) cùng các công nghệ riêng của chúng (TCQ, NCQ) giúp tối ưu cho hành động truy cập dữ liệu trên bề mặt đĩa nên ổ đĩa cứng sẽ thực hiện theo phương thức thứ 2 nhằm tăng tốc độ chung cho toàn hệ thống.

* **Đọc và ghi dữ liệu trên bề mặt đĩa**

Sự hoạt động của đĩa cứng cần thực hiện đồng thời hai chuyển động: Chuyển động quay của các đĩa và chuyển động của các đầu đọc. Sự quay của các đĩa từ được thực hiện nhờ các động cơ gắn cùng trục (với tốc độ rất

lớn: từ 3600 **rpm** cho đến 15.000 rpm) chúng thường được quay ổn định tại một tốc độ nhất định theo mỗi loại ổ đĩa cứng. Khi đĩa cứng quay đều, cần di chuyển đầu đọc sẽ di chuyển đến các vị trí trên các bề mặt chứa phủ vật liệu từ theo phương **bán kính** của đĩa. Chuyển động này kết hợp với chuyển động quay của đĩa có thể làm đầu đọc/ghi tới bất kỳ vị trí nào trên bề mặt đĩa. Tại các vị trí cần đọc ghi, đầu đọc/ghi có các bộ cảm biến với điện trường để đọc dữ liệu (và tương ứng: phát ra một điện trường để xoay hướng các hạt từ khi ghi dữ liệu). Dữ liệu được ghi/đọc đồng thời trên mọi đĩa. Việc thực hiện phân bố dữ liệu trên các đĩa được thực hiện nhờ các mạch điều khiển trên bo mạch của ổ đĩa cứng.

5.2.2 Các công nghệ chế tạo ổ đĩa cứng

5.2.2.1 S.M.A.R.T

S.M.A.R.T (Self-Monitoring, Analysis, and Reporting Technology) là công nghệ tự động giám sát, chuẩn đoán và báo cáo các hư hỏng có thể xuất hiện của ổ đĩa cứng để thông qua BIOS, các phần mềm thông báo cho người sử dụng biết trước sự hư hỏng để có các hành động chuẩn bị đối phó (như sao chép dữ liệu dự phòng hoặc có các kế hoạch thay thế ổ đĩa cứng mới). Trong thời gian gần đây S.M.A.R.T được coi là một tiêu chuẩn quan trọng trong ổ đĩa cứng. S.M.A.R.T chỉ thực sự giám sát những sự thay đổi, ảnh hưởng của phần cứng đến quá trình lỗi xảy ra của ổ đĩa cứng (mà theo hãng **Seagate** thì sự hư hỏng trong đĩa cứng chiếm tới 60% xuất phát từ các vấn đề liên quan đến cơ khí): Chúng có thể bao gồm những sự hư hỏng theo thời gian của phần cứng: đầu đọc/ghi (mất kết nối, khoảng cách làm việc với bề mặt đĩa thay đổi), động cơ (xuống cấp, rơ rão), bo mạch của ổ đĩa (hư hỏng linh kiện hoặc làm việc sai).

*S.M.A.R.T không nên được hiểu là từ "smart" bởi chúng không làm cải thiện đến tốc độ làm việc và truyền dữ liệu của ổ đĩa cứng. Người sử dụng có thể bật (enable) hoặc tắt (disable) chức năng này trong **BIOS** (tuy nhiên không phải **BIOS** của hãng nào cũng hỗ trợ việc can thiệp này).*

Ổ cứng lai (*hybrid hard disk drive*)

Ổ cứng lai (*hybrid hard disk drive*) là các ổ đĩa cứng thông thường được gắn thêm các phân bộ nhớ flash trên bo mạch của ổ đĩa cứng. Cụm bộ nhớ này hoạt động khác với cơ chế làm việc của bộ nhớ đệm (cache) của ổ đĩa cứng: Dữ liệu chứa trên chúng không bị mất đi khi mất điện.

Trong quá trình làm việc của ổ cứng lai, vai trò của phần **bộ nhớ flash** như sau:

- Lưu trữ trung gian dữ liệu trước khi ghi vào đĩa cứng, chỉ khi máy tính đã đưa các dữ liệu đến một mức nhất định (tùy từng loại ổ cứng lai) thì ổ đĩa cứng mới tiến hành ghi dữ liệu vào các đĩa từ, điều này giúp sự vận hành của ổ đĩa cứng tối hiệu quả và tiết kiệm điện năng hơn nhờ việc không phải thường xuyên hoạt động.
- Giúp tăng tốc độ giao tiếp với máy tính: Việc đọc dữ liệu từ bộ nhớ flash nhanh hơn so với việc đọc dữ liệu tại các đĩa từ.
- Giúp hệ điều hành khởi động nhanh hơn nhờ việc lưu các tập tin khởi động của hệ thống lên vùng bộ nhớ flash.
- Kết hợp với bộ nhớ đệm của ổ đĩa cứng tạo thành một hệ thống hoạt động hiệu quả.

Những ổ cứng lai được sản xuất hiện nay thường sử dụng bộ nhớ flash với dung lượng khiêm tốn ở 256 MB bởi chịu áp lực của vấn đề giá thành sản xuất. Do sử dụng dung lượng nhỏ như vậy nên chưa cải thiện nhiều đến việc giảm thời gian khởi động hệ điều hành, dẫn đến nhiều người sử dụng chưa cảm thấy hài lòng với chúng. Tuy nhiên người sử dụng thường khó nhận ra sự hiệu quả của chúng khi thực hiện các tác vụ thông thường hoặc việc tiết kiệm năng lượng của chúng.

Hiện ổ cứng lai có giá thành khá đắt (khoảng vài trăm USD cho dung lượng vài chục GB) nên chúng mới được sử dụng trong một số loại máy tính xách tay cao cấp.

5.2.3 Thông số và đặc tính của HDD

* Dung lượng

Dung lượng ổ đĩa cứng được tính bằng: $(\text{số byte/sector}) \times (\text{số sector/track}) \times (\text{số cylinder}) \times (\text{số đầu đọc/ghi})$.

Dung lượng của ổ đĩa cứng tính theo các đơn vị dung lượng cơ bản thông thường: byte, kB, MB, GB, TB.

Đa số các hãng sản xuất đều tính dung lượng theo cách có lợi (theo cách tính $1 \text{ GB} = 1000 \text{ MB}$ mà thực ra phải là $1 \text{ GB} = 1024 \text{ MB}$) nên dung lượng mà hệ điều hành (hoặc các phần mềm kiểm tra) nhận ra của ổ đĩa cứng thường thấp hơn so với dung lượng ghi trên nhãn đĩa (ví dụ ổ đĩa cứng 40 GB thường chỉ đạt khoảng 37-38 GB).

* Tốc độ quay của ổ đĩa cứng

Tốc độ quay của đĩa cứng thường được ký hiệu bằng rpm (viết tắt của từ tiếng Anh: *revolutions per minute*) số vòng quay trong một phút.

Tốc độ quay càng cao thì ổ càng làm việc nhanh do chúng thực hiện đọc/ghi nhanh hơn, thời gian tìm kiếm thấp.

Các tốc độ quay thông dụng thường là:

- 5.400 rpm: Thông dụng với các ổ đĩa cứng 3,5" sản xuất cách đây 2-3 năm; với các ổ đĩa cứng 2,5" cho các **máy tính xách tay** hiện nay đã chuyển sang tốc độ 5400 rpm để đáp ứng nhu cầu đọc/ghi dữ liệu nhanh hơn.
- 7.200 rpm: Thông dụng với các ổ đĩa cứng sản xuất trong thời gian hiện tại (2007)
- 10.000 rpm, 15.000 rpm: Thường sử dụng cho các ổ đĩa cứng trong các máy tính cá nhân cao cấp, máy trạm và các máy chủ có sử dụng giao tiếp SCSI

5.2.4 Các thông số về thời gian trong ổ đĩa cứng

* Thời gian tìm kiếm trung bình

Thời gian tìm kiếm trung bình (*Average Seek Time*) là khoảng thời gian trung bình (theo mili giây: ms) mà đầu đọc có thể di chuyển từ một cylinder này đến một cylinder khác ngẫu nhiên (ở vị trí xa chúng). Thời gian tìm kiếm trung bình được cung cấp bởi nhà sản xuất khi họ tiến hành hàng loạt các việc thử việc đọc/ghi ở các vị trí khác nhau rồi chia cho số lần thực hiện để có kết quả thông số cuối cùng. Thông số này càng thấp càng tốt.

Thời gian tìm kiếm trung bình không kiểm tra bằng các phần mềm bởi các phần mềm không can thiệp được sâu đến các hoạt động của ổ đĩa cứng.

* Thời gian truy cập ngẫu nhiên

Thời gian truy cập ngẫu nhiên (*Random Access Time*): Là khoảng thời gian trung bình để đĩa cứng tìm kiếm một **dữ liệu** ngẫu nhiên. Tính bằng mili giây (ms).

Đây là tham số quan trọng do chúng ảnh hưởng đến hiệu năng làm việc của hệ thống, do đó người sử dụng nên quan tâm đến chúng khi lựa chọn giữa các ổ đĩa cứng. Thông số này càng thấp càng tốt.

Tham số: Các ổ đĩa cứng sản xuất gần đây (2007) có thời gian truy cập ngẫu nhiên trong khoảng: 5 đến 15 **ms**.

* Thời gian làm việc tin cậy

Thời gian làm việc tin cậy MTBF: (*Mean Time Between Failures*) được tính theo giờ (hay có thể hiểu một cách đơn thuần là tuổi thọ của ổ đĩa cứng). Đây là khoảng thời gian mà nhà sản xuất dự tính ổ đĩa cứng hoạt động ổn định mà sau thời gian này ổ đĩa cứng có thể sẽ xuất hiện lỗi (và không đảm bảo tin cậy).

Một số nhà sản xuất công bố ổ đĩa cứng của họ hoạt động với tốc độ 10.000 rpm với tham số: MTBF lên tới 1 triệu giờ, hoặc với ổ đĩa cứng hoạt động ở tốc độ 15.000 rpm có giá trị MTBF đến 1,4 triệu giờ thì những thông số này chỉ là kết quả của các tính toán trên **lý thuyết**. Hãy hình dung số **năm** mà nó hoạt động tin cậy (khi chia thông số MTBF cho $(24 \text{ giờ/ngày} \times 365 \text{ ngày/năm})$ sẽ thấy rằng nó có thể dài hơn **lịch sử** của bất kỳ hãng sản xuất ổ đĩa cứng nào, do đó người sử dụng có thể không cần quan tâm đến thông số này.

* Bộ nhớ đệm

Bộ nhớ đệm (*cache* hoặc *buffer*) trong ổ đĩa cứng cũng giống như **RAM** của máy tính, chúng có nhiệm vụ lưu tạm dữ liệu trong quá trình làm việc của ổ đĩa cứng.

Độ lớn của bộ nhớ đệm có ảnh hưởng đáng kể tới hiệu suất hoạt động của ổ đĩa cứng bởi việc đọc/ghi không xảy ra tức thời (do phụ thuộc vào sự di chuyển của đầu đọc/ghi, dữ liệu được truyền tới hoặc đi) sẽ được đặt tạm trong bộ nhớ đệm.

Đơn vị thường tính bằng kB hoặc MB.

Trong thời điểm năm 2007, dung lượng bộ nhớ đệm thường là 2 hoặc 8 MB cho các loại ổ đĩa cứng dung lượng đến khoảng 160 GB, với các ổ đĩa cứng dung lượng lớn hơn chúng thường sử dụng bộ nhớ đệm đến 16 MB hoặc cao hơn. Bộ nhớ đệm càng lớn thì càng tốt, nhưng hiệu năng chung của ổ đĩa cứng sẽ chững lại ở một giá trị bộ nhớ đệm nhất định mà từ đó bộ nhớ đệm có thể tăng lên nhưng hiệu năng không tăng đáng kể.

- Hệ điều hành cũng có thể lấy một phần bộ nhớ của hệ thống (**RAM**) để tạo ra một bộ nhớ đệm lưu trữ dữ liệu được lấy từ ổ đĩa cứng nhằm tối ưu việc xử lý đối với các dữ liệu thường xuyên phải truy cập, đây chỉ là một cách dùng riêng của hệ điều hành mà chúng không ảnh hưởng đến cách hoạt động hoặc hiệu suất vốn có của mỗi loại ổ đĩa cứng. Có rất nhiều phần mềm cho phép tinh chỉnh các thông số này của hệ điều hành tùy thuộc vào sự dư thừa **RAM** trên hệ thống.

5.2.4 Các chuẩn kết nối ổ cứng

Hiện nay ổ cứng gắn trong có 2 chuẩn kết nối thông dụng là IDE và SATA.

IDE (EIDE)

Parallel ATA (PATA) hay còn được gọi là EIDE (Enhanced integrated drive electronics) được biết đến như là 1 chuẩn kết nối ổ cứng thông dụng hơn 10 năm nay. Tốc độ truyền tải dữ liệu tối đa là 100 MB/giây. Các bo mạch chủ mới nhất hiện nay gần như đã bỏ hẳn chuẩn kết nối này, tuy nhiên, người dùng vẫn có thể mua loại card PCI EIDE Controller nếu muốn sử dụng tiếp ổ cứng EIDE.

SATA (Serial ATA)

Nhanh chóng trở thành chuẩn kết nối mới trong công nghệ ổ cứng nhờ vào những khả năng ưu việt hơn chuẩn IDE về tốc độ xử lý và truyền tải dữ liệu. SATA là kết quả của việc làm giảm tiếng ồn, tăng các luồng không khí trong hệ thống do những dây cáp SATA hẹp hơn 400% so với dây cáp IDE. Tốc độ truyền tải dữ liệu tối đa lên đến 150 - 300 MB/giây. Đây là lý do vì sao ta không nên sử dụng ổ cứng IDE chung với ổ cứng SATA trên cùng một hệ thống. Ổ cứng IDE sẽ “kéo” tốc độ ổ cứng SATA bằng với mình, khiến ổ cứng SATA không thể hoạt động đúng với “sức lực” của mình. Ngày nay, SATA là chuẩn kết nối ổ cứng thông dụng nhất và cũng như ở trên, ta có thể áp dụng card PCI SATA Controller nếu bo mạch chủ không hỗ trợ chuẩn kết nối này

các phiên bản Windows 2000/XP/2003/Vista hay phần mềm sẽ nhận dạng và tương thích tốt với cả ổ cứng IDE lẫn SATA. Tuy vậy, cách thức cài đặt chúng vào hệ thống thì khác nhau. Do đó, ta cần biết cách phân biệt giữa ổ cứng IDE và SATA để có thể tự cài đặt vào hệ thống của mình khi cần thiết. Cách thức đơn giản nhất để phân biệt là nhìn vào phía sau của ổ cứng, phần kết nối của nó. Ổ cứng PATA (IDE) với 40-pin kết nối song song, phần thiết lập jumper (10-pin với thiết lập master/slave/cable select) và phần nối kết nguồn điện 4-pin, độ rộng là 3,5-inch. Có thể gắn 2 thiết bị IDE trên cùng 1 dây cáp, có

nghĩa là 1 cáp IDE sẽ có 3 đầu kết nối, 1 sẽ gắn kết vào bo mạch chủ và 2 đầu còn lại sẽ vào 2 thiết bị IDE.

Ổ cứng SATA có cùng kiểu dáng và kích cỡ, về độ dày có thể sẽ mỏng hơn ổ cứng IDE do các hãng sản xuất ổ cứng ngày càng cải tiến về độ dày. Điểm khác biệt dễ phân biệt là kiểu kết nối điện mà chúng yêu cầu để giao tiếp với bo mạch chủ, đầu kết nối của ổ cứng SATA sẽ nhỏ hơn, nguồn đóng chốt, jumper 8-pin và không có phần thiết lập Master/Slave/Cable Select, kết nối Serial ATA riêng biệt.

Cáp SATA chỉ có thể gắn kết 1 ổ cứng SATA.

* Hai chuẩn kết nối cho ổ cứng gắn ngoài là USB, FireWire. Ưu điểm của 2 loại kết nối này so với IDE và SATA là chúng có thể cắm “nóng” rồi sử dụng ngay chứ không cần phải khởi động lại hệ thống.

CHƯƠNG 6: LẮP RÁP CÀI ĐẶT MÁY TÍNH

- § 6.1. Khảo sát, lắp ráp các thành phần phần cứng máy tính
- § 6.2. Khảo sát BIOS – đọc hiểu catalog
- § 6.3. Cài đặt WINDOWS XP, các driver thiết bị
- § 6.4. Phân chia, định dạng đĩa cứng bằng WINDOWS
- § 6.5. Cài đặt phần mềm ứng dụng

CHƯƠNG 7: SAO LƯU PHỤC HỒI DỮ LIỆU

- § 7.1. Phân chia đĩa.
- § 7.2. Backup, restore.
- § 7.3. Tạo file Image cho 1 partition.
- § 7.4. Phục hồi partition từ file Image đã tạo.

CHƯƠNG 8: BẢO MẬT VỚI REGISTRY, GROUP POLICY

- § 8.1. Giới thiệu
- § 8.2. Cấu hình máy tính với Registry, Group Policy
- § 8.3. Bảo vệ máy tính với Registry, Group Policy