

NGUYỄN TRUNG ĐỒNG

Kỹ thuật
VI XỬ LÝ

Giáo trình dành cho sinh viên ngành Công nghệ thông tin

NHÀ XUẤT BẢN KHOA HỌC KỸ THUẬT
HÀ NỘI 2010

LỜI NÓI ĐẦU

Công nghệ thông tin đang được ứng dụng rộng rãi trong nhiều lĩnh vực khoa học công nghệ và cuộc sống thường nhật. Bên cạnh khối lượng phần mềm hệ thống và ứng dụng đồ sộ, công nghệ phần cứng cũng phát triển vô cùng nhanh chóng. Có thể nói các hệ thống máy tính được cải thiện trong những khoảng thời gian rất ngắn, càng ngày càng nhanh hơn, mạnh hơn và hiện đại hơn.

Những kiến thức cơ bản về phần cứng của các hệ thống máy tính luôn luôn là đòi hỏi cấp thiết của những người chọn công nghệ thông tin làm định hướng cho nghề nghiệp và sự nghiệp khoa học trong tương lai.

Giáo trình Kỹ thuật Vi xử lý này được viết trên cơ sở chương trình khung của Bộ Giáo dục và đào tạo, và những bài giảng theo sát đề cương môn học đã được thực hiện lâu nay, và luôn luôn được sửa chữa, bổ sung để đáp ứng nhu cầu kiến thức của sinh viên học tập ngành công nghệ thông tin.

Giáo trình được chia thành 5 chương:

Chương I giới thiệu những kiến thức tổng quan được sử dụng trong kỹ thuật Vi xử lý: các hệ đếm, cách thức biểu diễn thông tin trong các hệ Vi xử lý và máy tính, cũng như nhìn nhận qua về lịch sử phát triển của các trung tâm Vi xử lý.

Chương II giới thiệu cấu trúc và hoạt động của các đơn vị xử lý trung tâm từ $\mu P8085$ đến các cấu trúc của Vi xử lý họ 80x86, các kiến trúc RISC và CISC. Do những ứng dụng thực tế rộng lớn trong đời sống, trong chương II có giới thiệu thêm cấu trúc và chức năng của chip Vi xử lý chuyên dụng $\mu C8051$.

Chương III cung cấp những kiến thức về tổ chức bộ nhớ cho một hệ Vi xử lý, kỹ thuật và các bước xây dựng vi nhớ ROM, RAM cho hệ Vi xử lý.

Chương IV đi sâu khảo sát một số mạch chức năng khả lập trình như mạch điều khiển vào/ra dữ liệu song song, mạch điều khiển vào/ra dữ liệu nối tiếp, mạch định thời và mạch điều khiển ngắt.

Chương V giới thiệu các cấu trúc và cách xây dựng, phối ghép một số thiết bị vào/ra cơ bản cho một hệ Vi xử lý như bàn phím Hexa, hệ thống chỉ thị 7 thanh, bàn phím máy tính và màn hình.

Cuốn giáo trình chắc chắn có nhiều thiếu sót, rất mong được sự góp ý của các độc giả. Mọi ý kiến đóng góp xin gửi theo địa chỉ:

Email dongnt@hn.vnn.vn

Điện thoại: 0983 410 866

MỤC LỤC

MỤC LỤC	5
CHƯƠNG I. TỔNG QUAN VỀ CÁC HỆ VI XỬ LÝ	9
I.1 Các hệ đếm	9
I.1.1 Hệ đếm thập phân (R = 10 - Decimal)	9
I.1.2 Hệ đếm nhị phân (R = 2 - Binary)	10
I.1.3 Hệ đếm bát phân (R = 8 - Octal)	10
I.1.4 Hệ đếm 16 (R = 16 - Hexa)	10
I.2 Chuyển đổi lẫn nhau giữa các hệ đếm	11
I.2.1 Hệ nhị phân và hệ thập phân	11
I.2.2 Hệ nhị phân và hệ Hexa	13
I.3 Biểu diễn thông tin trong các hệ Vi xử lý	13
I.3.1 Mã hoá các thông tin không số	14
I.3.2 Mã hoá các thông tin số	14
I.3.3 Biểu diễn dữ liệu số trong máy tính	14
I.3.4 Bản chất vật lý của thông tin trong các hệ Vi xử lý	19
I.4 Vài nét về thực hiện các phép tính trong hệ đếm nhị phân	20
I.4.1 Phép cộng và phép trừ	21
I.4.2 Phép nhân và phép chia	22
I.5 Cấu trúc của hệ Vi xử lý và máy vi tính	23
I.5.1 Vài nét về lịch sử phát triển các trung tâm Vi xử lý	23
I.5.2 Cấu trúc cơ bản của hệ Vi xử lý	27
I.5.3 Từ hệ Vi xử lý đến máy vi tính PC	29
CÂU HỎI ÔN TẬP CHƯƠNG I	31
CHƯƠNG II. CÁC ĐƠN VỊ VI XỬ LÝ TRUNG TÂM (CPU – CENTRAL PROCESSING UNIT)	32
II.1 Trung tâm Vi xử lý μP8085	32
II.1.1 Các nhóm tín hiệu trong μ P8085	34
II.1.2 Khái niệm và bản chất vật lý của các BUS trong hệ Vi xử lý	36
II.1.3 Các mạch 3 trạng thái, mạch chốt và mạch khuếch đại BUS 2 chiều	37
II.1.4 Biểu đồ Timing thực hiện lệnh của CPU μ P8085	40
II.1.5 Khái niệm chu kỳ BUS	42
II.1.6 Ngắt (Interrupt)	43

II.1.7	Truy nhập trực tiếp bộ nhớ (Direct Memory Access – DMA).....	45
II.1.8	Vì chương trình (MicroProgram) và tập lệnh của μ P8085.....	46
II.1.9	Vài nét về lập trình cho 8085	51
II.1.10	Hệ lệnh của μ P8085	51
II.2	Các trung tâm Vi xử lý họ 80x86.....	54
II.2.1	Mô tả chân của μ P8086 và các tín hiệu	54
II.2.2	Cấu trúc Trung tâm Vi xử lý họ 80x86.....	57
II.2.3	Hệ thống thanh ghi trong các μ P80x86	58
II.2.4	Các chế độ làm việc MIN/MAX	63
II.2.5	Phương thức quản lý bộ nhớ, các mode địa chỉ	63
II.2.6	Phương thức đánh địa chỉ thiết bị ngoại vi.....	69
II.2.7	Các mạch Multiplexer, mạch Decoder, mạch PLA.....	69
II.2.8	Vài nét về lập trình hợp ngữ	71
II.3	Cấu trúc và tính năng của một số chip Vi xử lý hiện đại.	72
II.3.1	Cấu trúc chip Vi xử lý Pentium.....	75
II.3.2	Kiến trúc RISC, CISC	79
II.3.3	Quản lý bộ nhớ	81
II.3.4	Bộ nhớ cache	82
II.4	Single-Chip MicroComputer μC8051	83
II.4.1	Tổng quan	83
II.4.2	Mô tả cấu trúc và chức năng.....	85
II.4.3	Lập trình cho μ C8051	96
II.4.4	Các khả năng ứng dụng của μ C8051	96
II.5.	Vài nét về quy trình chế tạo chip CPU.....	97
II.5.1	Thiết kế	97
II.5.2	Chế tạo	98
II.5.3	Các nhà máy sản xuất chip (Fab)	99
II.5.4	Quy trình sản xuất	99
II.5.5	Đóng gói	101
II.5.6	Tái kiểm tra.....	101
II.5.7	Một số thuật ngữ trong sản xuất chip.....	101
CÂU HỎI ÔN TẬP CHƯƠNG II.....		104
CHƯƠNG III. BỘ NHỚ TRONG CỦA HỆ VI XỬ LÝ.....		106
III.1	Bộ nhớ trong hệ Vi xử lý.....	106
III.1.1	Phần tử nhớ, vi mạch nhớ, từ nhớ và dung lượng bộ nhớ	106
III.1.2	Vài nét về bộ nhớ trong của hệ Vi xử lý và máy tính PC	107
III.1.3	Phân loại các chip nhớ ROM, RAM	109
III.2	Tổ chức bộ nhớ cho hệ Vi xử lý.....	112
III.2.1	Tổ chức bộ nhớ vật lý	112

III.2.2	Thiết kế vi nhớ cho hệ Vi xử lý.....	117
CÂU HỎI ÔN TẬP CHƯƠNG III.		125
CHƯƠNG IV. CÁC CHIP KHẢ LẬP TRÌNH (PROGRAMMABLE)		126
IV.1 Tổng quan		126
IV.2 Một số mạch chức năng tiêu biểu		126
IV.2.1	Mạch vào/ra dữ liệu song song PPI-8255 (Programmable Peripheral Interface). .	126
IV.2.2	Mạch điều khiển ngắt PIC-8259.....	131
IV.3.3	Mạch đếm định thời đa năng PIT-8253 (Programmable Interval Timer).....	140
IV.4.4	Mạch điều khiển vào/ra nối tiếp đồng bộ/dị bộ USART-8251 (Universal Synchronous/Asynchronous Receiver Transmitter).....	146
CÂU HỎI ÔN TẬP CHƯƠNG IV.		160
CHƯƠNG V. THIẾT BỊ VÀO RA CỦA HỆ VI XỬ LÝ		162
V.1 Bàn phím Hex Keyboard		162
V.2 Ghép nối bàn phím với hệ Vi xử lý		166
V.2.1.	Hệ thống bàn phím của máy vi tính	166
V.2.2.	Quá trình truyền dữ liệu từ bàn phím cho CPU.....	167
V.3 Mạch điều khiển và lập trình chỉ thị 7-segments		168
V.4 Màn hình (Monitor)		169
V.4.1	Màn hình ống tia âm cực CRT (Cathode Ray Tube)	169
V.4.2	Ghép nối màn hình với hệ Vi xử lý.....	171
V.4.3	Bộ điều khiển màn hình CRTIC	172
CÂU HỎI ÔN TẬP CHƯƠNG V.		175
PHỤ LỤC 176		
PHỤ LỤC A		176
Bảng tóm tắt hệ lệnh của Trung tâm Vi xử lý họ x86		176
PHỤ LỤC B		179
Bảng lũy thừa 2ⁿ		179
Bảng mã ASCII (American Standard Code for Information Interchange)		180
Kí tự ASCII in được	180
Kí tự điều khiển ASCII	181
Bảng mã EBCDIC	182

(Extended Binary Coded Decimal Interchange Code)	182
PHỤ LỤC D	184
Các nhóm lệnh của μ C8051	184
1. Tạo vòng lặp và lệnh nhảy	184
2. Lệnh gọi Call	185
a. Nhóm lệnh xử lý số học	186
b. Nhóm lệnh logic	187
c. Nhóm lệnh chuyển dữ liệu	189
d. Nhóm lệnh chuyển điều khiển	190
TÀI LIỆU THAM KHẢO	192

CHƯƠNG I. TỔNG QUAN VỀ CÁC HỆ VI XỬ LÝ

1.1 Các hệ đếm

Hệ đếm thông dụng nhất trong đời sống là hệ đếm cơ số 10 (thập phân – Decimal), sử dụng 10 ký tự số từ 0 đến 9. Ngoài ra, trong sản xuất, kinh doanh còn có khi sử dụng hệ đếm cơ số 12 (tá – dozen).

Trong các hệ thống máy tính, để xử lý, tính toán, ta sử dụng hệ đếm cơ số 2 (nhị phân – Binary), hệ cơ số 8 (bát phân – Octal), hệ cơ số 16 (Hexa). Tuy nhiên, việc nhập dữ liệu hay đưa kết quả xử lý, ta lại dùng hệ đếm cơ số 10.

Một số N trong một hệ đếm bất kỳ có $n+l$ chữ số, trong đó gồm n chữ số thuộc phần nguyên và l chữ số thuộc phần thập phân, được triển khai theo công thức tổng quát:

$$N = \sum_{k=-l}^n a_k R^k \quad \text{trong đó:}$$

R là cơ số của hệ đếm

a_k là trọng của chữ số ở vị trí thứ k ($0 \leq a_k < R$)

$$\{a_k\}_R = \{0, 1, 2, 3, \dots, R-1\}$$

l, n là số nguyên

$$N = a_n a_{n-1} \dots a_1 a_0, a_{-1} a_{-2} \dots a_{-l}$$

Theo công thức trên, các số được biểu diễn trong các hệ đếm khác nhau sẽ như sau:

1.1.1 Hệ đếm thập phân (R = 10 - Decimal)

$$\{a_k\}_D = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

$$123,45_D = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 4 \times 10^{-1} + 5 \times 10^{-2}$$

I.1.2 Hệ đếm nhị phân (R = 2 - Binary)

$$\{ a_k \}_B = \{0, 1\}$$

$$\begin{aligned} 11011.01_B &= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = \\ &= 16 + 8 + 0 + 2 + 1 + 0 + 0,25 = 27,25_D \end{aligned}$$

I.1.3 Hệ đếm bát phân (R = 8 - Octal)

$$\{ a_k \}_O = \{0, 1, 2, 3, 4, 5, 6, 7\}$$

$$\begin{aligned} 653,12_O &= 6 \times 8^2 + 5 \times 8^1 + 3 \times 8^0 + 1 \times 8^{-1} + 2 \times 8^{-2} = \\ &= 384 + 40 + 3 + 0,125 + 0,03125 = 427,1562_D \end{aligned}$$

Lưu ý: Các chữ số trong hệ này có thể biểu diễn nhờ 3 ký tự số (“0” và “1”) trong hệ đếm nhị phân theo bảng sau:

Octal	Binary	Octal	Binary	Octal	Binary	Octal	Binary
0 _O	000 _B	2 _O	010 _B	4 _O	100 _B	6 _O	110 _B
1 _O	001 _B	3 _O	011 _B	5 _O	101 _B	7 _O	111 _B

I.1.4 Hệ đếm 16 (R = 16 - Hexa)

$$\{ a_k \}_H = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$$

$$\begin{aligned} 3A7,C_H &= 3 \times 16^2 + 10 \times 16^1 + 7 \times 16^0 + 12 \times 16^{-1} = \\ &= 768 + 160 + 7 + 0,75 = 935,75_D \end{aligned}$$

Lưu ý: Một giá trị ký tự số Hexa có thể biểu diễn thông qua 4 ký tự số ở hệ nhị phân theo bảng sau:

Hexa	Binary	Hexa	Binary	Hexa	Binary	Hexa	Binary
0 _H	0000 _B	4 _H	0100 _B	8 _H	1000 _B	C _H	1100 _B
1 _H	0001 _B	5 _H	0101 _B	9 _H	1001 _B	D _H	1101 _B
2 _H	0010 _B	6 _H	0110 _B	A _H	1010 _B	E _H	1110 _B
3 _H	0011 _B	7 _H	0111 _B	B _H	1011 _B	F _H	1111 _B

Nhận xét:

1. Trong các hệ đếm vừa được nêu, hệ đếm cơ số 2 có rất nhiều ưu điểm khi xử lý trong máy tính. Thứ nhất, việc mô phỏng giá trị của một ký tự số là rất đơn giản: chỉ cần một phần tử có hai trạng thái khác biệt. Sử dụng bản chất vật lý của vật mang thông tin để biểu diễn hai trạng thái này rất dễ thực hiện. Trên đây dẫn điện là các trường hợp có dòng điện (tương ứng với trọng số là 1) hoặc không có dòng điện (tương ứng với trọng số là 0).
2. Việc chuyển đổi giữa hai giá trị 0 hoặc 1 có thể thực hiện thông qua một công tắc, trong thực tế là các phần tử logic điện tử thực hiện các chức năng của khoá điện tử: đóng (dòng điện đi qua được) hoặc mở (không kết nối - dòng điện không đi qua).

1.2 Chuyển đổi lẫn nhau giữa các hệ đếm

1.2.1 Hệ nhị phân và hệ thập phân

- a) Từ nhị phân sang thập phân: Sử dụng biểu thức triển khai tổng quát đã nêu, cộng tất cả các số hạng theo giá trị số thập phân, tổng số là dạng thập phân của số nhị phân đã cho.

$$\begin{aligned} \text{Ví dụ: } 11011.11_B &= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} \\ &= 16 + 8 + 0 + 2 + 1 + 0.5 + 0.25 = 27.75_D \end{aligned}$$

- b) Từ thập phân sang nhị phân:

Phần nguyên: Ta có đẳng thức sau (vế trái là số thập phân, vế phải là biểu diễn nhị phân của số đó):

$$\begin{aligned} S_D &= k_n 2^n + k_{n-1} 2^{n-1} + k_{n-2} 2^{n-2} + \dots + k_1 2^1 + k_0 2^0 + = \\ &= 2(k_n 2^{n-1} + k_{n-1} 2^{n-2} + k_{n-2} 2^{n-3} + \dots + k_1) + k_0 = \\ &= 2[2(k_n 2^{n-1} + k_{n-1} 2^{n-2} + k_{n-2} 2^{n-3} + \dots + k_2) + k_1] + k_0 \end{aligned}$$

Vì $k_i = \{0, 1\}$, đồng phân với số 0, 1 trong số thập phân, nên ta có thể viết:

$$\frac{S_D - k_0}{2} = k_n 2^{n-1} + k_{n-1} 2^{n-2} + k_{n-2} 2^{n-3} + \dots + k_1 = 2(k_n 2^{n-2} + k_{n-1} 2^{n-3} + \dots + k_2) + k_1$$

Thấy rằng: Ký tự đầu tiên của số nhị phân là k_0 , đúng với số dư khi chia S_D cho 2, ký tự tiếp theo, k_1 chính là số dư khi chia thương cho 2, v. v... nên ta có thể tìm tất cả các ký tự khác như sau:

Ví dụ: Đổi số 173_D ra số nhị phân

173	2	đư 1	k ₀	↑ ⋮
86	2	đư 0	k ₁	
43	2	đư 1	k ₂	
21	2	đư 1	k ₃	
10	2	đư 0	k ₄	
5	2	đư 1	k ₅	
2	2	đư 0	k ₆	
1	2	đư 1	k ₇	
0				

Ghép phần dư theo thứ tự từ dưới lên, ta được:

$$173_D = 10101101_B$$

Phân phân số: Đẳng thức quan hệ giữa số thập phân và số nhị phân (phần phân số) (vế trái là số thập phân, vế phải là số nhị phân) như sau:

$$S_D = k_1 2^{-1} + k_2 2^{-2} + k_3 2^{-3} + \dots + k_{m+1} 2^{-m+1} + k_m 2^{-m}$$

$$2S_D = k_1 + (k_2 2^{-1} + k_3 2^{-2} + \dots + k_{m+1} 2^{-m+2} + k_m 2^{-m+1})$$

Thấy rằng k_1 trở thành phần nguyên của vế phải, vậy:

$$2S_D - k_1 = (k_2 2^{-1} + k_3 2^{-2} + \dots + k_{m+1} 2^{-m+2} + k_m 2^{-m+1})$$

$$2(2S_D - k_1) = k_2 + (k_3 2^{-1} + \dots + k_{m+1} 2^{-m+3} + k_m 2^{-m+2})$$

k_2 là phần nguyên tiếp theo của vế phải có thể bằng “0” hoặc bằng “1”. Tiếp tục tương tự, thu được các ký tự số của các phần tử còn lại.

Ví dụ: Chuyển đổi số 0.8128 thành số nhị phân

Thực hiện phép nhân liên tiếp với 2, phần nguyên của tích bao giờ cũng là các giá trị hoặc bằng “0” hoặc bằng “1”, thu được kết quả sau:

0.8128	x 2	= 1.6256	= 1	+ 0.6256
0.6256	x 2	= 1.2512	= 1	+ 0.2512
0.2512	x 2	= 0.5024	= 0	+ 0.5024
0.5024	x 2	= 1.0048	= 1	+ 0.0048
0.0048	x 2	Quá nhỏ có thể bỏ qua		

Lưu ý: Quá trình biến đổi này kết thúc khi phần phân số của tích số bằng 0, tuy nhiên, nếu quá dài, tùy theo yêu cầu của độ chính xác dữ liệu khi tính toán và xử lý, có thể bỏ qua.

1.2.2 Hệ nhị phân và hệ Hexa

Chuyển đổi một dữ liệu nhị phân sang hệ Hexa rất đơn giản, nếu chú ý rằng ta có $2^4 = 16$, có nghĩa là một số Hexa tương ứng với một nhóm 4 số của số nhị phân (từ 0 đến F). Vì vậy, khi chuyển đổi, chỉ cần thay nhóm 4 chữ số của số nhị phân bằng một chữ số tương ứng của hệ Hexa như sau:

Tổ hợp nhị phân	Ký tự số Hexa	Tổ hợp nhị phân	Ký tự số Hexa	Tổ hợp nhị phân	Ký tự số Hexa	Tổ hợp nhị phân	Ký tự số Hexa
0 0 0 0	0	0 1 0 0	4	1 0 0 0	8	1 1 0 0	C
0 0 0 1	1	0 1 0 1	5	1 0 0 1	9	1 1 0 1	D
0 0 1 0	2	0 1 1 0	6	1 0 1 0	A	1 1 1 0	E
0 0 1 1	3	0 1 1 1	7	1 0 1 1	B	1 1 1 1	F

Ví dụ:

$$\begin{array}{cccc|cccc} \leftarrow & & & & & & & \rightarrow \\ 110 & | & 1101 & | & 0011 & | & 1001 & . & 0110 & | & 0101 & \text{B} & = & 6\text{D}39.65_{\text{H}} \\ 6 & & \text{D} & & 3 & & 9 & & 6 & & 5 & & & & \end{array}$$

Lưu ý: Phần nguyên được nhóm tính từ vị trí của chữ số có trọng nhỏ nhất, phần phân số được nhóm tính từ vị trí của chữ số có trọng lớn nhất.

Từ cách chuyển đổi trên, dễ dàng nhận ra phép chuyển đổi ngược từ một số hệ Hexa sang số hệ nhị phân bằng cách thay một chữ số trong hệ Hexa bằng một nhóm 4 chữ số trong hệ nhị phân.

Ví dụ: $\text{F5E7.8C}_{\text{H}} = 1111\ 0101\ 1110\ 0111.1000\ 1100_{\text{B}}$

$$\begin{array}{c|c|c|c|c|c} \text{F} & 5 & \text{E} & 7 & 8 & \text{C}_{\text{H}} \\ \hline 1111 & 0101 & 1110 & 0111 & 1000 & 1100 \end{array} = 1111\ 0101\ 1110\ 0111.1000\ 1100_{\text{B}}$$

1.3 Biểu diễn thông tin trong các hệ Vi xử lý

Các hệ Vi xử lý xử lý các thông tin số và chữ. Các thông tin được biểu diễn dưới dạng mã nhất định. Bản chất vật lý của việc biểu diễn thông tin là điện áp (“0” ứng với không có điện áp, “1” ứng với điện áp ở mức quy chuẩn trong mạch điện tử) và việc mã hoá các thông tin số và chữ được tuân theo chuẩn quốc tế. Một biến logic với chỉ hai giá trị duy nhất là “0” hoặc “1” được gọi là một bit. Hai trạng thái này của bit được sử dụng để mã hoá cho tất cả các ký tự (gồm số, chữ và các ký tự đặc biệt khác). Các bit được ghép lại thành các đơn vị mang thông tin đầy đủ cho các ký tự biểu diễn các số, các ký tự chữ và các ký tự đặc biệt khác.

Bit (Binary digiT) là đơn vị cơ bản của thông tin theo hệ đếm nhị phân. Các mạch điện tử trong máy tính phát hiện sự khác nhau giữa hai trạng thái (điện áp mức “1” và điện áp mức “0”) và biểu diễn hai trạng thái đó dưới dạng một trong hai số nhị phân “1” hoặc “0”.

Nhóm 8 bit ghép kề liền nhau, tạo thành đơn vị dữ liệu cơ sở của hệ Vi xử lý được gọi là 1 Byte. Do được lưu giữ tương đương với một ký tự (số, chữ hoặc ký tự đặc biệt) nên Byte cũng là đơn vị cơ sở để đo các khả năng lưu giữ, xử lý của hệ Vi xử lý. Các thuật ngữ như KiloByte, MegaByte, GigaByte hay TetraByte thường được dùng làm bội số trong việc đếm Byte, dĩ nhiên theo hệ đếm nhị phân, nghĩa là:

1KiloByte	=	1024 Bytes,
1MegaByte	=	1024 KiloBytes,
1GigaByte	=	1024 MegaBytes,
1TetraByte	=	1024 GigaByte

Các đơn vị này được viết tắt tương ứng là KB, MB, GB và TB.

1.3.1 Mã hoá các thông tin không số

Có hai loại mã phổ cập nhất được sử dụng là mã ASCII và EBCDIC.

- Mã ASCII (*American Standard Code for Information Interchange* - Chuẩn mã trao đổi thông tin Hoa Kỳ), thường được phát âm là *át-xơ-ki*, là bộ kí tự và bộ mã kí tự dựa trên bảng chữ cái La Tinh được dùng trong tiếng Anh hiện đại và các ngôn ngữ Tây Âu khác. Nó thường được dùng để hiển thị văn bản trong máy tính và các thiết bị thông tin khác. Nó cũng được dùng bởi các thiết bị điều khiển làm việc với văn bản hoá các ký tự.
- Mã EBCDIC (*Extended Binary Coded Decimal Interchange Code*) dùng cả 8 bits (1 Byte) để mã hoá thông tin.
- Còn một loại mã được dùng trong ngành bưu điện, trong các máy teletype là mã BAUDOT, chỉ sử dụng 5 bits để mã hoá thông tin.

1.3.2 Mã hoá các thông tin số

Các số được mã hoá theo các loại mã sau:

- Mã nhị phân sử dụng các số được biểu diễn theo hệ đếm nhị phân như đã nêu ở trên, một dạng khác của biểu diễn nhị phân của thông tin số là *mã Hexa*, tương ứng với cách ghép 4 digits nhị phân thành một digit Hexa.
- Mã nhị thập phân (BCD Code – Binary Coded Decimal Code) sử dụng cách nhóm 4 bits nhị phân để biểu diễn một giá trị thập phân từ 0 đến 9. Các giá trị vượt quá giới hạn này (> 9) không được sử dụng.

1.3.3 Biểu diễn dữ liệu số trong máy tính

- Biểu diễn dữ liệu là số nguyên có dấu: Giả sử dùng 2 bytes (16 bits) để biểu diễn một số nguyên có dấu, bit cao nhất (MSB – Most

Significant Bit) được dùng để đánh dấu. Số dương có bit dấu $S = "0"$, số âm có bit dấu $S = "1"$.

D ₁₅	D ₁₄	D ₁₃	D ₁₂	D ₁₁	D ₁₀	D ₉	D ₈	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
S	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

- Biểu diễn dữ liệu là số thực có dấu: Về nguyên tắc, dấu của số vẫn là giá trị của MSB như đã quy ước ở trên. Có hai dạng số có dấu phẩy được sử dụng trong máy tính: Số dấu phẩy tĩnh (Fixed point) và số dấu phẩy động (Floating point).
 - Dấu phẩy tĩnh sẽ phân chia chuỗi chữ số thành phần nguyên và phần phân số. Ví dụ ta có thể viết:

$$\pm 001\ 1101.0110\ 1101$$

Nhưng nói chung, trong các máy chuyên dụng, thường phải tìm một phương pháp thích hợp để có thể biểu diễn số có dấu phẩy cố định mà dấu phẩy được đặt ngay sau ô dấu, nghĩa là số dấu phẩy tĩnh có dạng:

$$\pm 0.k_n k_{n-1} k_{n-2} \dots k_1 k_0$$

- Dấu phẩy động được dùng rất phổ biến, giá trị thực được thể hiện nhờ việc thêm vào lũy thừa của cơ số đếm. Một dữ liệu số theo hệ đếm nhị phân được biểu diễn theo dạng chuẩn tắc như sau:

$$N = \pm F * 2^{\pm E}$$

trong đó: F là phần định trị được chuẩn hóa (Mantissa)
 E là phần đặc tính (Exponent - số mũ)

MSB	Các bit thể hiện phần đặc tính	Các bit thể hiện phần định trị
S	E	F

Lưu ý: Khi kết quả phép tính vượt quá các giới hạn trên, nếu số mũ (exponent) là dương, sẽ được coi là $-\infty$ hoặc $+\infty$. Trong trường hợp số mũ là âm và vượt qua số mũ cực đại cho phép, kết quả được coi là bằng 0.

Cũng cần lưu ý rằng, đối với các dữ liệu số có dấu, để thuận tiện cho xử lý và tính toán, trong máy thường được biểu diễn dưới các dạng mã thuận, mã ngược (complement) hoặc mã bù 2 (two-complement). Giả sử ta có số

$A=+0.10010$, các mã trên đều biểu diễn như nhau, nhưng với số $B = -0.10010$ thì sẽ được biểu diễn như sau:

Bình thường $A = -0.10010$

Mã ngược $A = 1.00110$ (bù 1, tức là đảo các chữ số trong số đó)

Mã bù 2 $A = 1.00111$ (tương ứng với bù 1 cộng thêm 1)

Chuẩn IEEE 754 được dùng rộng rãi trong khoa học máy tính hiện nay. Trong cách biểu diễn này, phần định trị có dạng $1.F$ với số 1 được ẩn (được coi là mặc định) và F là phân số lẻ.

S	E (8 bits)	F22	F21	F1	F0
D ₃₁	D ₃₀ --D ₂₃	D ₂₂	D ₂₁	D ₁	D ₀

Biểu diễn số có dấu phẩy động chính xác đơn với 32 bit

Chuẩn IEEE 754 định nghĩa hai dạng biểu diễn số dấu phẩy động:

- Số dấu phẩy động chính xác đơn với định dạng được định nghĩa: chiều dài số: 32 bit được chia thành các trường: dấu S (Sign bit - 1 bit), mũ E (Exponent - 8 bit), phần lẻ F (Fraction - 23 bit).

Số này tương ứng với số thực $(-1)^S * (1.F1 F2 F23) * 2^{(E - 127)}$

- Số dấu phẩy động chính xác kép (Double precision) với định dạng được định nghĩa: chiều dài số: 64 bit được chia thành các trường: dấu S (Sign bit - 1 bit), mũ E (Exponent - 11 bit), phần lẻ F (Fraction - 52 bit)

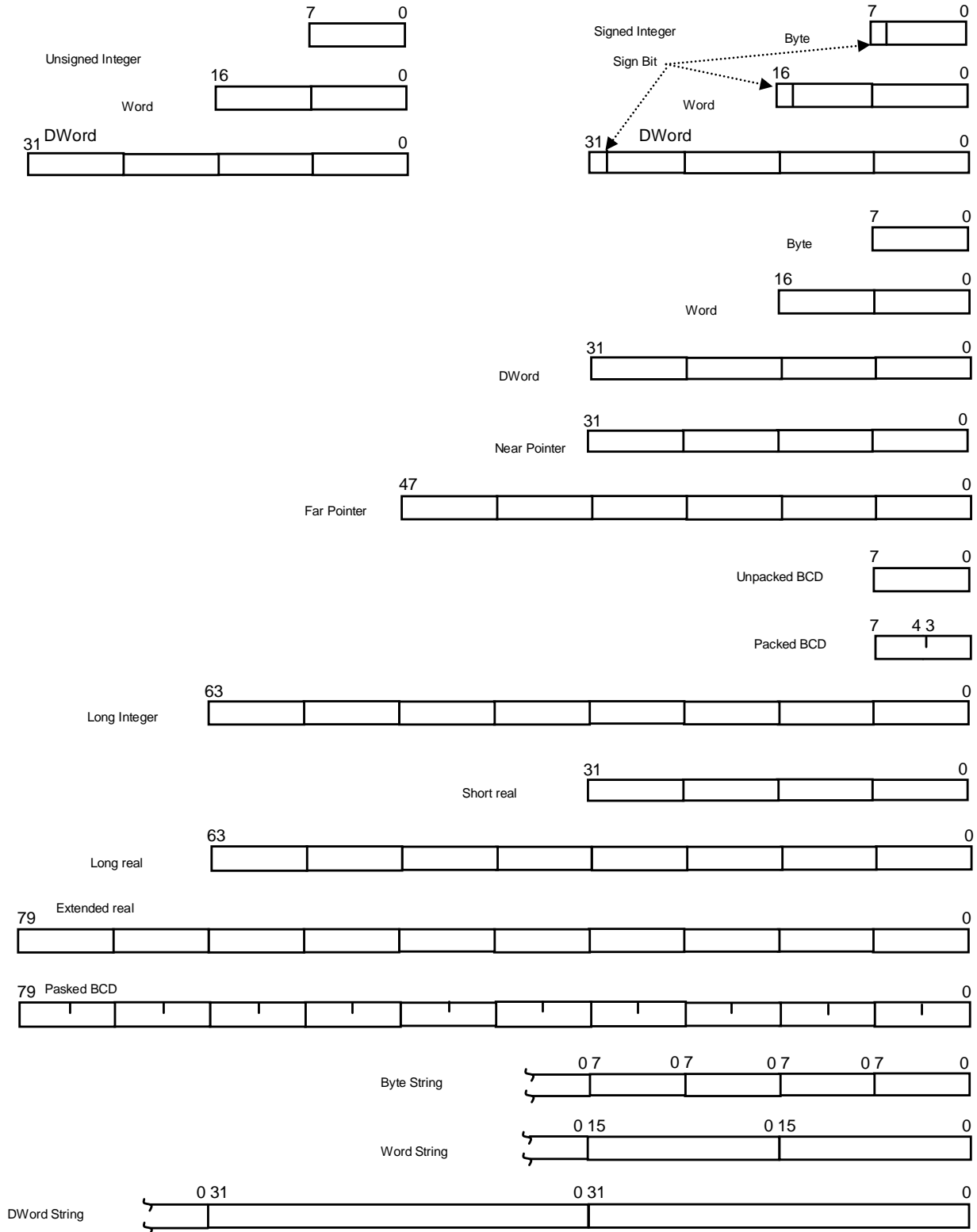
Số này tương ứng với số thực $(-1)^S * (1.F1 F2 F52) * 2^{(E - 1023)}$

S	E (8 bits)	F52	F51	F1	F0
D ₆₃	D ₆₂ —D ₅₃	D ₅₁	D ₅₀	D ₁	D ₀

Biểu diễn số có dấu phẩy động chính xác kép với 64 bit

Với chuẩn IEEE 754, các giá trị số dấu phẩy động chính xác đơn được biểu diễn nằm trong khoảng từ $1*2^{-126}$ đến $1*2^{127}*(2-2^{-23})$, tức là tương đương biểu diễn thập phân các giá trị từ $1,8*10^{-38}$ đến $3,4*10^{38}$, các giá trị số dấu phẩy động chính xác kép nằm trong khoảng từ $1*2^{-1022}$ đến $1*2^{1023}*(2-2^{-52})$, tương đương biểu diễn thập phân của các giá trị từ $2,23*10^{-308}$ đến $1,8*10^{308}$.

Để thuận lợi trong một số phép tính toán, IEEE định nghĩa một số dạng mở rộng của chuẩn IEEE 754.



Hình 1.0 Các dạng dữ liệu trong máy Vi tính PC

Chuẩn IEEE 754 cho phép biểu diễn các số chuẩn hoá (các bit của E không cùng lúc bằng 0 hoặc bằng 1), các số không chuẩn hoá (các bit của E không cùng lúc bằng 0 và phần số lẻ F22 F21 ... khác không), trị số 0 (các bit của E không cùng lúc bằng 0 và phần số lẻ bằng không), và các ký tự đặc biệt (các bit của E không cùng lúc bằng 1 và phần lẻ khác không).

Tham số	Chính xác đơn	Mở rộng chính xác đơn	Chính xác kép	Mở rộng chính xác kép
Chiều dài (bit)	32	$43 \geq$	64	$79 \geq$
Chiều dài trường mũ (E)	8	$11 \geq$	11	$15 \geq$
Số mũ	127	-	1023	-
Giá trị mũ tối đa	127	$1023 \geq$	1023	$16383 \geq$
Giá trị mũ tối thiểu	-126	$-1022 \leq$	-1022	$-16382 \leq$
Chiều dài trường lẻ F (bit)	23	$31 \geq$	52	$63 \geq$

Ví dụ các bước biến đổi số thập phân -12.62510 sang số dấu phẩy động chuẩn IEEE 754 chính xác đơn (32 bit):

- Bước 1: Đổi số -12.625_{10} sang nhị phân: $-12.625_{10} = -1100.101_2$.
- Bước 2: Chuẩn hoá: $-1100.101_2 = -1.100101_2 * 2^3$ (Số 1.100101_2 dạng 1.F)
- Bước 3: Điền các bit vào các trường theo chuẩn:

Số âm: bit dấu S có giá trị 1.

Phần mũ E với số mũ $K=127$, ta có: $E-127=3$

$E = 3_D + 127_D = 130_D$ (tức là 100000102_2). Các giá trị được phân tích ở trên sẽ được biểu diễn thành 32 bit gồm các phần S E F. Kết quả nhận được:

1 1000 0010 1001 0100 0000 0000 0000 0000
 S E F

Ví dụ: Chuyển đổi số -46,725 sang cách biểu diễn số thực chính xác đơn theo chuẩn IEEE 754.

Lời giải:

Chuyển đổi 46.725_D ra số nhị phân

46	0	↑	0.725	1	↓
23	1		1.550	1	
11	1		1.100	0	
5	1		0.200	0	
2	0		0.400	0	
1	1		0.800	1	
			1.600	1	
			1.200	0	
			1.400	0	
			1.800	1	
			1.600	1	
			1.200	0	

Chuyển đổi phân nguyên: $46_D = 101110_B$

Chuyển đổi phân thập phân:

$0.725_D \sim 0.11000110011001100110011_B$

Như vậy ta có biểu diễn nhị phân của số -46.725_D là

$-101110.1100011011001100110011_B$ và có thể viết dưới dạng số dấu phẩy động là $-1.0111011000110110011001100110011_B * 2^5$

Từ cách viết trên, thấy rằng, nếu biểu diễn theo chuẩn IEEE754 thì, bit 1 (cao nhất) sau dấu trừ là bị ẩn, nên 32 bit được biểu diễn gồm:

+) bit dấu $S = 1$

+) $F = 01110110001100110011001_B$ (chỉ sử dụng 23 bit cho biểu diễn các giá trị từ $F_{22}, F_{21}, F_{20}, \dots, F_3, F_2, F_1, F_0$).

+) số mũ là $E - 127 = 5$, hay $E = 132_D$, biểu diễn nhị phân của số mũ là 10000100_B

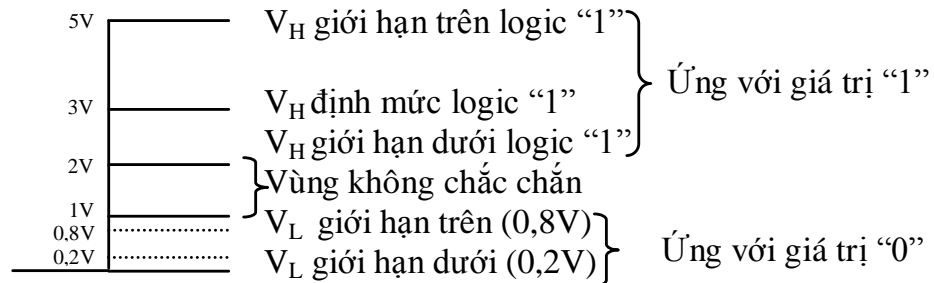
Vậy số thập phân $-46,725$ được biểu diễn trong máy tính thành 4 byte theo chuẩn IEEE754 như sau:

S	E (8.bits)	F (23.bits)
1	1 1 0 0 0 0 1 0 0	0 1 1 1 0 1 1 0 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1

1.3.4 Bản chất vật lý của thông tin trong các hệ Vi xử lý

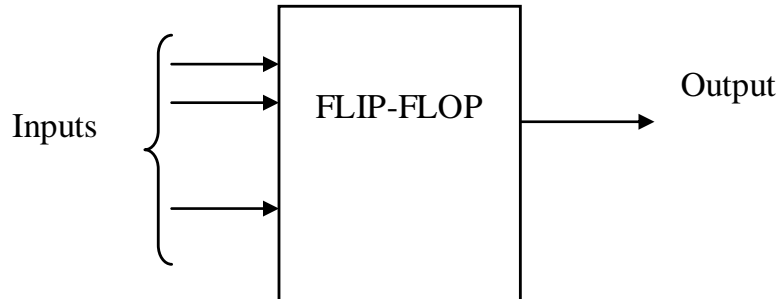
Trong các hệ Vi xử lý, thông tin về các giá trị “0” hay “1” được biểu diễn thông qua một mức điện áp so với mức chuẩn chung, thường là đất

(GND - Ground). Độ lớn của điện áp biểu diễn các giá trị này phụ thuộc vào công nghệ được sử dụng để tạo nên phần tử mang thông tin. Đối với các mạch tổ hợp TTL (Transistor-Transistor-Logic), các mức điện áp được mô tả trong Hình I.1



Hình I.1 Mức điện áp cao ứng với logic "1", thấp ứng với logic "0" của mạch TTL

Ta thường dùng ký hiệu V_H để chỉ mức cao, V_L để chỉ mức thấp. Trong mạch TTL, ta dùng mức cao, mức thấp để chỉ điện áp cao, điện áp thấp so với điện áp chuẩn chung. Các mức cao, thấp không phải là một giá trị cố định, mà là một vùng giới hạn cho phép. Ngoài phạm vi đã nêu, vùng không thuộc hai mức trên là vùng không chắc chắn, không xác định.



Hình I.2 Một phần tử mang thông tin

Vật mang thông tin về các giá trị "0" hoặc "1" là một mạch điện tử đặc biệt mà đầu ra của nó sẽ tương ứng với một trong hai mức trên, được gọi chung là Flip-Flop. Tùy theo yêu cầu sử dụng, các Flip-Flop có các khả năng thu nhận các tín hiệu vào và đưa tín hiệu ra theo những quy luật nhất định (Hình I.2)

I.4 Vài nét về thực hiện các phép tính trong hệ đếm nhị phân

Phép cộng và phép trừ hai số nhị phân 1 bit được thực hiện theo quy tắc nêu trong bảng sau:

A		B		Σ	Carry (Nhớ)
0	+	0	=	0	0
0	+	1	=	1	0
1	+	0	=	1	0
1	+	1	=	0	1

A		B		Hiệu	Borrow (Mượn)
0	-	0	=	0	0
0	-	1	=	1	1
1	-	0	=	1	0
1	-	1	=	0	0

I.4.1 Phép cộng và phép trừ

a) Phép cộng đại số các số hạng dấu phẩy cố định

Đối với phép cộng đại số: Thực hiện bình thường. Trong trường hợp có một toán hạng là một số âm, ta sử dụng mã ngược hoặc mã bù 2 của nó, hiệu chỉnh kết quả theo các quy tắc thông qua các ví dụ minh họa sau:

A = 0.10010	A = 0.10010	A = 0.10010
B = -0.11001	$(B)_{ng} = 1.00110$	$(B)_b = 1.00111$
$\Sigma = -0.00111$	$\Sigma = 1.11000$	$\Sigma = 1.11001$
	$(\Sigma)_{ng} = -0.00111$	$(\Sigma)_b = -0.00111$

Thấy rằng:

- Số biểu thị kết quả sẽ là mã thuận nếu là một số dương
- Số biểu thị kết quả là mã ngược nếu ta dùng mã ngược đối với số hạng âm và cho kết quả là một số âm
- Số biểu thị kết quả là một số bù 2 nếu dùng mã bù 2 đối với số hạng âm và kết quả là một số âm.

b) Phép cộng đại số các số hạng dấu phẩy động:

Đối với phép cộng đại số các số hạng dấu phẩy động, cần tiến hành các bước sau:

- Cân bằng phần đặc tính (số mũ) bằng cách dịch chuyển phần định trị
- Đặc tính của tổng bằng đặc tính chung
- Định trị của tổng bằng tổng các định trị
- Chuẩn hoá kết quả nếu cần.

I.4.2 Phép nhân và phép chia

a) Phép nhân:

Đối với phép nhân các toán hạng dấu phẩy tĩnh, việc quan trọng là phải xác định dấu của kết quả, theo đó dấu của kết quả bằng tổng modulo 2 (tức là lấy phần dư của phép chia tổng số cho 2) của các bit dấu. Trị số của tích là kết quả của phép tính tiền (dịch phải) và phép cộng.

Với các toán hạng có dấu phẩy động, dấu của tích được xác định như ở phép nhân với dấu phẩy tĩnh, sau đó tiến hành tìm tích số như sau:

- Cộng phần đặc tính (số mũ), kết quả là đặc tính của tích
- Nhân phần định trị, không để ý đến dấu của các toán hạng
- Chuẩn hoá kết quả nếu cần.

b) Phép chia:

Đối với phép chia các toán hạng dấu phẩy tĩnh, việc quan trọng là phải xác định dấu của kết quả, theo đó dấu của kết quả bằng tổng modulo 2 của các bit dấu. Trị số của thương số là kết quả của phép dịch trái và phép trừ.

Với các toán hạng có dấu phẩy động, dấu của thương số được xác định như ở phép chia với dấu phẩy tĩnh, sau đó tiến hành tìm thương số như sau:

- Trừ phần đặc tính (số mũ), kết quả là đặc tính của thương số
- Chia phần định trị, không để ý đến dấu của các toán hạng
- Chuẩn hoá kết quả nếu cần.

Nhận xét: Dễ dàng nhận thấy rằng các phép tính số học nêu trên chung quy lại vẫn chủ yếu là thực hiện phép cộng và phép dịch (shift).

1.5 Cấu trúc của hệ Vi xử lý và máy vi tính

1.5.1 Vài nét về lịch sử phát triển các trung tâm Vi xử lý

Sự xuất hiện của máy tính điện tử (MTĐT) vào khoảng năm 1948 đã mở ra một trang mới trong nghiên cứu khoa học nói chung và khoa học tính toán nói riêng. Nhưng phải mãi đến năm 1971, các hệ Vi xử lý mới bắt đầu xuất hiện. Sự ra đời của vi mạch Single chip 4-bit Microprocessor Intel[®]4004 (μ P4004) vào năm đó thực sự là một cuộc cách mạng trong ngành công nghiệp máy tính. Có thể nói μ P4004, với độ dài từ xử lý 4 bits, đã làm đổi thay toàn bộ cách nhìn nhận về các thiết bị đầu cuối của MTĐT, hay các cơ cấu chấp hành trong điều khiển quá trình. μ P4004 có thể quản lý trực tiếp 4K từ lệnh 8bit của bộ nhớ chương trình và 5120 bits bộ nhớ dữ liệu RAM. CPU còn có 16 thanh ghi chỉ số được sử dụng làm bộ nhớ tạm cho dữ liệu. Với tập lệnh gồm 46 lệnh, μ P4004 đã chiếm được nhiều ưu thế trong các ứng dụng thực tế lúc bấy giờ. Tiếp tục của dòng μ P 4bit này là μ P4040, có nhiều cải tiến mạnh mẽ so với μ P4004 và một loạt các chip chức năng, chip nhớ ra đời.

Trong giai đoạn tiếp theo từ năm 1974 đến 1977, Intel[®] đã đi đầu trong việc chế tạo các CPU 8bit, μ P8008, μ P8080 và đặc biệt là μ P8085, những CPU có BUS dữ liệu 8 bits và BUS địa chỉ 16 bits. Các loại CPU này đã có khả năng quản lý được 64K từ nhớ của bộ nhớ và 256 thiết bị ngoại vi. Điều đáng chú ý ở μ P8085 là công nghệ dồn kênh và chia sẻ thời gian hợp lý trên BUS đã cho phép đưa ra thêm những tín hiệu điều khiển rất mạnh, cho phép xây dựng những máy vi tính đầu tiên.

Khoảng thời gian năm 1978 đến năm 1982 là giai đoạn ra đời và phát triển mạnh mẽ của các trung tâm Vi xử lý 16 bits. Đặc biệt ở cuối giai đoạn này là sự xuất hiện các trung tâm Vi xử lý μ P8088, μ P8086, với khả năng xử lý dữ liệu 16 bits và BUS địa chỉ 20 bits, được sử dụng để tạo ra các máy vi tính XT, có ổ đĩa mềm để lưu giữ chương trình ứng dụng và dữ liệu.

Tiếp theo của giai đoạn này là sự phát triển vũ bão của các loại μ P80186, μ P80286, 80386SX, 80486-SX và 80486-DX, với nhịp đồng hồ lên đến 100MHz. Máy vi tính AT và các máy tính PC ra đời trong giai đoạn này dù giá thành còn rất cao, nhưng đã trở thành rất thông dụng trong đời sống con người.

Từ khoảng giữa những năm 1993 trở lại đây, các trung tâm vi xử lý Pentium ra đời, tốc độ ngày càng cao, với nhịp đồng hồ lên đến hàng GHz, và sự xuất hiện của các trung tâm xử lý đa phân luồng như các chip Pentium IV và hiện nay là các trung tâm vi xử lý đa lõi.

Những dấu mốc lịch sử sự phát triển của các CPU

Intel μ P4004 (1971)

Intel bắt đầu *phát triển vi xử lý từ năm 1969* theo dự án của nhà sản xuất máy tính Nhật Busicom. Kế hoạch ban đầu của Busicom là xây dựng 12 chip có khả năng tùy biến. Nhưng **kỹ sư Ted Hoff** của Intel đã hình thành ý tưởng về một thiết bị logic đa mục đích hoạt động hiệu quả hơn.

Ban đầu, Busicom nắm giữ các quyền liên quan tới bộ vi xử lý đó và trả cho Intel 60.000 USD. Nhận thấy tiềm năng của "bộ não" này, Intel quyết định trả lại số tiền trên để đổi lấy quyền thiết kế chip. Ngày 15/11/71, Intel giới thiệu μ P4004 ra thị trường thế giới với giá 200 USD.

μ P4004 có tốc độ 108 KHz với 2.300 bóng bán dẫn.

Intel μ P8008 (1972)

Phiên bản μ P8008 mạnh gấp đôi bộ vi xử lý μ P4004. Tạp chí *Radio Electronics* năm 1974 đã đề cập tới thiết bị mang tên Mark-8 sử dụng μ P8008. Mark-8 là một trong những máy tính đầu tiên dành cho người sử dụng gia đình - một hệ thống mà nếu theo các tiêu chuẩn ngày nay thì rất khó xây dựng, bảo trì và vận hành.

μ P8008 có tốc độ 200 KHz với 3.500 bóng bán dẫn.

Intel μ P8080 (1974)

Thiết bị xử lý μ P8080 đã trở thành bộ não của hệ thống máy tính cá nhân đầu tiên Altair - được cho là tên một địa điểm trong chương trình truyền hình Star Trek. Người đam mê điện toán có thể mua bộ thiết bị cho Altair với giá 395 USD. Trong nhiều tháng, hàng chục nghìn sản phẩm đã được xuất xưởng.

μ P8080 có tốc độ 2 MHz với 6.000 bóng bán dẫn.

Intel 8086 - 8088 (1978)

Bộ vi xử lý 8088 đã được trang bị cho chiếc máy tính cá nhân thương mại đầu tiên IBM PC. Thành công này đã đưa Intel vào bảng xếp hạng 500 doanh nghiệp hàng đầu thế giới của *Fortune*. Tạp chí này gọi Intel là một trong những "thành công kinh doanh của thập kỷ 70".

8088 có tốc độ 5 MHz, 8 MHz và 10 MHz với 29.000 bóng bán dẫn.

Intel 286 (1982)

286, còn được biết đến với tên gọi 80286, là chip đầu tiên của Intel có thể chạy tất cả các phần mềm được viết cho thiết bị xử lý trước đó. Sáu năm kể từ ngày được công bố, khoảng 15 triệu máy tính cá nhân sử dụng 286 đã được bán trên toàn thế giới.

286 có tốc độ 6 MHz, 8 MHz, 10 MHz và 12,5MHz với 134.000 bóng bán dẫn.

Intel 386 (1985)

Chip Intel 386 có 275.000 bóng bán dẫn, nhiều gấp 100 lần so với bộ xử lý 4004 ban đầu. Đây là chip 32 bit và có khả năng xử lý đa nhiệm, tức là nó có thể chạy nhiều chương trình khác nhau cùng một thời điểm.

386 có tốc độ 16 MHz, 20 MHz, 25 MHz và 33 MHz.

Intel 486 DX (1989)

Thế hệ vi xử lý 486 giúp người sử dụng không phải gõ lệnh thực thi và chuyển sang điện toán "trỏ và nhấn" (point-and-click) với tốc độ cao. Đây cũng là *chip đầu tiên tích hợp sẵn bộ đồng xử lý toán học*, hỗ trợ giảm tải các hàm phức tạp cho vi xử lý trung tâm.

486 DX có tốc độ 25 MHz, 33 MHz và 50 MHz với 1,2 triệu bóng bán dẫn.

Pentium (1993)

Pentium cho phép máy tính dễ dàng tích hợp những dữ liệu "thế giới thực" như giọng nói, âm thanh, ký tự viết tay và ảnh đồ họa. Tên gọi Pentium được nhắc nhiều trong các vở kịch, chương trình truyền hình và nhanh chóng trở thành một từ quen thuộc trong các gia đình.

Pentium có tốc độ 60 MHz và 66 MHz với 3,1 triệu bóng bán dẫn.

Pentium II (1997)

Pentium II tích hợp *Intel MMX - công nghệ được thiết kế đặc biệt để xử lý dữ liệu video, audio và đồ họa* một cách hiệu quả. Với chip này, người sử dụng có thể chụp, chỉnh sửa và chia sẻ ảnh số với bạn bè và người thân qua Internet, biên tập và thêm nội dung text, nhạc hoặc chuyển đổi cảnh trong phim...

Pentium II có tốc độ 200 MHz, 233 MHz, 266 MHz và 300 MHz với 7,5 triệu bóng bán dẫn.

Pentium III (1999)

Pentium III có 70 lệnh xử lý mới giúp tăng cường hiệu suất hoạt động của các ứng dụng ảnh, 3-D, audio, video và nhận dạng giọng nói. Nó hỗ trợ người sử dụng thực hiện các tác vụ dễ dàng hơn qua Internet, cho phép duyệt web và tải video chất lượng cao. Phiên bản Pentium này sử dụng công nghệ 0,25 micron.

Pentium III có tốc độ từ 650 MHz đến 1,2 GHz với 9,5 triệu bóng bán dẫn.

Pentium IV (2000)

Người sử dụng hệ thống máy tính *Pentium IV có thể tạo ra những bộ phim mang tính chuyên nghiệp, liên kết thời gian thực giữa video và thoại, chồng hình đồ họa 3D, nhanh chóng mã hóa nhạc và chạy các ứng dụng đa phương tiện...* Bộ vi xử lý sử dụng mạch 0,18 micron.

Pentium IV có tốc độ 1,3 GHz, 1,4 GHz, 1,5 GHz, 1,7 GHz và 1,8 GHz với 42 triệu bóng bán dẫn.

Tháng 8/2001, Pentium IV đạt mốc 2 GHz. Đến tháng 11/2002, chip này được trang bị công nghệ siêu phân luồng và có tốc độ 3,06 GHz, sau tăng lên 3,2 GHz vào năm 2003 và 3,4 GHz trong tháng 6/2004.

Chip lõi kép

Tháng 4/2005, Intel giới thiệu nền tảng sử dụng *bộ vi xử lý lõi kép đầu tiên* gồm chip Pentium Extreme Edition 840, xung nhịp 3,2 GHz, và *chipset 955X Express*.

Kết hợp với *công nghệ siêu phân luồng*, hệ thống có thể khai thác hiệu quả hơn những tài nguyên chưa được sử dụng hết.

Tháng 5/2005, chip Intel Pentium D lõi kép ra đời cùng chipset 945 Express, hỗ trợ những tính năng của các thiết bị điện tử tiêu dùng như âm thanh vòm, video độ phân giải cao và xử lý đồ họa tăng cường.

Tháng 5/2006, Intel công bố nhãn hiệu *Core 2 Duo*.

Tháng 7/2006: Intel giới thiệu 10 thiết bị xử lý Intel Core 2 Duo và *Core Extreme* cho hệ thống để bàn và máy tính xách tay, phục vụ người sử dụng doanh nghiệp, gia đình và người yêu điện toán tốc độ cao như giới game thủ... Những chip này nâng cao tới 40% hiệu suất hoạt động và tiết kiệm 40% điện năng so với bộ xử lý Intel Pentium tốt nhất.

Các bộ vi xử lý Core 2 Duo có 291 triệu bóng bán dẫn.

Tuyên bố ngày 4/8/2008 của Intel về thế hệ chip mới khẳng định chip Larrabee sẽ tích hợp “hơn 10 nhân (core) trong một lõi đơn”. Điều đó có nghĩa Intel sẽ loại bỏ hẳn thuộc tính “xung nhịp” để đánh giá tốc độ chip.

Xung nhịp trở thành thói quen đánh giá tốc độ chip từ gần 40 năm qua. Tại thời điểm 1970, những máy Atari hoặc Commodore chỉ hoạt động với tốc độ 1 MHz. Những chip hiện nay có xung nhịp lên tới 4GHz, tức là gấp 4.000 lần trước đây. Cùng với xung nhịp cao hơn là công hao phí cũng tăng lên, thể hiện bằng nhiệt lượng tỏa ra ngày càng nhiều trong những chip xung nhịp cao.

Việc áp dụng chip đa lõi để giảm xung nhịp được Intel và AMD áp dụng trong một vài năm gần đây. Những chip lõi kép và lõi tứ đã trở nên quen thuộc trên thị trường phổ thông. Tuy nhiên, việc phân định “cao thấp” trong mỗi dòng chip vẫn dựa trên xung nhịp của con chip đó với số lõi cố định.

Theo tuyên bố của Intel, kiến trúc Larrabee có thể đảm bảo nhu cầu điện toán cho đến năm 2015. Đối tượng được nhắm đến đầu tiên là những chuyên gia xử lý đồ họa, giới game thủ và những người đòi hỏi tính toán phục vụ nghe nhìn. Năng lực tính toán của chip Larrabee có thể đạt tới hàng nghìn tỉ phép tính mỗi giây.

Sản phẩm chip Larrabee thương mại đầu tiên sẽ có mặt trên thị trường vào khoảng cuối năm 2009 đến đầu năm 2010. Những laptop xuất hiện trong năm này sẽ dùng chip 10 lõi.

Kiến trúc lõi mới cũng đặt các hãng phần mềm vào một cuộc đua mới. Để tận dụng được sức mạnh của chip đa lõi, phần mềm cũng phải tự yêu cầu xử lý thành các luồng đến các lõi để xử lý song song. Các chương trình máy tính cũng phải có tập lệnh định hướng tự động chia yêu cầu thành 2, 4 hoặc 10 hướng theo số lõi có trong chip. Điều quan trọng là các luồng này phải được xử lý song song.

Thực tế, những PC dùng chip 10 lõi có thể nhanh chóng đến tay người tiêu dùng, nhưng phần mềm tối ưu với chip này thì phải đợi thêm thời gian để

phát triển. Để giải quyết vấn đề, các ngôn ngữ lập trình hỗ trợ chip đa lõi vẫn đang trong giai đoạn phát triển. Bên cạnh đó, các trung tâm nghiên cứu và trường đại học cũng nỗ lực hoàn chỉnh thuật toán về xử lý song song.

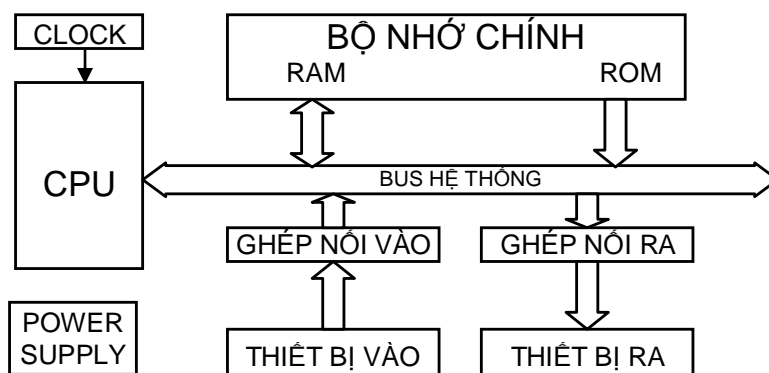
“Đại gia phần mềm” Microsoft nghiên cứu kiến trúc đa lõi từ năm 2001 đến nay cũng tỏ ý chùn chân và vẫn “dùng dằng” không muốn chuyển đổi quyết liệt. Hiện tại, một vài công ty game như Crytek và Valve cũng đã đưa sản phẩm của mình tiếp cận dần tới chip đa lõi, nhưng giải pháp của họ vẫn còn rất dặt dỏ.

1.5.2 Cấu trúc cơ bản của hệ Vi xử lý

Các khối chức năng cơ bản của một hệ Vi xử lý (hình I.3) gồm:

- Đơn vị xử lý trung tâm (CPU)
- Bộ nhớ ROM, RAM
- Thiết bị vào (nhập dữ liệu - Input device)
- Thiết bị ra (đưa dữ liệu ra - Output device)
- Ngoài ra còn phải kể đến khối tạo xung nhịp (Clock Generator) và khối nguồn (Power Supply).

Các khối chức năng cơ bản được nối với nhau qua một tập đường dây truyền dẫn tín hiệu điện gọi là *BUS hệ thống*. Dựa vào loại tín hiệu được truyền dẫn, BUS hệ thống được chia thành 3 BUS thành phần: *BUS địa chỉ*, *BUS dữ liệu* và *BUS điều khiển*. Thiết bị vào/ra thường được ghép nối với BUS hệ thống thông qua các mạch giao diện ghép nối (I/O Interface).



Hình I.3 Sơ đồ khối cấu trúc cơ bản hệ Vi xử lý

Đơn vị xử lý trung tâm (Central Processing Unit – CPU) là khối chức năng cơ bản nhất để tạo nên một hệ Vi xử lý hay máy tính cá nhân (Personal Computer – PC). Máy vi tính là một trong những ứng dụng

cụ thể của một hệ thống gọi là *Hệ Vi xử lý*.

- a) CPU thực hiện chức năng xử lý dữ liệu thông qua các hoạt động chính sau:
 - Đọc mã lệnh – đọc tập các bit thông tin “0” và “1” từ bộ nhớ chính

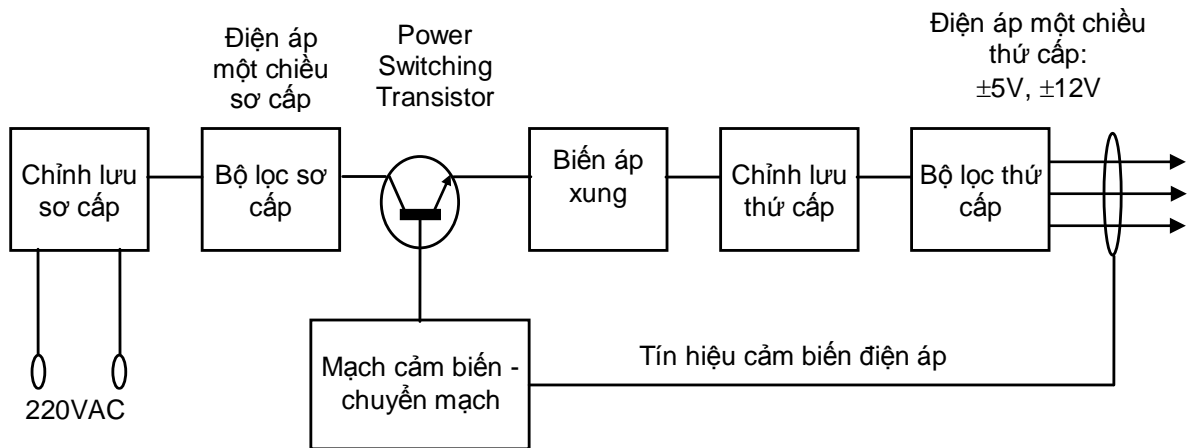
- Giải mã lệnh – tạo các xung điều khiển tương ứng với mã lệnh để điều khiển hoạt động của các khối chức năng khác
- Thực hiện từng bước các thao tác xử lý dữ liệu theo yêu cầu của lệnh.

Bên trong CPU có các thanh ghi (*Registers*):

- Thanh ghi con trỏ lệnh IP (*Instruction Pointer*), trong các trung tâm vi xử lý trước đây còn gọi là thanh đếm chương trình PC (*Program Counter*) chứa địa chỉ của lệnh kế tiếp cần được thực hiện trong tuần tự thực hiện chương trình
 - Các thanh ghi đa dụng khác GPRs (*General Purpose Registers*) để lưu trữ tạm thời dữ liệu, kết quả trung gian hay trạng thái của hệ thống cùng với đơn vị số học và logic ALU (*Arithmetic and Logic Unit*) thực hiện các thao tác xử lý dữ liệu
 - Đơn vị điều khiển CU (*Control Unit*) là thành phần phức tạp nhất, có chức năng giải mã lệnh và tạo các tín hiệu điều khiển hoạt động của toàn hệ thống.
- b) Bộ nhớ chính được tổ chức từ các từ nhớ, theo chuẩn của IBM/PC từ nhớ có độ dài 1 byte (8 bits). Bộ nhớ này gồm các chip nhớ chỉ đọc ROM (*Read Only Memory*) và các chip nhớ truy xuất ngẫu nhiên RAM (*Random Access Memory*) có tốc độ truy cập nhanh. Bộ nhớ được sử dụng để chứa các chương trình và các dữ liệu điều khiển hoạt động của hệ thống. các chương trình ứng dụng và dữ liệu có thể được chứa ở ROM hoặc RAM, các kết quả trung gian hay kết quả cuối cùng của các thao tác xử lý có thể được chứa trong các thanh ghi đa dụng hoặc trong khối nhớ RAM
- c) Các mạch ghép nối vào/ra là các mạch điện tử cho phép CPU trao đổi dữ liệu với các thiết bị ngoại vi như bàn phím, màn hình, máy in... làm giao diện với người dùng hoặc các bộ chuyển đổi số-tương tự DAC (*Digital/Analog Converter*), chuyển đổi tương tự-số ADC (*Analog/Digital Converter*), các mạch vào/ra dữ liệu dạng số DO (*Digital Outputs*), DI (*Digital Inputs*)...
- d) Hệ Vi xử lý còn có một mạch tạo xung nhịp gọi là đồng hồ hệ thống (*Clock Generator*) điều khiển và duy trì hoạt động đồng bộ của tất cả các khối chức năng. Bộ tạo xung này được điều khiển bằng một mạch thạch anh có tần số thích hợp và đảm bảo tần số làm việc ổn định cho toàn bộ hệ thống.
- e) Một khối nguồn nuôi (*Power Supply*) cung cấp năng lượng cho hệ thống từ mạng điện lưới.

Bộ nguồn của các hệ Vi xử lý thông thường là bộ nguồn xung với kỹ thuật đóng-ngắt dùng bán dẫn công suất (*Switching Power Supply*), vừa gọn

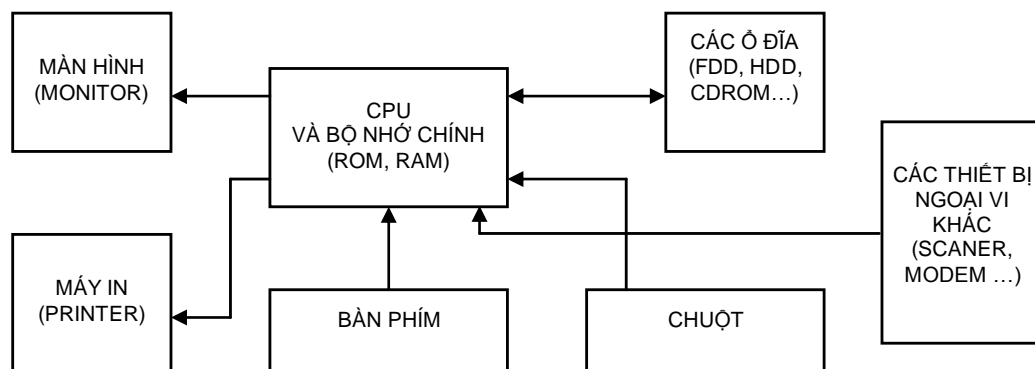
nhẹ, công suất lớn lại vừa đảm bảo độ gợn sóng nhỏ nhất và khả năng chống nhiễu cao. Hình 1.4 là sơ đồ khối của bộ nguồn đóng-ngắt. Điện áp lưới (220VAC) được chỉnh lưu trực tiếp, lọc bằng tụ hoá để cung cấp cho một bộ dao động tần số cao (từ 20KHz đến 40KHz). Các xung điện áp tần số cao được chuyển sang biến áp xung công suất hạ áp. Điện áp ở lõi ra của biến áp xung được chỉnh lưu và lọc thành điện áp nguồn một chiều cung cấp cho hệ thống. Nguyên lý ổn áp ở đây là thay đổi độ rộng của các xung có tần số ổn định, do vậy sự dao động của điện áp đầu ra khi có tải được chuyển qua bộ cảm biến để điều chỉnh độ rộng này, đảm bảo sự ổn định của điện áp ra.



Hình 1.4 Sơ đồ khối bộ nguồn nuôi máy tính

1.5.3 Từ hệ Vi xử lý đến máy vi tính PC

Trong thực tế, các hệ Vi xử lý hiện đại được trang bị thêm nhiều thiết bị ngoại vi tiện dụng tùy theo yêu cầu, mục đích sử dụng và có giao diện thân thiện với con người. Đó là các máy vi tính PC. Cũng có thể là những hệ Vi xử lý chuyên dụng cho những mục đích tính toán hay điều khiển.



Hình 1.5 Máy Vi tính PC

a. **Máy tính xử lý dữ liệu:** Là các máy tính được dùng để tính toán, xử lý các dữ liệu như quản lý nhân viên trong cơ quan, tính toán tiền lương, tính toán kết cấu công trình, phân tích dữ liệu trong kinh doanh, v.v... Quan điểm đúng cho rằng máy tính chỉ gồm CPU và bộ nhớ chính, còn các thiết bị phụ trợ khác như bàn phím, máy in, các ổ đĩa cứng, đĩa mềm, ổ CD, chuột, màn hình, máy in..., là những thiết bị ngoại vi. Các chương trình để xử lý dữ liệu được lưu giữ trong bộ nhớ chính hoặc trong các ổ đĩa, có nhiệm vụ xử lý những dữ liệu được người dùng nhập vào, và đưa kết quả xử lý ra màn hình, in ra giấy hoặc lưu giữ trong các ổ đĩa. Để đánh giá tính năng và chất lượng của các máy này, ta thường căn cứ vào tốc độ xử lý dữ liệu, dung lượng bộ nhớ, ổ đĩa, chất lượng màn hình, máy in v.v...

b. **Máy tính là bộ xử lý số:** Đối với các máy tính này, thời gian dành cho xử lý dữ liệu rất nhỏ, còn thời gian để tính toán, xử lý các số liệu lại vô cùng lớn. Các máy tính loại này được sử dụng chủ yếu trong các cơ quan dự báo, như dự báo khí tượng, thủy văn, trong tính toán quỹ đạo bay của tên lửa, máy bay, tàu thủy, v.v... hay trong các phòng nghiên cứu khoa học. Những máy tính loại này thông thường thực hiện những chương trình tính toán khổng lồ, nên chúng được trang bị các CPU rất mạnh và các thiết bị ngoại vi, bộ nhớ ngoài rất lớn. Đó là những siêu máy tính (*Supercomputer*).

c. **Máy tính đo lường và điều khiển:** Sự phát triển nhanh chóng của các hệ thống máy tính đã tạo ra những ứng dụng lớn lao trong các hệ thống đo lường và điều khiển tự động. Đối với các ứng dụng thông thường như trong các dụng cụ gia dụng, từ Tivi, điều hoà nhiệt độ, máy giặt v.v... Đó là những máy tính nhỏ được chế tạo dưới dạng một vi mạch (*Single-Chip Microcomputer*). Tuy nhiên, cũng cần phải tính đến những máy tính này trong các thiết bị hiện đại và phức tạp như trong các hệ thống tự động lái máy bay (*Autopilot*), tàu thủy, tên lửa...

d. **Căn cứ vào tính năng kỹ thuật và các chỉ tiêu về kích thước:** Các máy tính còn được chia ra thành *máy tính lớn* để giải các bài toán cực lớn với tốc độ rất nhanh, *máy tính nhỏ* sử dụng trong gia đình, trong trường học hay các tính toán thông dụng, điều khiển các quá trình công nghệ vừa và nhỏ.

Cũng cần nhắc đến ở đây một sự khác biệt giữa hai khái niệm hệ Vi xử lý và máy vi tính: các máy vi tính luôn luôn được trang bị một phần mềm cơ bản là Hệ điều hành, ví dụ: MS-DOS hay các phiên bản điều hành đa nhiệm (MS-WINDOWS của hãng phần mềm Microsoft, hoặc các hệ điều hành của các hãng khác...) và các chương trình hay phần mềm ứng dụng, trong khi các hệ Vi xử lý chỉ cần trang bị một chương trình Monitor (chương trình giám sát) đơn giản được ghi trong bộ nhớ ROM.

CÂU HỎI ÔN TẬP CHƯƠNG I.

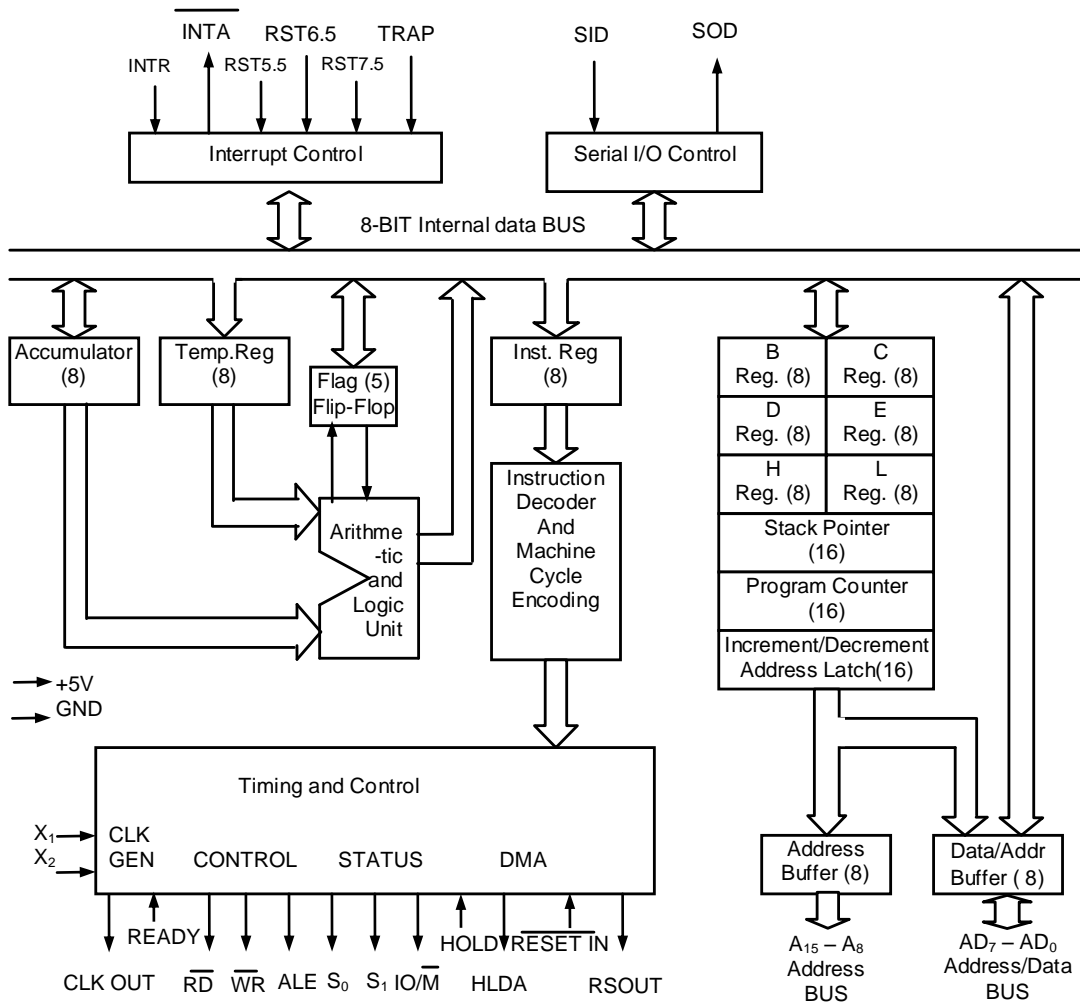
1. Tại sao hệ đếm nhị phân lại được chọn để biểu diễn thông tin trong máy tính số?
2. Trình bày phương pháp chuyển đổi một số thực dấu phẩy động trong hệ đếm thập phân sang hệ đếm nhị phân.
3. Trình bày một số chuẩn được sử dụng để biểu diễn thông tin số và thông tin không số trong các hệ vi xử lý.
4. Chuẩn IEEE 754 biểu diễn một số thực chính xác đơn và số thực chính xác kép.
5. Dạng mở rộng của chuẩn IEEE 754 biểu diễn số thực.
6. Chuyển đổi số $-46,725$ sang cách biểu diễn số thực chính xác đơn theo chuẩn IEEE 754.
7. Phát biểu bản chất vật lý của biểu diễn thông tin trong hệ Vi xử lý, cho ví dụ cụ thể.
8. Trình bày các nguyên tắc thực hiện 4 phép tính số học trên các dữ liệu nhị phân dạng dấu phẩy tĩnh, và dấu phẩy động.
9. Sơ đồ cấu trúc cơ bản của một hệ Vi xử lý, tổ chức và chức năng các thanh ghi trong hệ Vi xử lý.

CHƯƠNG II. CÁC ĐƠN VỊ VI XỬ LÝ TRUNG TÂM (CPU – Central Processing Unit)

Vì hầu hết các máy vi tính đang được sử dụng ở Việt nam đều được xây dựng trên cơ sở của các chip xử lý của hãng Intel[®], nên tài liệu này cũng giới hạn sự trình bày trong khuôn khổ các trung tâm vi xử lý của hãng này. Các độc giả có thể tìm hiểu thêm về các trung tâm vi xử lý của các hãng khác như Motorola, AMD,...ở một số tài liệu tham khảo liệt kê ở phần cuối giáo trình.

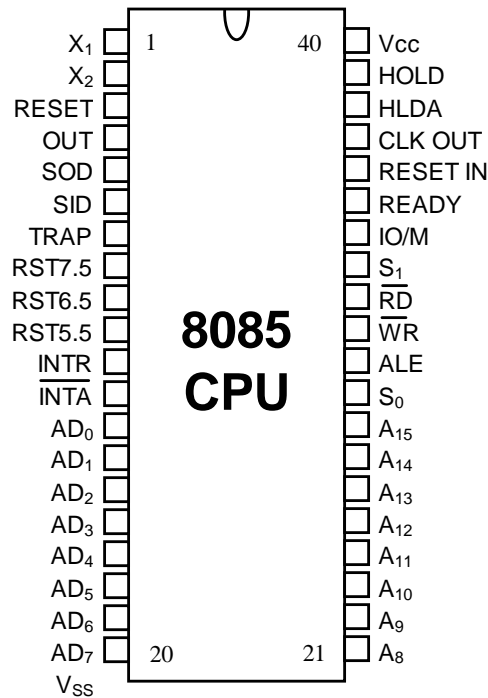
Khi CPU được chế tạo từ một mạch vi điện tử có độ tích hợp rất cao thì nó được gọi là bộ Vi xử lý (μP – *Microprocessor*). Trong quá trình phát triển, hãng Intel[®] đã cho ra đời nhiều thế hệ μP từ đơn giản đến phức tạp, từ thông dụng đến chuyên dụng. Tính phát huy và kế thừa luôn được coi trọng trong quá trình này, vì vậy, các chương trình ứng dụng chuẩn phần lớn có thể thực hiện được trên bất kỳ máy vi tính được xây dựng từ thế hệ μP nào.

II.1 Trung tâm Vi xử lý $\mu P8085$



Hình II.1a) Sơ đồ khối cấu trúc $\mu P8085$

Hình II.1a) là sơ đồ khối cấu trúc của μP8085 , Hình II.1b) là sơ đồ nối chân của μP8085 . Khác với các loại μP xuất hiện trước đó như μP8008 hay μP8080 , μP8085 có những bước phát triển có tính đột phá như sau:



Hình II.1b) Sơ đồ nối chân của μP8085

1. Cơ cấu ngắt theo nhiều mức khác nhau được hình thành qua một khối điều khiển ngắt, tạo ra một vector ngắt tránh được sự chèn nhau do lệnh RET N trên BUS dữ liệu. Tín hiệu nhận biết yêu cầu ngắt INTA được tạo bởi khối điều khiển ngắt, chứ không phải từ mạch phụ 8228 như ở μP8080 .
2. Các tín hiệu điều khiển ghi/đọc $\overline{\text{WR}}$ và $\overline{\text{RD}}$ được tạo ra từ khối định thời và điều khiển chức năng. Các tín hiệu INTA , $\overline{\text{WR}}$ và $\overline{\text{RD}}$ được tạo ngay trong CPU, chứ không do mạch phụ trợ bên ngoài.
3. μP8085 có mạch tạo xung đồng hồ được tích hợp ngay trong CPU.
4. Khối chức năng điều khiển vào/ra nối tiếp được tích hợp cũng cho phép μP8085 thực hiện các lệnh vào/ra dữ liệu nối tiếp mà nhiều khi không cần đến sự hỗ trợ của vi mạch chuyên dụng.
5. Đặc biệt hơn, μP8085 có hai thanh ghi đệm địa chỉ, đó là thanh ghi đệm $A_{15} - A_8$ và thanh ghi đệm $AD_7 - AD_0$ cho cả dữ liệu và địa chỉ. Việc dồn kênh như trên tạo điều kiện cho những chân chức năng khác được tạo thêm, làm tăng thêm sức mạnh cho CPU.

II.1.1 Các nhóm tín hiệu trong μP8085

$A_8 - A_{15}$. Nhóm tín hiệu ra: 8 bit cao của địa chỉ, các chân này là các chân được nối với bên ngoài qua mạch 3 trạng thái. Các phần tử 3 trạng thái sẽ được đặt ở trạng thái high-Z (còn gọi là trạng thái không kết nối) trong các trường hợp một trong các tín hiệu HOLD hay HALT là tích cực.

$AD_0 - AD_7$. Nhóm tín hiệu dồn kênh cho các tín hiệu địa chỉ và tín hiệu dữ liệu theo chia sẻ thời gian, 3 trạng thái. Ở giai đoạn đầu của chu kỳ máy, T_1 của M_1 , sẽ là byte thấp của 16 bit địa chỉ từ A_0 đến A_7 .

ALE (Address Latch Enable). Tín hiệu ra qua mạch 3 trạng thái. Được sử dụng để chốt byte thấp của tín hiệu địa chỉ ($A_0 - A_7$). Tín hiệu này được tạo ra trong giai đoạn đầu tiên của chu kỳ máy, T_1 của M_1 , và cũng được dùng để chốt các tín hiệu trạng thái S_0 và S_1 khi cần thiết.

S_0, S_1 (Data BUS Status). Là các tín hiệu chỉ trạng thái của các chân thuộc BUS dữ liệu trong mỗi chu kỳ máy. Tổ hợp của hai tín hiệu này cũng cho biết trạng thái của CPU như sau:

S_1	S_0	Trạng thái hoạt động của BUS dữ liệu
0	0	Trạng thái HALT
0	1	CPU đang thực hiện thao tác WRITE
1	0	CPU đang thực hiện thao tác READ
1	1	CPU đang thực hiện thao tác nhận lệnh (Instruction Fetch)

\overline{RD} (Read). Chân ra 3 trạng thái. Nằm trong nhóm tín hiệu điều khiển. Tín hiệu tích cực khi CPU tiến hành đọc dữ liệu từ bộ nhớ hoặc từ thiết bị ngoại vi. Trong chế độ HALT hoặc DMA, chân ra này ở trạng thái high-Z.

\overline{WR} (Write). Chân ra 3 trạng thái. Nằm trong nhóm tín hiệu điều khiển. Tín hiệu tích cực khi CPU tiến hành ghi dữ liệu vào bộ nhớ hoặc đưa dữ liệu ra thiết bị ngoại vi. Trong các chế độ HALT hoặc DMA, chân ra này ở trạng thái high-Z.

IO/\overline{M} . Trạng thái logic của đầu ra này cho biết CPU đang làm việc với thiết bị ngoại vi hay với bộ nhớ. Nếu là logic “1”, CPU đang truy cập thiết bị vào/ra, còn nếu là “0”, CPU đang truy cập bộ nhớ. Kết hợp với hai đầu ra \overline{RD} và \overline{WR} để tạo ra các tín hiệu I/OR , I/OW , $MEMR$, và $MEMW$ trong trường hợp sử dụng địa chỉ tách biệt đối với thiết bị vào/ra. Nằm trong nhóm tín hiệu điều khiển, IO/\overline{M} cũng là đầu ra 3 trạng thái.

Interrupts. μP8085 có ngắt đa mức. Có 5 chân ngắt tất cả: (INTR, RST5.5, RST6.5, RST7.5 và TRAP). Ngoài chân ngắt không che được là TRAP, các chân khác đều có thể che hoặc không che nhờ lập trình phần mềm.

- **INTR:** Chân nhận yêu cầu ngắt từ bên ngoài, được đáp ứng theo nguyên tắc polling hoặc vectoring thông qua lệnh RST
- **Các yêu cầu ngắt RST:** Có 3 đầu vào yêu cầu ngắt với các mức ưu tiên khác nhau là RST7.5, RST6.5 và RST5.5. Khi yêu cầu ngắt xuất hiện tại các chân này, CPU tự động chuyển đến các vector ngắt tương ứng. Cụ thể như sau:
 - **RST5.5** là mức ưu tiên thấp nhất, phản ứng theo mức điện áp trên chân yêu cầu ngắt, địa chỉ vector ngắt này nằm ở ô nhớ có địa chỉ $2C_H$.
 - **RST6.5:** Ngắt ưu tiên thấp thứ 2, phản ứng theo mức điện áp trên chân yêu cầu ngắt, địa chỉ vector ngắt này nằm ở ô nhớ 34_H
 - **RST7.5:** Mức ưu tiên cao nhất. Phản ứng theo sườn lên của xung yêu cầu ngắt. Sườn lên của xung này tác động lên một Flip-Flop, mạch này giữ lại yêu cầu ngắt cho đến khi được xóa nhờ tín hiệu nhận biết yêu cầu ngắt Acknowledge. Địa chỉ của vector ngắt này nằm ở ô nhớ $3C_H$
- **TRAP:** Là chân nhận yêu cầu ngắt không che được (dĩ nhiên là nó có mức ưu tiên cao nhất). Địa chỉ của vector ngắt này ở ô nhớ 24_H .

INTA. Tín hiệu ra nhận biết yêu cầu ngắt tại chân INTR. Các yêu cầu ngắt RST5.5, RST6.5, RST7.5 và TRAP không tác động đến INTA.

HOLD. Trạng thái logic “1” ở chân này là yêu cầu của thao tác DMA. Các đầu ra \overline{RD} , \overline{WR} , IO/M và ALE sẽ được đưa về trạng thái high-Z.

HLDA. Tín hiệu nhận biết yêu cầu HOLD.

RESET IN. Logic thấp “0” ở đầu vào của chân này yêu cầu tái khởi động hệ Vi xử lý. Do tác động của tín hiệu RESET IN tích cực, giá trị của thanh đếm chương trình PC sẽ được nạp lại là 0000_H . Các mặt nạ ngắt và tín hiệu HLDA cũng được tái thiết lập về giá trị mặc định.

RESET OUT. Đầu ra nhận biết hệ Vi xử lý được tái khởi động. Dùng tín hiệu này để tái khởi động toàn bộ hệ thống.

READY. Logic “1” ở đầu vào này thông báo trạng thái sẵn sàng cung cấp dữ liệu cho CPU hoặc nhận dữ liệu từ CPU của các thiết bị ngoại vi.

SID (Serial Input Data). Là cổng vào của dữ liệu nối tiếp của hệ Vi xử lý. Bit hiện diện tại cổng này được đọc vào CPU nhờ lệnh RIM, bit sẽ được đưa vào bit cao của Acc (MSB).

SOD (Serial Output Data). Bit cao (MSB) của Acc được truyền ra ngoài chân này khi sử dụng lệnh SIM.

X₁, X₂. Lối nối thạch anh hoặc một mạch dao động để tạo xung nhịp cho CPU. Có thể sử dụng thạch anh có tần số dao động trong khoảng từ 0.5 đến 3MHz.

CLK. Đầu ra của xung nhịp, có thể làm xung nhịp cho các thành phần chức năng khác trong hệ Vi xử lý.

Vcc, Vss. Lối nối nguồn +5V và GND cho μ P8085. Cũng cần nhắc lại rằng, μ P8085 chỉ cần một nguồn nuôi duy nhất là +5V, khả năng cung cấp dòng của nguồn cần được thiết kế tùy theo nhu cầu của toàn hệ Vi xử lý.

II.1.2 Khái niệm và bản chất vật lý của các BUS trong hệ Vi xử lý

Hoạt động của một hệ Vi xử lý thực chất là việc trao đổi và xử lý các giá trị nhị phân giữa các thành phần, các khối và các mạch vi điện tử trong toàn bộ hệ thống. Như đã biết, các giá trị nhị phân (hoặc “0” hoặc “1”) được thể hiện qua mức điện áp so với một chuẩn nhất định. Giá trị “0” tương ứng với mức điện áp thấp (từ 0V đến +0,8V) và giá trị “1” tương ứng với mức điện áp từ khoảng +3V đến +5V. Để biểu diễn một số liệu nhị phân, các phần tử mang thông tin được liên kết kề nhau theo nhóm (ví dụ 1byte là 8 bits). Để đảm nhận công việc di chuyển các dữ liệu này trong toàn bộ hệ thống, có các đường dây truyền dẫn điện chuyên dụng được ghép song song thành hệ thống, mỗi dây truyền dẫn dành riêng cho 1 bit. Tập các đường truyền dẫn dành riêng cho các tín hiệu có cùng chức danh (dữ liệu, địa chỉ, điều khiển) được gọi là BUS. Như vậy, trong một hệ Vi xử lý, có ba loại BUS: BUS dữ liệu, BUS địa chỉ và BUS điều khiển. Các BUS này hợp lại thành BUS hệ thống.

Từ khái niệm trên, dễ dàng suy ra bản chất vật lý của các BUS trong một hệ Vi xử lý: Đó là các *đường truyền dẫn điện, có thể dưới các dạng cáp nhiều sợi, đường dẫn trong các bảng mạch in v.v...* Khả năng và chất lượng dẫn điện của các đường truyền dẫn này đóng vai trò quan trọng và quyết định đối với hoạt động của một hệ Vi xử lý. Đường truyền dẫn kém, trở kháng cao có thể gây ra sự suy giảm của tín hiệu điện dẫn đến các hiện tượng mất mát hoặc sai dữ liệu, rất nguy hiểm.

BUS là đường dẫn điện nội bộ mà theo đó các tín hiệu được truyền từ bộ phận này đến các bộ phận khác trong hệ Vi xử lý. Có 3 loại BUS trong một hệ Vi xử lý cũng như trong máy tính PC:

- BUS dữ liệu truyền dữ liệu theo hai chiều giữa bộ nhớ và trung tâm Vi xử lý, giữa các thiết bị ngoại vi và Trung tâm Vi xử lý

- BUS địa chỉ truyền dẫn các tín hiệu xác định vị trí nhớ trong bộ nhớ, hoặc một thiết bị ngoại vi
- BUS điều khiển truyền các tín hiệu điều khiển đến các bộ phận hoặc các thành phần chức năng cần được điều khiển

Các BUS được xây dựng bằng cách sử dụng các khe cắm, theo một yêu cầu rất khắt khe đối với từng tiếp điểm. Đối với các khe cắm, các tiếp điểm tương ứng sẽ được nối với nhau bằng các dây dẫn hoặc đường dẫn song song trên mạch in. Nhờ vậy, khi dữ liệu được truyền đi, tất cả các bit (8, 16, 32, hay 64) đều được truyền đi đồng thời, cùng một hướng (truyền dẫn song song).

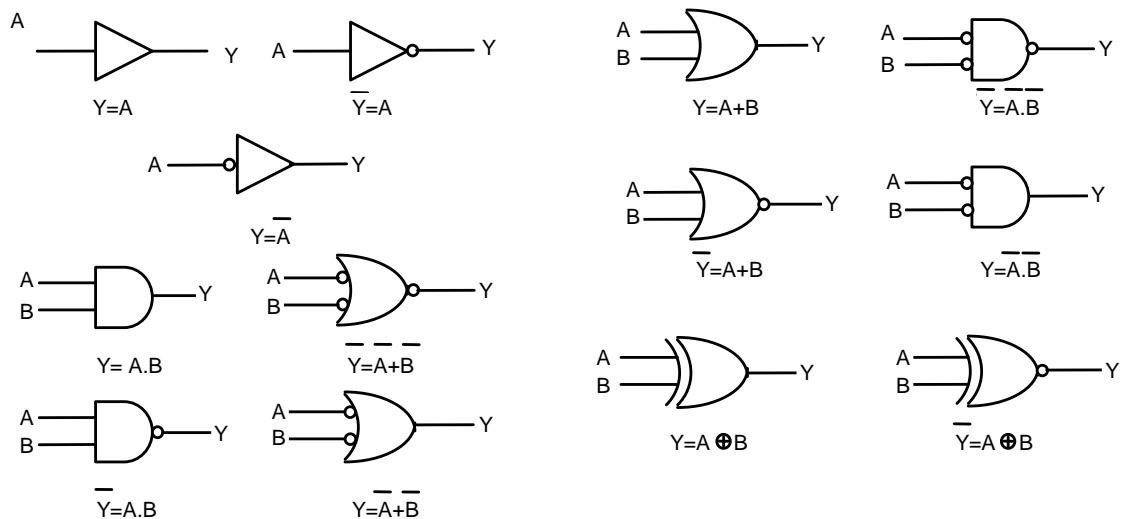
Cũng cần nói thêm rằng, trong máy PC, có 3 loại cấu trúc BUS thường gặp là ISA (Industrial Standard Architecture) EISA (Enhanced ISA) và PCI (Peripheral Component Interconnect).

II.1.3 Các mạch 3 trạng thái, mạch chốt và mạch khuếch đại BUS 2 chiều

Trước tiên, cũng cần nhắc lại một số linh kiện điện tử số cơ bản sử dụng trong máy vi tính. Nhờ công nghệ cao, các linh kiện có độ tích hợp lớn và rất lớn đã ra đời, nhưng không thể không nhìn lại một số mạch tổ hợp thực hiện những hàm logic cơ bản nhất.

a) Các cổng logic

Ký hiệu các mạch được chỉ ra trên Hình II.2, cùng biểu thức hàm logic gồm: mạch đệm (buffer), mạch đảo (NOT), mạch và (AND), mạch NAND, mạch hoặc (OR), mạch NOR và mạch XOR.



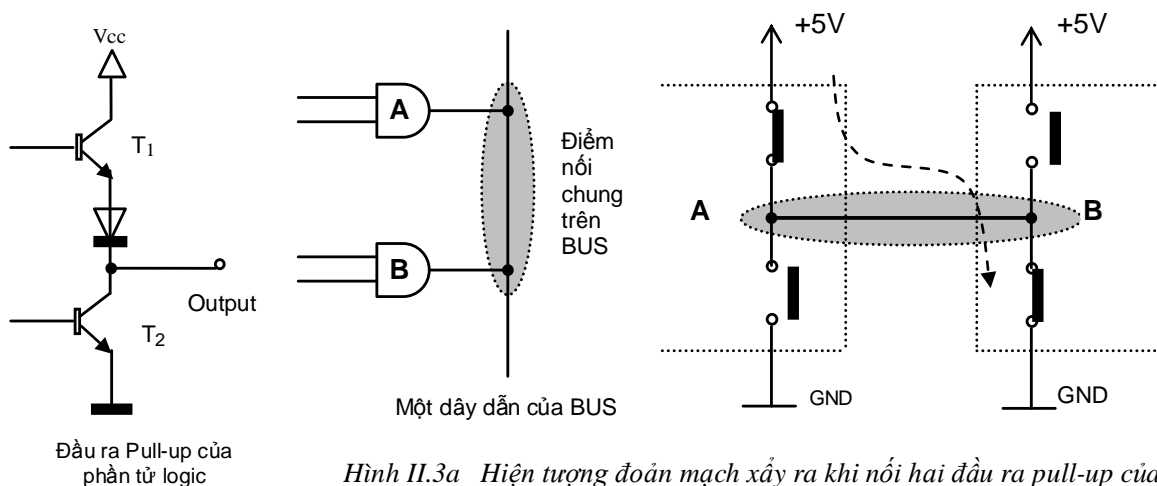
Hình II.2 Một số cổng Logic thông dụng

Các loại mạch này thường được sử dụng để tạo nên các mạch tổ hợp logic thực hiện các chức năng lập mã, giải mã, dồn kênh và phân kênh. Cũng cần lưu ý rằng, một số mạch chức năng như giải mã dồn kênh và phân kênh đã được các hãng tích hợp dưới dạng các mạch MSI. Một số mạch có thể kể ra như mạch giải mã 3/8 SN74138, mạch dồn kênh 74151, mạch cộng, và mạch nhân v.v...

b) Mạch 3 trạng thái (Tristate Component)

Trong hệ Vi xử lý, có nhiều khối chức năng cần thông tin, nhưng tại một thời điểm, bao giờ cũng chỉ có một khối đưa tín hiệu ra (dữ liệu) và một số hạn chế các khối thu nhận tín hiệu. Thay vì nối dây dẫn liên kết các khối qua từng đôi phần tử một, các tín hiệu này được đưa lên BUS. Với các cổng logic thông thường, không thể nối trực tiếp chúng lên cùng một đường dây vì sẽ xảy ra tranh chấp BUS vì đoạn mạch. Ví dụ đầu ra của phần tử A là “1” trong lúc đầu ra của phần tử B là “0”. (Hình II.3). Các đầu ra của loại mạch này đều theo cấu trúc *pull-up*, nghĩa là có hai transistor được nối nối tiếp với nhau (xem hình vẽ), emitter của transistor này qua một diode rồi đến đầu ra, đến collector của transistor kia. Với hai trạng thái logic “1” và “0”, tương ứng sẽ là T_1 mở, T_2 đóng và ngược lại, T_2 mở và T_1 đóng. Trên hình vẽ II.2 hiện tượng nguy hiểm xảy ra khi lối ra của phần tử logic A là “1”, các khoá mở hay đóng tương đương việc transistor thông báo hoà hay ngắt, lối ra của phần tử logic B là “0” và hiện tượng đoản mạch xảy ra.

Để tránh hiện tượng này, một loại cổng logic gọi là cổng 3 trạng thái (tri-state gate) được sử dụng cho lối ra của các khối nối chung vào BUS. Hình II.3a là một phần tử đảo đầu ra 3 trạng thái. Hình II.3b là sơ đồ tương đương của trạng thái high-Z, tương ứng với trường hợp đầu ra bị tách khỏi BUS.

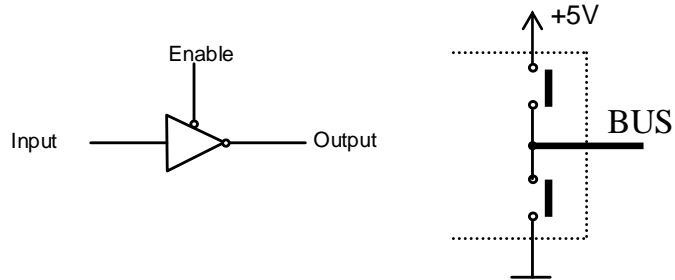


Hình II.3a Hiện tượng đoản mạch xảy ra khi nối hai đầu ra pull-up của hai phần tử lên cùng một đường dây của BUS

Như vậy, để tránh xung đột trên BUS, các phần tử có đầu ra nối với BUS cần phải đưa qua cổng 3 trạng thái. Hoạt động của mạch đảo ba trạng thái được mô tả qua bảng chân lý sau, trạng thái HZ có thể coi như đầu vào của mạch được tách hẳn khỏi đầu ra, hay còn gọi là trạng thái không kết nối.

Input	Enable	Output
0	1	HZ
1	1	HZ
0	0	1
1	0	0

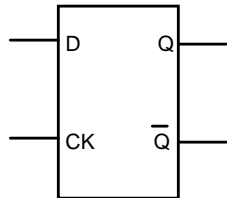
Bảng chân lý mạch 3 trạng thái



Hình II.3b Phần tử đảo 3 trạng thái và sơ đồ tương đương đầu ra của phần tử ở trạng thái high-impedance (HZ)

c) Mạch chốt, thanh ghi:

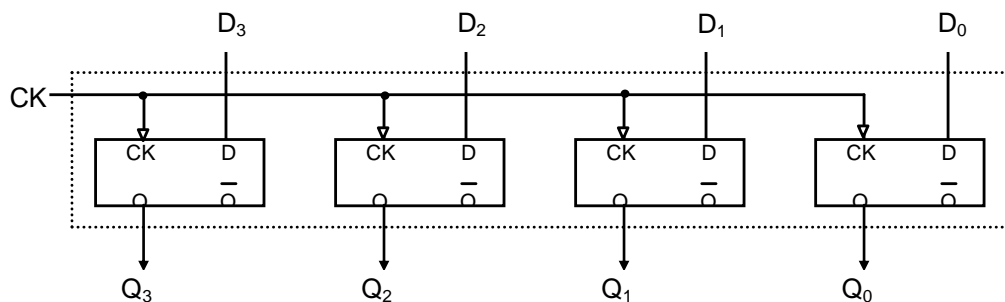
Mạch chốt là một mạch gồm các phần tử có khả năng lưu giữ các giá trị “0” hoặc “1” ở lối ra. Có thể dùng D Flip-Flop làm một mạch chốt với tín hiệu để chốt dữ liệu tại đầu ra Q theo bảng giá trị chân lý sau:



D	CK	Q	\bar{Q}
X	0	Q	Q_n
0	1	0	1
1	1	1	0

Hình II.4 Mạch chốt (hay phần tử nhớ) D Flip-Flop

Biết rằng $Q^{n+1} = D$ với tín hiệu điều khiển là sự xuất hiện sườn dương của xung nhịp CK. Như vậy, giá trị logic (0 hoặc 1) tại D đã được chuyển sang đầu ra Q (chốt). Nếu CK giữ nguyên trạng thái bằng “1”, thì trạng thái đầu ra Q được giữ nguyên. Như vậy, giá trị logic của D đã được lưu giữ ở Q (nhớ).



Hình II.5 Thanh ghi 4 bits

Thanh ghi (*Register*) là một nhóm các mạch Flip-Flop được nối song song với nhau, có thể lưu giữ được các số liệu nhị phân. Hình II.5 là sơ đồ một thanh ghi lưu giữ số liệu nhị phân 4 bits được tạo từ 4 phần tử D Flip-Flop.

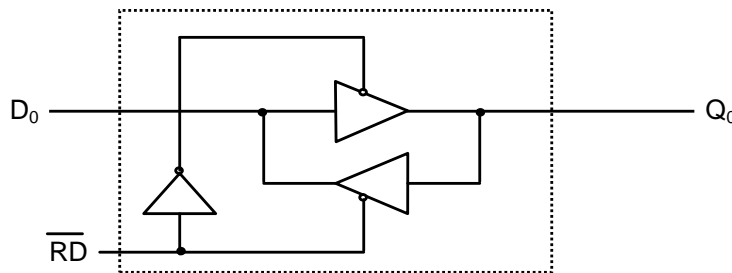
Một số liệu nhị phân bất kỳ từ D_3 đến D_0 sẽ được chốt sang các lối ra từ Q_3 đến Q_0 mỗi khi có một sườn lên của xung nhịp CK được đưa tới lối vào xung nhịp. Từ nhị phân này được lưu giữ ở lối ra cho đến khi có dữ liệu mới được đưa vào lối D và có xuất hiện sườn lên của xung nhịp CK.

d) Mạch khuếch đại BUS 2 chiều

Trên cơ sở của các mạch 3 trạng thái, các mạch khuếch đại BUS hai chiều được xây dựng theo nguyên lý sau:

Hai phần tử 3 trạng thái sẽ được ghép ngược với nhau (Hình II.6), chân điều khiển sẽ dùng tín hiệu đảo của tín hiệu đọc \overline{RD} . Khi xuất hiện tín hiệu \overline{RD} , dữ liệu được phép đi từ Q_0 sang D_0 , ngược lại, tín hiệu chỉ được phép đi từ D_0 sang Q_0 và cho phép CPU đưa tín hiệu ghi dữ liệu ra ngoài.

Ghép nối đủ số phần tử cho tất cả các dây dữ liệu, ta có mạch khuếch đại BUS hai chiều. Trong thực tế, mạch có chức năng trên đã được tích hợp theo chuẩn của TTL, được ký hiệu là 8228 hoặc 8288 (Octal BUS Transceiver).



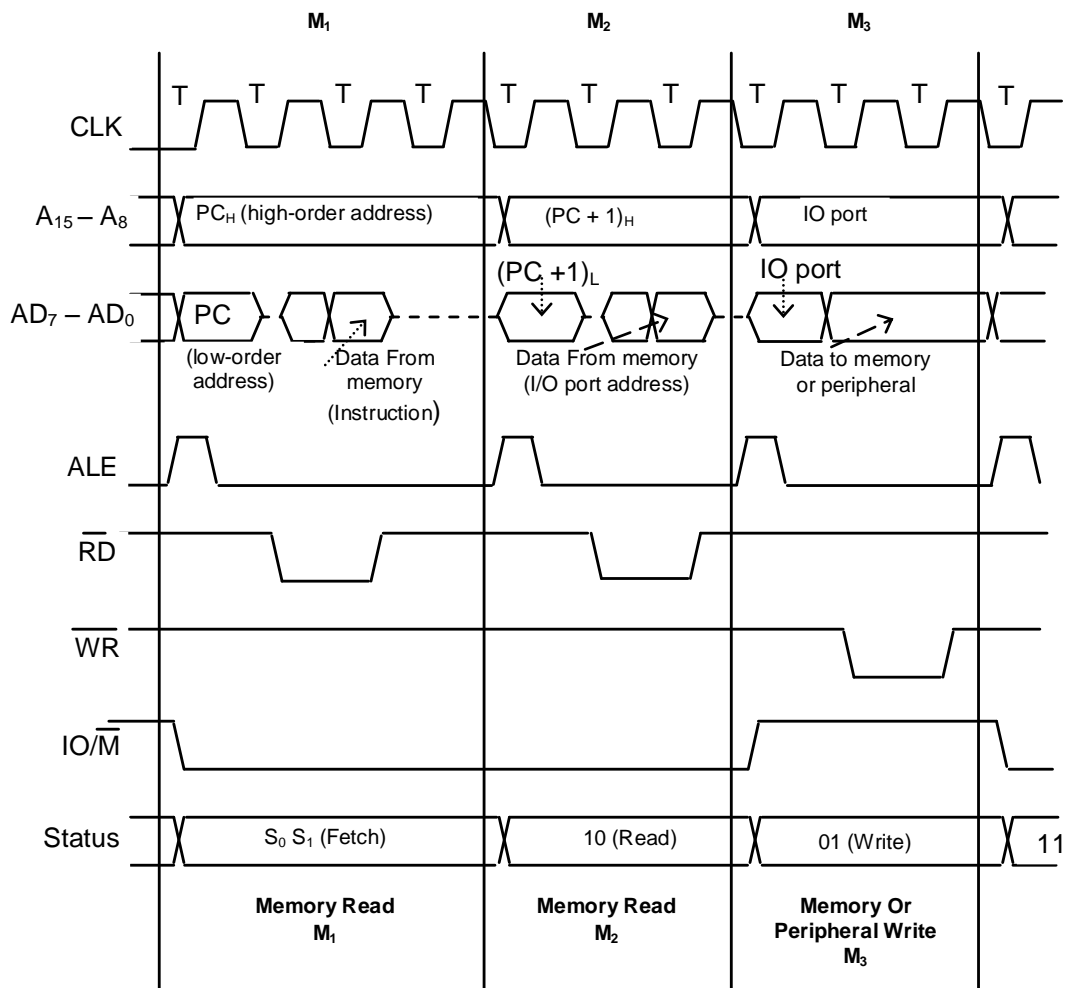
Hình II.6 Phần tử khuếch đại BUS hai chiều

II.1.4 Biểu đồ Timing thực hiện lệnh của CPU $\mu P8085$

Việc thực hiện một lệnh trong $\mu P8085$ thực tế là một chuỗi các thao tác READ và WRITE. Mỗi thao tác READ hay WRITE tương ứng với một chu kỳ máy M. Mỗi lệnh được thực hiện qua 1 đến 5 chu kỳ máy. Mỗi chu kỳ máy cần từ 3 đến 5 nhịp đồng hồ (hay còn gọi là trạng thái T)

Ở chu kỳ máy thứ nhất, CPU thực hiện việc nhận mã lệnh (Instruction Code Fetch), còn gọi là chu kỳ Opcode Fetch. Theo biểu đồ thời gian trên

hình II.8, thấy rằng việc thực hiện chu kỳ máy M_1 (chu kỳ nhận lệnh - Opcode Fetch), CPU gửi ra các tín hiệu $\overline{IO/\overline{M}}$, S_1 và S_0 (tương ứng 0, 1, 1 trên biểu đồ thời gian) xác định thao tác của chu kỳ.



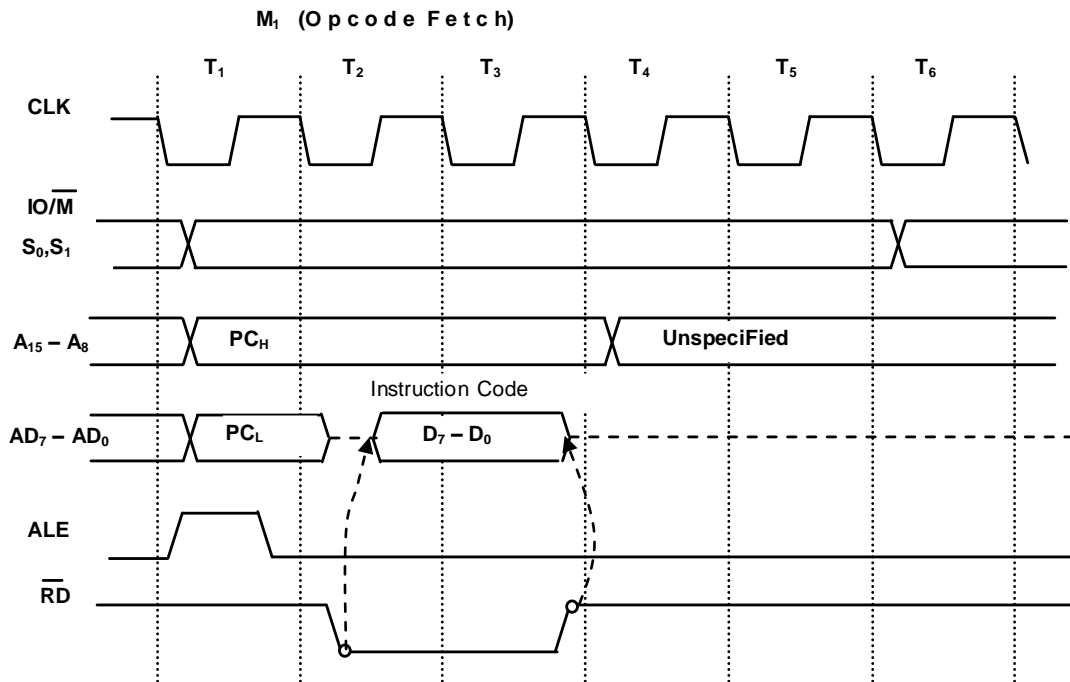
Hình II.7 Định thời cơ sở của $\mu P8085$ (Theo tài liệu của hãng Intel)

CPU cũng đồng thời gửi 16 bit địa chỉ ra ở chu kỳ máy đầu tiên, ngay từ nhịp đầu tiên (T_1) để xác định ô nhớ hay thiết bị I/O. Nội dung PC_L chỉ tồn tại trong thời gian 1 nhịp nên cần phải được chốt lại nhờ tín hiệu ALE ở mức cao.

Khi $D_7 - D_0$ đã ổn định trên các dây dữ liệu, CPU gửi tín hiệu \overline{RD} . Khi đã nhận được dữ liệu, \overline{RD} chuyển lên mức cao để cấm vị trí ô nhớ hay thiết bị I/O.

Số lượng chu kỳ máy và trạng thái cần cho thực hiện một lệnh là cố định, song số lượng này khác nhau đối với các lệnh khác nhau, tùy theo độ dài của từ lệnh (1 byte, 2 bytes, 3 bytes). Số lượng chu kỳ máy phụ thuộc vào số

lần CPU phải liên lạc với các phần tử khác trong hệ thống, chủ yếu là với các chip khác.



Hình II.8 Biểu đồ thời gian của các tín hiệu trong chu kỳ máy nhận lệnh (Opcode Fetch)

II.1.5 Khái niệm chu kỳ BUS

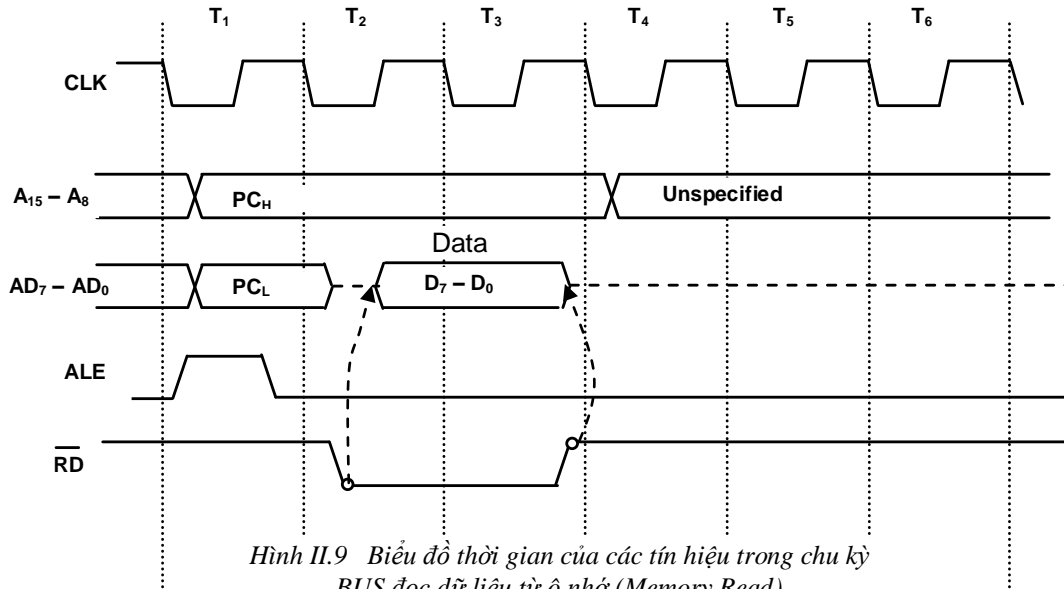
Khoảng thời gian (tính theo số lượng chu kỳ xung nhịp) để CPU (hoặc đơn vị làm chủ BUS) thực hiện hoàn thiện một thao tác di chuyển dữ liệu từ CPU đến bộ nhớ, đến thiết bị ngoại vi hoặc theo chiều ngược lại được gọi là *chu kỳ BUS*.

Một chu kỳ BUS được CPU hoặc đơn vị làm chủ BUS thực hiện trong hai giai đoạn:

- Giai đoạn một: CPU gửi địa chỉ vị trí cần truy xuất (ô nhớ hoặc thiết bị ngoại vi) lên BUS địa chỉ, khoảng thời gian này được gọi là *thời gian địa chỉ* (address time). Địa chỉ đích (destination - địa chỉ của một ô nhớ hay địa chỉ thanh ghi dữ liệu của thiết bị ngoại vi cần truy xuất) được CPU (hoặc đơn vị làm chủ BUS) gửi lên BUS địa chỉ cùng các tín hiệu xác định loại chu kỳ BUS
- Giai đoạn hai: CPU kiểm tra tín hiệu sẵn sàng (READY) của đơn vị cần truy xuất (bộ nhớ hoặc thiết bị ngoại vi) để thực hiện việc di chuyển và chốt dữ liệu. Khoảng thời gian này được gọi là *thời gian dữ liệu*.

Tồn tại 4 loại chu kỳ BUS cơ bản:

- Chu kỳ BUS đọc dữ liệu từ bộ nhớ (Memory Read)
- Chu kỳ BUS ghi dữ liệu vào bộ nhớ (Memory Write)
- Chu kỳ BUS đọc dữ liệu từ thiết bị ngoại vi (I/O Read)
- Chu kỳ BUS ghi dữ liệu vào thiết bị ngoại vi (I/O Write)



Hình II.9 Biểu đồ thời gian của các tín hiệu trong chu kỳ BUS đọc dữ liệu từ ô nhớ (Memory Read)

Ngoài ra, do sự khác nhau về vận tốc, khả năng xử lý và chuẩn bị, hoàn thiện dữ liệu, tín hiệu READY chưa ở mức tích cực, các thao tác di chuyển dữ liệu của CPU phải tạo thêm các trạng thái đợi (Wait State), do vậy các loại chu kỳ BUS có độ dài khác nhau.

II.1.6 Ngắt (Interrupt)

Trong thực tế, tốc độ xử lý dữ liệu của CPU cao hơn rất nhiều so với “sự chế biến dữ liệu” của các thiết bị I/O. Vì vậy cần tạo ra một cơ chế vào/ra hợp lý để tăng hiệu suất làm việc của CPU. Ngắt trong hệ thống Vi xử lý nhằm mục đích giải quyết sự bất hợp lý do CPU phải chờ đợi thiết bị ngoại vi. Thiết bị ngoại vi chỉ yêu cầu CPU phục vụ việc nhận hay chuyển giao dữ liệu khi bản thân nó đã sẵn sàng. Để thực hiện tốt yêu cầu này, cơ chế phục vụ ngắt là hợp lý nhất.

Ngắt nghĩa là yêu cầu CPU tạm thời dừng công việc hiện tại để trao đổi hay xử lý dữ liệu không thuộc tuần tự của chương trình đang được thực hiện. Ngắt là một hiện tượng xuất hiện ngẫu nhiên về phương diện thời điểm nhưng được dự đoán trước.

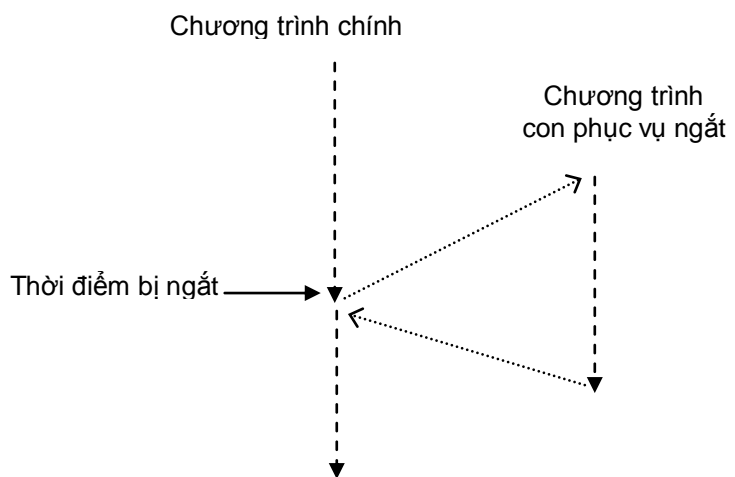
Ngắt là hiện tượng một tín hiệu xuất hiện báo với CPU rằng có một sự kiện đã xảy ra yêu cầu CPU phải xử lý. Quá trình xử lý của CPU sẽ bị tạm thời dừng lại để thực hiện một thao tác khác phục vụ sự

kiện có yêu cầu. Khi thao tác này kết thúc, quá trình xử lý vừa bị tạm dừng sẽ được tiếp tục. Bản thân sự kiện thông thường là yêu cầu phục vụ trao đổi dữ liệu của thiết bị ngoại vi đối với CPU.

Trong thực tế, ngắt được sử dụng chủ yếu khi các thiết bị ngoại vi (thường rất chậm so với tốc độ xử lý của CPU) cần trao đổi thông tin với CPU.

Khi cần trao đổi thông tin, thiết bị ngoại vi gửi tín hiệu *yêu cầu ngắt* (Interrupt Request) tới CPU. CPU sẽ thực hiện nốt lệnh hiện tại và trả lời bằng tín hiệu *nhận biết yêu cầu ngắt* (\overline{INTA}). Chương trình chính lúc này bị tạm dừng (ngắt) và CPU chuyển sang thực hiện *chương trình con phục vụ ngắt*, tức là chương trình con trao đổi thông tin với thiết bị ngoại vi yêu cầu ngắt. Sau khi xong công việc phục vụ ngắt, CPU quay về thực hiện tiếp chương trình chính kể từ lệnh tiếp theo sau khi bị ngắt.

Các tín hiệu yêu cầu phục vụ ngắt từ một thiết bị ngoại vi bất kỳ được gửi tới chân nhận yêu cầu ngắt của CPU có thể thông qua một khối điều khiển ngắt. Tùy theo người lập trình mà yêu cầu ngắt đó có được chuyển tới CPU hay không. Trong trường hợp yêu cầu ngắt được gửi tới CPU, xử lý của CPU gồm các bước sau:

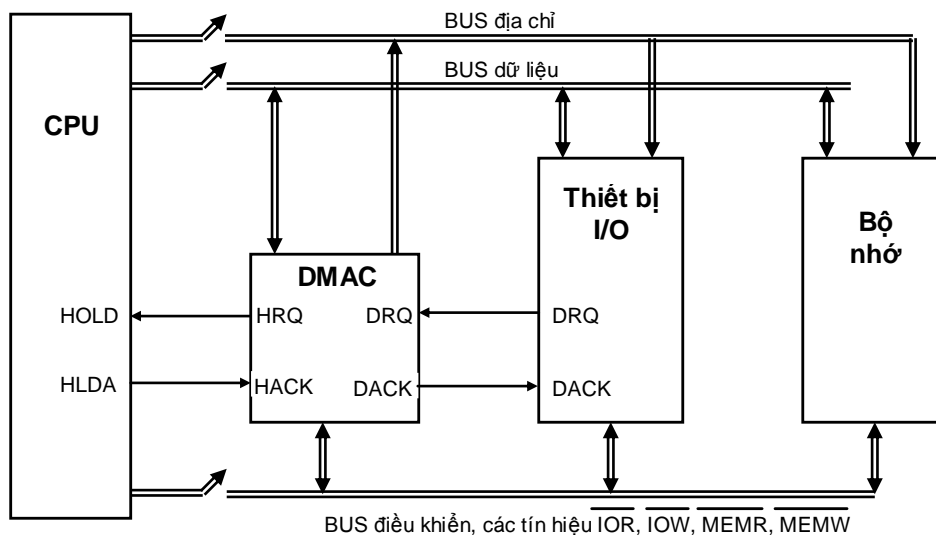


Hình II.10 Quá trình phục vụ ngắt

1. Thực hiện nốt lệnh đang được xử lý
2. Phát tín hiệu *nhận biết yêu cầu ngắt* gửi cho thiết bị yêu cầu phục vụ ngắt qua chân \overline{INTA}
3. Cất các cờ trạng thái hiện tại vào ngăn xếp
4. Xoá các cờ IF (*Interrupt Flag*) và cờ TF (*Trap Flag*)
5. Cất địa chỉ lệnh tiếp theo trong tuần tự chương trình đang thực hiện vào ngăn xếp
6. Lấy địa chỉ của chương trình con phục vụ ngắt trong bảng vector ngắt
7. Thực hiện chương trình con phục vụ ngắt.

II.1.7 Truy nhập trực tiếp bộ nhớ (Direct Memory Access – DMA)

Trong nhiều trường hợp, xảy ra hiện tượng phải chuyển một khối dữ liệu từ thiết bị ngoại vi vào một vùng nhớ hoặc ngược lại. Với phương pháp vào/ra bằng chương trình, dữ liệu nào cũng phải đi qua CPU, do vậy làm chậm tốc độ trao đổi dữ liệu. Để khắc phục tình trạng này, ta dùng phương pháp trao đổi dữ liệu giữa một vùng nhớ với thiết bị ngoại vi một cách trực tiếp không thông qua CPU, đó là phương pháp *truy nhập trực tiếp bộ nhớ* (DMA). Trong phương pháp này, CPU giao quyền điều khiển BUS dữ liệu cho một chip điện tử chuyên dụng gọi là chip DMAC (DMA Controller). Chip DMAC tự tạo ra địa chỉ, tạo các tín hiệu điều khiển việc ghi/đọc bộ nhớ, đếm số từ dữ liệu đã được ghi vào hoặc đọc từ bộ nhớ và sẽ thông báo cho CPU khi đã thực hiện xong việc trao đổi dữ liệu với bộ nhớ. Quá trình được thực hiện hoàn toàn bằng phần cứng, trực tiếp giữa thiết bị vào/ra và bộ nhớ nên tốc độ trao đổi thông tin tương đối nhanh. CPU không cần nhận lệnh, giải mã lệnh và thực hiện các lệnh di chuyển dữ liệu.



Hình II. 11 Mô tả các tín hiệu điều khiển trong quá trình DMA

Khi có yêu cầu trao đổi dữ liệu theo DMA, thiết bị ngoại vi gửi tín hiệu yêu cầu DRQ tới chip DMAC, chip này gửi tín hiệu yêu cầu treo HRQ tới chân HOLD của CPU. Nếu yêu cầu được chấp nhận, CPU sẽ gửi xung ghi nhận HLDA tới chân HACK của chip DMAC và tự treo các BUS, cho phép DMAC sử dụng BUS. DMAC gửi tín hiệu DACK tới thiết bị ngoại vi cho phép thiết bị này thực hiện việc trao đổi dữ liệu. Kết thúc quá trình trao đổi dữ liệu, chip DMAC chuyển trạng thái của tín hiệu HRQ về mức thấp để thông báo cho CPU.

II.1.8 Vi chương trình (MicroProgram) và tập lệnh của $\mu P8085$

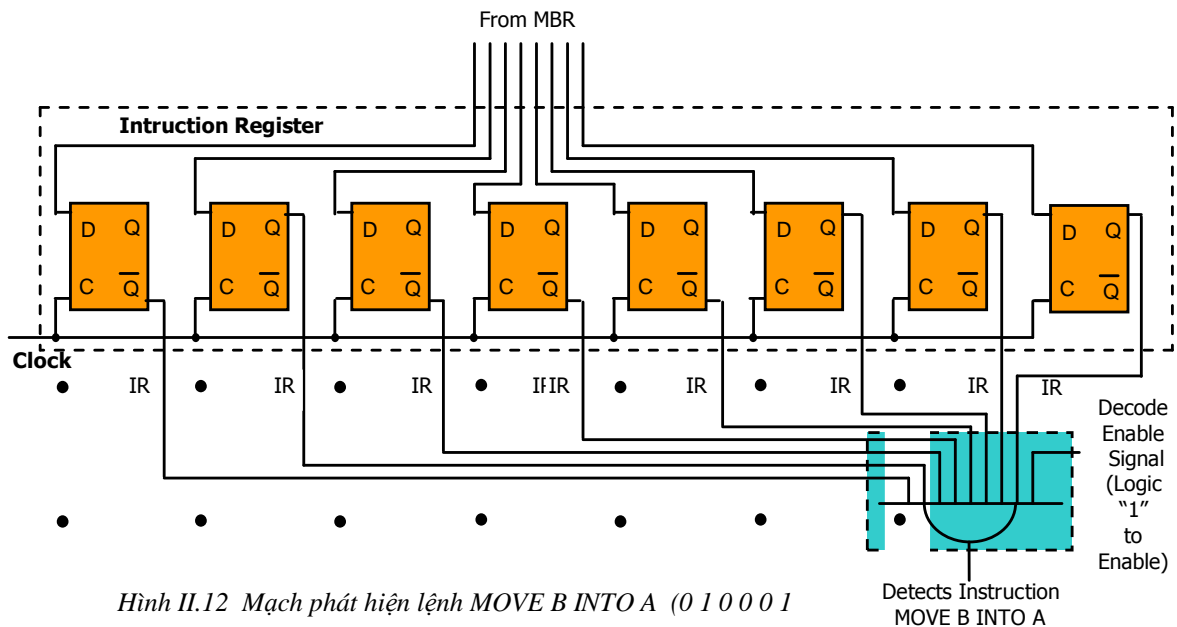
a) Đơn vị điều khiển CU – Control Unit

CU - Control Unit là đơn vị điều khiển, điều phối mọi hoạt động của các bộ phận chức năng trong CPU thông qua Control BUS. Có thể coi CU là khối dịch lệnh của CPU, nó tạo ra các tín hiệu tương ứng làm đầu vào cho Controller Unit để điều khiển hoạt động của các khối chức năng. Các tín hiệu do CU tạo ra có thể phân thành 2 loại: Tín hiệu định thời và tín hiệu điều hành hoạt động của CPU. Các tín hiệu định thời do CU tạo ra xác định trạng thái của CPU làm việc:

- Đang ở chế độ đọc dữ liệu vào (Input mode)
- Đang đưa dữ liệu ra (Output mode)
- Đang bắt đầu một hoạt tác khác (Beginning another operation).

Các tín hiệu trạng thái của CPU xác định CPU đang:

- Đọc dữ liệu từ bộ nhớ (Memory Read)
- Ghi dữ liệu vào bộ nhớ (Memory Write)
- Nhận lệnh (Instruction Fetch)



Hình II.12 Mạch phát hiện lệnh MOVE B INTO A (010001)

- Đọc dữ liệu từ thiết bị ngoại vi (I/O Read)
- Đưa dữ liệu ra thiết bị ngoại vi (I/O Write)

Cũng có thể có những thao tác không được nêu ở đây, nhưng chỉ các thao tác trên là quan trọng nhất.

Cần hiểu rằng mạch Controller Logic tạo các tín hiệu điều khiển dựa vào các tín hiệu trạng thái của CPU và tín hiệu định thời, có nghĩa là tạo tín hiệu gì và vào thời điểm nào.

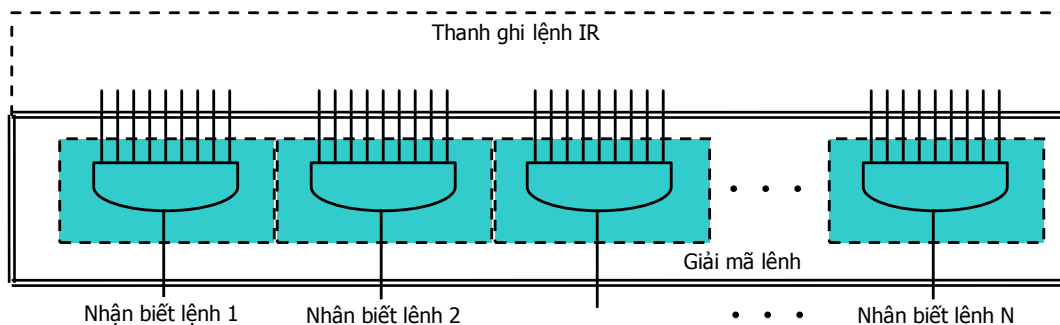
Để hiểu được kiến trúc khối CU, hãy tìm lời giải đáp cho câu hỏi: *Sau khi nhận lệnh, CPU làm sao “biết” phải thực hiện những thao tác nào để thực hiện lệnh?*

Tất cả các lệnh đều được biểu diễn dưới dạng mã nhị phân. Giả sử lệnh được biểu diễn bằng một mã 8 bits 01000111B (chuyển nội dung thanh ghi B sang thanh ghi A, ký hiệu là $[A] \leftarrow [B]$).

Trước hết, lệnh phải được *giải mã*. Một mạch AND có thể sử dụng để tạo ra tín hiệu nhận biết lệnh (Hình II.12). Đầu vào của mạch AND này được nối với các đầu ra của thanh ghi lệnh. Đầu ra của các phần tử trong thanh ghi lệnh xác nhận sự hiện diện của lệnh MOVE B TO A theo công thức

$$(\text{MOVE B TO A}) = \bar{IR}_7 \cdot IR_6 \cdot \bar{IR}_5 \cdot \bar{IR}_4 \cdot \bar{IR}_3 \cdot IR_2 \cdot IR_1 \cdot IR_0.$$

trong đó IR_n là đầu ra của các Flip-Flop tương ứng với các giá trị nhị phân của mã lệnh MOVE B TO A. Mạch AND nhận biết mã lệnh được gọi là mạch giải mã lệnh. Như vậy, nếu CPU sử dụng 8 bit để mã hoá các lệnh, có thể có 256 lệnh, và mạch giải mã lệnh cũng sẽ cần đến 256 mạch AND tương tự, tuy nhiên đầu vào của mỗi mạch là một tổ hợp duy nhất trong 256 tổ hợp có thể.



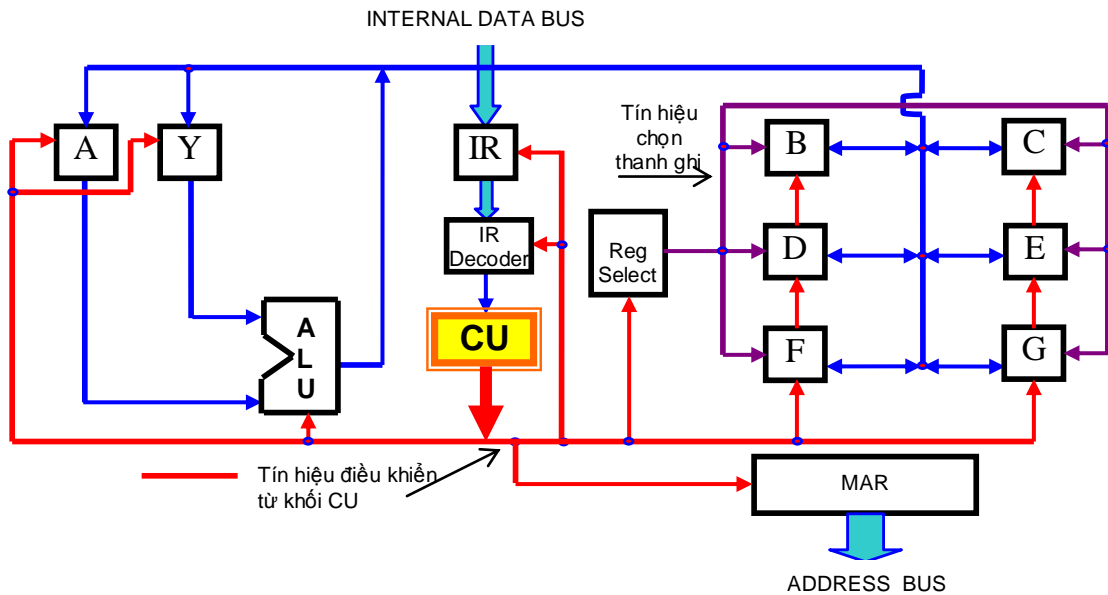
Hình II.13 Nhận biết các lệnh từ tổ hợp mã nhị phân

Để thực hiện lệnh, khối điều khiển CU xúc tiến mọi thao tác ngay bên trong CPU bằng cách tạo ra các tín hiệu điều khiển và các xung nhịp để định thời cho các khối chức năng thực hiện các thao tác.

Sau khi nhận tín hiệu từ khối giải mã lệnh (Instruction Decoder), CU sẽ tạo ra các tín hiệu điều khiển và các xung nhịp. Tín hiệu điều khiển sẽ cho phép (Enable) khối chọn thanh ghi (Reg Select) chọn thanh ghi B và thiết lập hệ thống đường truyền thông suốt giữa hai thanh ghi B và A. tiếp theo CU sẽ tạo các tín hiệu tương ứng để việc truyền dữ liệu giữa hai thanh ghi được thực hiện.

Tiếp theo, CU điều khiển thanh đếm chương trình PC tăng lên 1 để nhận tiếp lệnh từ bộ nhớ. Vì CU có nhiệm vụ giám sát và điều khiển mọi thao tác của các thành phần chức năng trong CPU, nên các dây điều khiển phải được nối trực tiếp từ CU tới mọi khối chức năng trong CPU như trên hình

II.13a. Cũng cần nhận thức rằng, lệnh được CPU lấy từ bộ nhớ. Trong thực tế, dữ liệu để xử lý cũng có thể xuất phát từ bộ nhớ, và các thanh ghi cũng có thể được chọn bất kỳ ngoại trừ thanh ghi lệnh IR và thanh đếm chương trình PC.



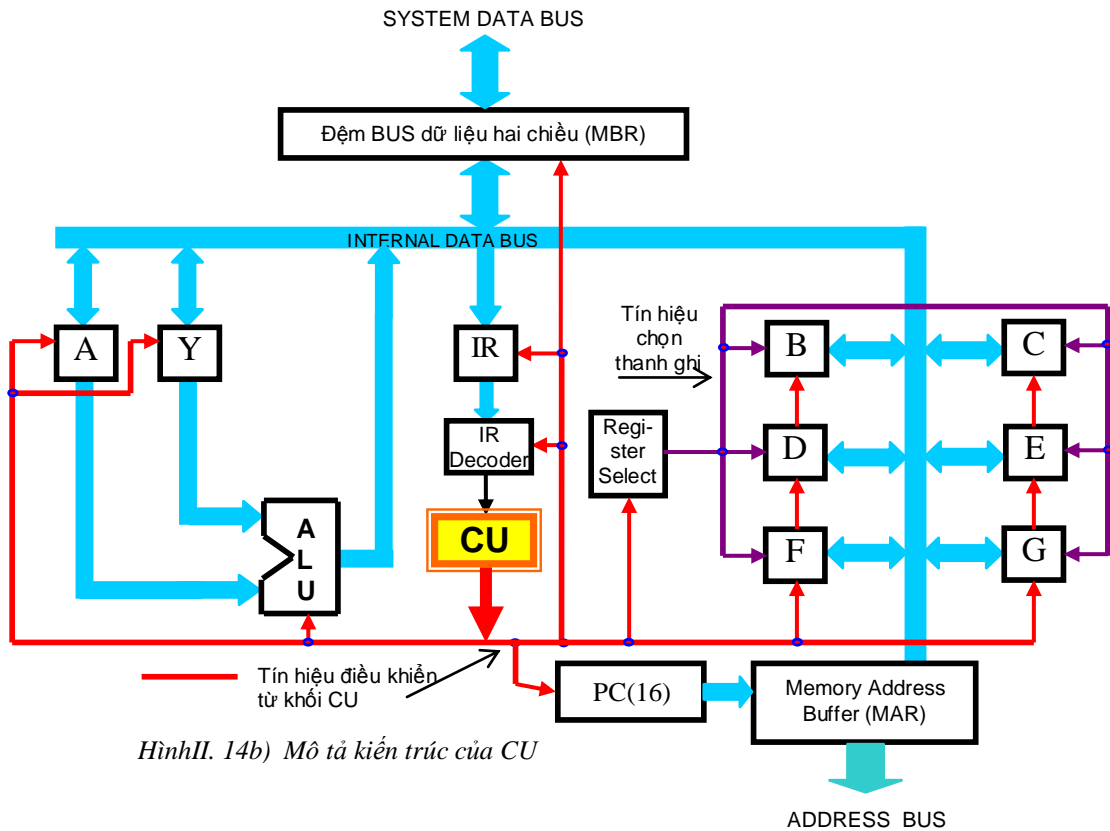
Hình II.14a) Mô tả kiến trúc của CU

Như vậy, lại cần thêm một thanh ghi liên lạc với BUS dữ liệu có nhiệm vụ truy nhập được vào bộ nhớ. Thanh ghi này làm trung gian giữa BUS dữ liệu bên ngoài và các thanh ghi đa năng khác, và nó được liên lạc với nhau thông qua BUS dữ liệu nội bộ (Internal Data BUS) - một BUS mà các thanh ghi được truy xuất trực tiếp. CU phải làm nhiệm vụ xác định thanh ghi nào được truy xuất qua BUS dữ liệu nội bộ tại thời điểm đó. Cũng vì BUS dữ liệu nội bộ của CPU truy xuất đến BUS dữ liệu hệ thống, nên cần phải có một cách thức để hoặc cách ly chúng khi cần thiết, hoặc cho phép ghép nối, nên cần thiết phải có thêm *thanh ghi đệm dữ liệu hai chiều*. Và như vậy, CU phải làm nhiệm vụ *điều khiển hướng di chuyển* của dữ liệu khi đi qua thanh ghi đệm (xem hình II.14b).

b) Vi chương trình

Giả thiết rằng lối ra của khối giải mã lệnh và tạo các tín hiệu điều khiển phải tạo ra 12 tín hiệu tại các cửa $G_1 - G_{12}$, 2 tín hiệu điều khiển bộ nhớ và 5 tín hiệu xung nhịp kích hoạt các thanh ghi PC (thanh đếm chương trình), MAR (thanh ghi đệm địa chỉ), MBR (thanh ghi đệm bộ nhớ), DO (thanh ghi dữ liệu) và IR (thanh ghi lệnh) để điều khiển quá trình nhận và thực hiện lệnh ADD. Các tín hiệu này được gửi tới để điều khiển hoạt động của các thành phần khác nhau trong CPU. Một chu trình thực hiện lệnh trên sẽ được thi hành.

Thực tế trong CPU của máy tính có từ 64 đến hơn 200 các tín hiệu điều khiển như thế. Sự khác nhau quan trọng giữa các lệnh và vi lệnh là ở chỗ vi lệnh có nhiều trường hơn. Tám bước trong bảng trên là một *vi chương trình* dịch một giai đoạn nhận lệnh (OPCODE FETCH) được thực thi sau lệnh cộng ADD. Như vậy một lệnh được dịch thành một chuỗi các vi lệnh, hay nói cách khác, mỗi mã lệnh có một vi chương trình.



Hình 11. 14b) Mô tả kiến trúc của CU

Số bước	Các tín hiệu điều khiển cửa												Điều khiển bộ nhớ		Các xung nhịp kích hoạt thanh ghi				
	G ₁	G ₂	G ₃	G ₄	G ₅	G ₆	G ₇	G ₈	G ₉	G ₁₀	G ₁₁	G ₁₂	EN	E/W	PC	MAR	MBR	DO	IR
1	1	0	0	0	0	0	0	0	0	0	0	0	0	x	0	1	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	x	1	0	0	0	0
3	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	1	0	0
4	0	0	0	0	0	0	1	0	0	0	0	0	0	x	0	0	0	0	1
5	0	0	1	0	0	0	0	0	0	0	0	0	0	x	0	1	0	0	0
6	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	1	0	0
7	0	0	0	0	0	0	1	0	0	1	0	0	0	x	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	1	0	x	0	0	0	1	0

Có thể thấy rằng, khối giải mã lệnh và tạo các tín hiệu điều khiển:

- + “Biết” phải thực hiện lệnh “như thế nào”, một khi lệnh từ IR (Instruction Register) được chuyển tới.

- + Giải quyết việc thực hiện một lệnh bằng cách điều khiển các khối chức năng liên quan thực hiện các phần việc.

Từ cách nhìn nhận trên, dễ dàng nhận ra rằng khối giải mã lệnh và tạo các tín hiệu điều khiển là bộ não thực thụ của CPU. Có thể coi khối này là một máy tính đặc dụng (*Special-purpose Computer*^(*)) bên trong CPU. Nó là hạt nhân cơ bản nhất dành riêng cho việc thực hiện một lệnh. Để thiết kế và xây dựng được khối này, cần phải có một “chương trình” (*program*^(*)) thật chi tiết. Chương trình dùng để xây dựng nên khối này cần phải có những thủ tục tuyệt đối chính xác nhằm mục đích thực hiện các lệnh.

Chương trình đó được gọi là *Vi chương trình* (MicroProgram) và được chế tạo như là một phần tích hợp cứng bên trong CPU, người lập trình không thể thay thế cũng như không thể truy nhập vào được.

Đối với các loại μP dạng *bit-slice microprocessor*, Vi chương trình hoàn toàn do người sử dụng xây dựng.

b) Tập lệnh của $\mu P8085$

Tập lệnh của $\mu P8085$ có thể chia thành nhiều nhóm lệnh nhỏ tùy theo từng cách tiếp cận. Theo phương thức xử lý và kết quả của việc xử lý dữ liệu, các lệnh trong tập lệnh được chia thành 4 nhóm chính:

1. Nhóm lệnh chuyển dữ liệu: các lệnh trong nhóm này thực hiện việc di chuyển dữ liệu giữa các thanh ghi với nhau, giữa thanh ghi với bộ nhớ và ngược lại, các lệnh vào/ra dữ liệu v.v...
2. Nhóm lệnh số học và logic: các lệnh trong nhóm này thực hiện các phép tính số học cơ bản là cộng và trừ 2 toán hạng, các lệnh tăng giảm, hay so sánh nội dung thanh ghi, các phép tính logic trong số sọc nhị phân, các phép dịch trái, phải dữ liệu trong thanh ghi, lệnh quay vòng trái phải v.v...
3. Nhóm lệnh điều khiển: Bao gồm các nhóm lệnh rẽ nhánh có điều kiện và không điều kiện, các lệnh gọi chương trình con
4. Nhóm lệnh đặc biệt: Nhóm lệnh đặc biệt bao gồm các lệnh lấy bù 1 của số liệu trong nội dung thanh ghi, lệnh thiết lập và xoá các cờ, lệnh hiệu chỉnh thập phân một số liệu Hexa và lệnh vào/ra dữ liệu nối tiếp.

(*) Thuật ngữ do hãng Intel sử dụng trong tài liệu gốc

II.1.9 Vài nét về lập trình cho 8085

Phát triển phần mềm (lập trình) và các kỹ thuật liên quan đóng vai trò quan trọng bậc nhất trong các ứng dụng từ đơn giản đến phức tạp của các hệ Vi xử lý và máy vi tính. Đối với các hệ Vi xử lý, mọi ứng dụng đều được phát triển nhờ vào một “công cụ” phát triển phần mềm hoàn chỉnh: Lập trình hợp ngữ.

Quá trình phát triển một chương trình (phần mềm ứng dụng) cho một hệ Vi xử lý, kể từ khi xác định nhiệm vụ cần thực hiện cho đến khi chương trình được cài đặt hoàn chỉnh vào hệ thống có thể chia ra năm bước cơ bản sau đây:

a) **Đặt vấn đề (xác nhận vấn đề):** Trước khi giải quyết vấn đề, người lập trình cần xác định xem, liệu vấn đề có thể được giải quyết nhờ một chương trình trong một hệ Vi xử lý hay không. Phải thấy rằng không phải hệ Vi xử lý “vạn năng” đến mức có thể giải quyết tất cả mọi vấn đề nảy sinh trong thực tiễn, thậm chí đôi khi còn làm cho sự việc càng thêm phức tạp.

b) **Xác định phương pháp giải quyết vấn đề:** Đây chính là bước tìm thuật giải (Algorithm) tối ưu cho vấn đề được đặt ra. Người lập trình phải tìm và lựa chọn được từ nhiều giải pháp một giải pháp tốt nhất, nhưng kinh tế nhất để thực hiện. Không chỉ tìm giải thuật tốt nhất mà còn phải tìm ngôn ngữ lập trình phù hợp nhất để giải quyết vấn đề.

c) **Thực hiện giải pháp:** Phương pháp giải quyết vấn đề thường được xác nhận qua từng bước theo một lưu đồ. Lưu đồ là cách thể hiện tường minh các bước thực hiện chương trình trong hệ thống, đồng thời nó giúp người lập trình định hướng tốt khi viết chương trình.

d) **Viết chương trình :** Bản thân lưu đồ đã cho thấy rõ giải pháp giải quyết vấn đề theo quan điểm lập trình. Việc chuyển từ lưu đồ sang ngôn ngữ chương trình là bước dễ dàng hơn rất nhiều so với cách viết chương trình không có lưu đồ. Đây chỉ là bước cụ thể hóa lưu đồ nhờ tuân tự thực hiện các lệnh, và là bước thực tế hóa giải pháp thực hiện vấn đề.

e) **Kiểm tra và gỡ rối:** Sau khi cài đặt việc kiểm tra tính chính xác là vô cùng quan trọng. Những sai sót phải được phát hiện và hiệu chỉnh, đôi khi là từ chính thuật giải. Việc gỡ rối chương trình tức là thực hiện từng bước chương trình, phát hiện các sai sót ẩn, hiệu chỉnh các sai sót này.

Để thực hiện được tất cả các bước trên người lập trình phải có kỹ thuật lập trình hoàn thiện để thiết kế chương trình, phải có các công cụ lập trình tốt.

II.1.10 Hệ lệnh của $\mu P8085$

Các lệnh của $\mu P8085$ được thống kê trong bảng II.1

Mnemonic	Instruction Code									Mô tả nhiệm vụ
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		
MOVE, LOAD, AND STORE										
MOV r1,r2	0	1	D	D	D	S	S	S		Move Register To Register
MOV M,r	0	1	1	1	0	S	S	S		Move Register To Memory
MOV r,M	0	1	D	D	D	1	1	0		Move Memory To Register
MVI r	0	0	D	D	D	1	1	0		Move Immediate Register
MVI M	0	0	1	1	0	1	1	0		Move Immediate Memory
LXI B	0	0	0	0	0	0	0	1		Load Immediate Register Pair B & C
LXI D	0	0	0	1	0	0	0	1		Load Immediate Register Pair D & E
LXI H	0	0	1	0	0	0	0	1		Load Immediate Register Pair H & L
STAX B	0	0	0	1	0	0	1	0		Store A indirect
STAX D	0	0	0	0	1	0	1	0		Store A indirect
LDAX B	0	0	0	1	0	0	1	0		Load A indirect
LDAX D	0	0	0	1	1	0	1	0		Load A indirect
STA	0	0	1	1	0	0	1	0		Store A direct
LDA	0	0	1	1	1	0	1	0		Load A direct
SHLD	0	0	1	0	0	0	1	0		Store H & L direct
LHLD	0	0	1	0	1	0	1	0		Load H & L direct
XCHG	1	1	1	0	1	0	1	1		Exchange D & E, H & L registers
STACK OPS										
PUSH B	1	1	0	0	0	1	0	1		Push Register Pair B & C on stack
PUSH D	1	1	0	1	0	1	0	1		Push Register Pair D & E on stack
PUSH H	1	1	1	0	0	1	0	1		Push Register Pair H & L on stack
PUSH PSW	1	1	1	1	0	1	0	1		Push A ang Flags on stack
POP B	1	1	0	0	0	0	0	1		Pop Register Pair B & C oFF stack
POP D	1	1	0	1	0	0	0	1		Pop Register Pair D & E oFF stack
POP H	1	1	1	0	0	0	0	1		Pop Register Pair H & L oFF stack
POP PSW	1	1	1	1	0	0	0	1		Pop A ang Flags oFF stack
XTHL	1	1	1	0	0	0	1	1		Exchange register pair H & L, top oF
SPHL	1	1	1	1	1	0	0	1		H & L to stack pointer
LXI SP	0	0	1	1	0	0	0	1		Load immediate stack pointer
INX SP	0	0	1	1	0	0	1	1		Increment stack pointer
DCX SP	0	0	1	1	1	0	1	1		Decrement stack pointer
JUMP										
JMP	1	1	0	0	0	0	1	1		Jump unconditional
JC	1	1	0	1	1	0	1	0		Jump on carry
JNC	1	1	0	1	0	0	1	0		Jump on no carry
JZ	1	1	0	0	1	0	1	0		Jump on zero
JNZ	1	1	0	0	0	0	1	0		Jump on no zero
JP	1	1	1	1	0	0	1	0		Jump on positive
JM	1	1	1	1	1	0	1	0		Jump on minus
JPE	1	1	1	0	1	0	1	0		Jump on parity even
JPO	1	1	1	0	0	0	1	0		Jump on parity odd
PCHL	1	1	1	0	1	0	0	1		H & L to program counter
CALL										
CALL	1	1	0	0	1	1	0	1		Call unconditional
CC	1	1	0	1	1	1	0	0		Call on carry
CNC	1	1	0	1	0	1	0	0		Call on no carry
CZ	1	1	0	0	1	1	0	0		Call on zero
CNZ	1	1	0	0	0	1	0	0		Call on no zero
CP	1	1	1	1	0	1	0	0		Call on positive
CM	1	1	1	1	1	1	0	0		Call on minus
CPE	1	1	1	0	1	1	0	0		Call on parity even
CPO	1	1	1	0	0	1	0	0		Call on patity odd
RETURN										
RET	1	1	0	0	1	0	0	1		Return

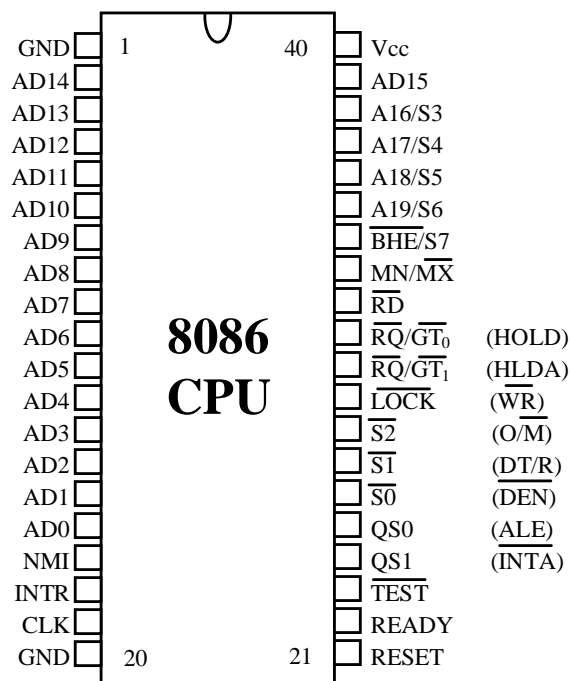
RC	1	1	0	1	1	0	0	0	Return on carry
RNC	1	1	0	1	0	0	0	0	Return on no carry
RZ	1	1	0	0	1	0	0	0	Return on zero
RNZ	1	1	0	0	0	0	0	0	Return on no zero
RP	1	1	1	1	0	0	0	0	Return on positive
RM	1	1	1	1	1	0	0	0	Return on minus
RPE	1	1	1	0	1	0	0	0	Return on parity even
RPO	1	1	1	0	0	0	0	0	Return on parity odd
RESTART									
RST	1	1	A	A	A	1	1	1	Restart
INPUT/OUTPUT									
IN	1	1	0	1	1	0	1	1	Input
OUT	1	1	0	1	0	0	1	1	Output
RIM	0	0	1	0	0	0	0	0	Read interrupt mask
SIM	0	0	1	1	0	0	0	0	Set interrupt mask
INCREMENT AND DECREMENT									
INR r	0	0	D	D	D	1	0	1	Increment register
DCR r	0	0	D	D	D	1	0	1	Decrement register
INR M	0	0	1	1	0	1	0	0	Increment memory
DCR M	0	0	1	1	0	1	0	1	Decrement memory
INX B	0	0	0	0	0	0	1	1	Increment B&C registers
INX D	0	0	0	1	0	0	1	1	Increment D&E registers
INX H	0	0	1	0	0	0	1	1	Increment H&L registers
DCX B	0	0	0	0	1	0	1	1	Decrement B&C registers
DCX D	0	0	0	1	1	0	1	1	Decrement D&E registers
DCX H	0	0	1	0	1	0	1	1	Decrement H&L registers
ADD									
ADD r	1	0	0	0	0	S	S	S	Add register to A
ADC r	1	0	0	0	1	S	S	S	Add register to A with carry
ADD M	1	0	0	0	0	1	1	0	Add memory to A
ADC M	1	0	0	0	1	1	1	0	Add memory to A with carry
ADI	1	1	0	0	0	1	1	0	Add immediate to A
ACI	1	1	0	0	1	1	1	0	Add immediate to A with carry
DAD B	0	0	0	0	1	0	0	1	Add B&C to H&L
DAD D	0	0	0	1	1	0	0	1	Add D&E to H&L
DAD H	0	0	1	0	1	0	0	1	Add H&L to H&L
DAD SP	0	0	1	1	1	0	0	1	Add SP to H&L
SUBTRACT									
SUB r	1	0	0	1	0	S	S	S	Subtract register From A
SBB r	1	0	0	1	1	S	S	S	Subtract register From A with borrow
SUB M	1	0	0	1	0	1	1	0	Subtract memory From A
SBB M	1	0	0	1	1	1	1	0	Subtract memory From A with borrow
SUI	1	1	0	1	0	1	1	0	Subtract immediate From A
SBI	1	1	0	1	1	1	1	0	Subtract immediate From A with borrow
LOGICAL									
ANA r	1	0	1	0	0	S	S	S	And register with A
XRA r	1	0	1	0	1	S	S	S	Exclusive OR register with A
ORA r	1	0	1	1	0	S	S	S	OR register with A
CMP r	1	0	1	1	1	S	S	S	Compare register with A
ANA M	1	0	1	0	0	1	1	0	And memory with A
XRA M	1	0	1	0	1	1	1	0	Exclusive memory with A
ORA M	1	0	1	1	0	1	1	0	OR memory with A
CMP M	1	0	1	1	1	1	1	0	Compare memory with A
ANI	1	1	1	0	0	1	1	0	And immediate with A
XRI	1	1	1	0	1	1	1	0	Exclusive OR immediate with A
ORI	1	1	1	1	0	1	1	0	OR immediate with A
CPI	1	1	1	1	1	1	1	0	Compare immediate with A

ROTATE								
RLC	0	0	0	0	1	1	1	Rotate A left
RRC	0	0	0	0	1	1	1	Rotate A right
RAL	0	0	0	1	0	1	1	Rotate A left through carry
RAR	0	0	0	1	1	1	1	Rotate A right through carry
SPECIALS								
CMA	0	0	1	0	1	1	1	Complement A
STC	0	0	1	1	0	1	1	Set carry
CMC	0	0	1	1	1	1	1	Complement carry
DAA	0	0	1	0	0	1	1	Decimal adjust A
CONTROL								
EI	1	1	1	1	0	1	1	Enable interrupt
DI	1	1	1	1	0	0	1	Disable interrupt
NOP	0	0	0	0	0	0	0	No-operation
HLT	0	1	1	1	1	1	0	Halt

II.2 Các trung tâm Vi xử lý họ 80x86

II.2.1 Mô tả chân của μ P8086 và các tín hiệu

μ P8086 được chế tạo theo công nghệ HMOS, đóng vỏ CerDIP 40 chân. Là loại Vi xử lý có khả năng xử lý trực tiếp dữ liệu 8 hoặc 16 bit. Về tập lệnh, μ P8086 hoàn toàn tương thích với tập lệnh của iAPX86/10 và về phần cứng, hoàn toàn tương thích với các mạch ngoại vi của các trung tâm 8080/8085 của Intel.



Hình II. 15. Sơ đồ nối chân trung tâm Vi xử lý 8086

μ P8086 có thể hoạt động ở một trong hai chế độ:

Chế độ MIN: CPU tự tạo ra các tín hiệu điều khiển hoạt động của BUS (các chân từ 24 đến 34)

- **Chế độ MAX:** CPU chỉ đưa ra các tín hiệu trạng thái, cần thêm một chip điều khiển BUS (BUS controller 8288) và chip này sẽ thông

dịch các tín hiệu trạng thái thành các tín hiệu điều khiển BUS tương thích với cấu trúc MULTIBUSTM, cách này đảm bảo hoạt động đọc số liệu ổn định hơn.

Hình II. 11 là sơ đồ nối chân của μP8086

- + AD15 – AD0: BUS dồn kênh dữ liệu và địa chỉ 16 bits
- + A19 – A16 / S6 – S3: 4 bits địa chỉ cao hoặc 4 tín hiệu trạng thái hoạt động hiện tại của CPU

S4	S3	Thanh ghi được truy xuất
0	0	ES
0	1	SS
1	0	CS
1	1	DS

S5 chỉ trạng thái chờ ngắt
S6 luôn luôn bằng 0
- + $\overline{\text{BHE}}/\text{S7}$: Tín hiệu này kết hợp với chân địa chỉ A0 cho chỉ thị các trạng thái sau:

BHE	A0	
0	0	Một từ đã được truyền qua D15 – D0
0	1	Một Byte trên D15 – D8 được truy xuất tới một địa chỉ Byte lẻ
1	0	Một Byte trên D7 – D0 được truy xuất tới một địa chỉ Byte chẵn
1	1	chưa xác định
- + $\overline{\text{RD}}$: Nếu bằng “1” đang đọc bộ nhớ (hoặc thiết bị vào/ra)
Nếu bằng “0” đang ghi ra bộ nhớ (hoặc thiết bị vào/ra)
- + READY: nếu bộ nhớ (hoặc thiết bị vào/ra) cần truy nhập hoàn tất việc chuyển dữ liệu đến (hoặc đi) chúng cần phát ra tín hiệu READY ở mức “1” tới chân CPU, chỉ khi ấy CPU mới đọc số liệu vào hoặc đưa dữ liệu ra
- + INTR: CPU kiểm tra trạng thái chân này sau khi thực hiện xong mỗi lệnh để xét xem có yêu cầu ngắt từ phần cứng đến hay không, nếu ở mức “1”, CPU sẽ chuyển sang phục vụ ngắt. Thao tác kiểm tra này có thể “che” được nhờ dùng mặt nạ che ngắt
- + $\overline{\text{TEST}}$: Lối vào này của CPU luôn luôn được kiểm tra trong lệnh WAIT. Nếu bằng “0” CPU tiếp tục thực hiện chương trình, nếu bằng “1”, CPU chạy các chu trình giả cho tới khi $\overline{\text{TEST}} = “0”$.

- + NMI: Chân ngắt theo sườn lên của xung, không che được.
- + RESET: Chân nhận tín hiệu tái khởi động hệ thống. Nếu có sự thay đổi từ “0” lên “1” và tồn tại tối thiểu trong 4 nhịp đồng hồ thì hệ thống sẽ tự khởi động lại.
- + CLK: Lối vào của xung nhịp đồng hồ
- + Vcc: Nguồn nuôi +5V
- + GND: Chân nối đất (0V)
- + MN/ \overline{MX} : Khi được nối với Vcc, $\mu P8086$ hoạt động ở chế độ MIN, nếu nối với GND, hoạt động ở chế độ MAX
- + $\overline{S2}$, $\overline{S1}$, $\overline{S0}$: Ở chế độ MAX, chip điều khiển BUS sử dụng 3 tín hiệu trạng thái này để phát ra các tín hiệu điều khiển truy xuất bộ nhớ và thiết bị vào/ra. Tổ hợp có ý nghĩa như sau

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	
0	0	0	yêu cầu ngắt cứng qua chân INTR được chấp nhận
0	0	1	đọc thiết bị vào/ra
0	1	0	Ghi thiết bị vào/ra
0	1	1	CPU bị treo
1	0	0	nạp mã chương trình vào hàng nhận lệnh
1	0	1	đọc bộ nhớ
1	1	0	ghi vào bộ nhớ
1	1	1	trạng thái thụ động
- + $\overline{RQ/GT_0}$, $\overline{RQ/GT_1}$: Tín hiệu phục vụ việc chuyển mạch BUS cục bộ (Local BUS) giữa các đơn vị làm chủ BUS (BUS master). BUS cục bộ là BUS giữa các đơn vị xử lý (không phải là BUS nối với các thiết bị ngoại vi). Đơn vị làm chủ BUS là $\mu P8086$ hoặc một chip điều khiển nào đó (ví dụ DMAC) hiện đang nắm quyền điều khiển BUS cục bộ.
- + \overline{LOCK} : nếu bằng “0”, đơn vị làm chủ BUS không nhượng quyền làm chủ BUS cục bộ
- + QS_1 , QS_0 chỉ thị trạng thái của hàng nhận lệnh trước PQ

0	0	không hoạt động
0	1	byte 1 của mã toán trong PQ được xử lý
1	0	hàng đợi lệnh được xoá
1	1	byte 2 của mã toán trong PQ được xử lý

II.2.2 Cấu trúc Trung tâm Vi xử lý họ 80x86

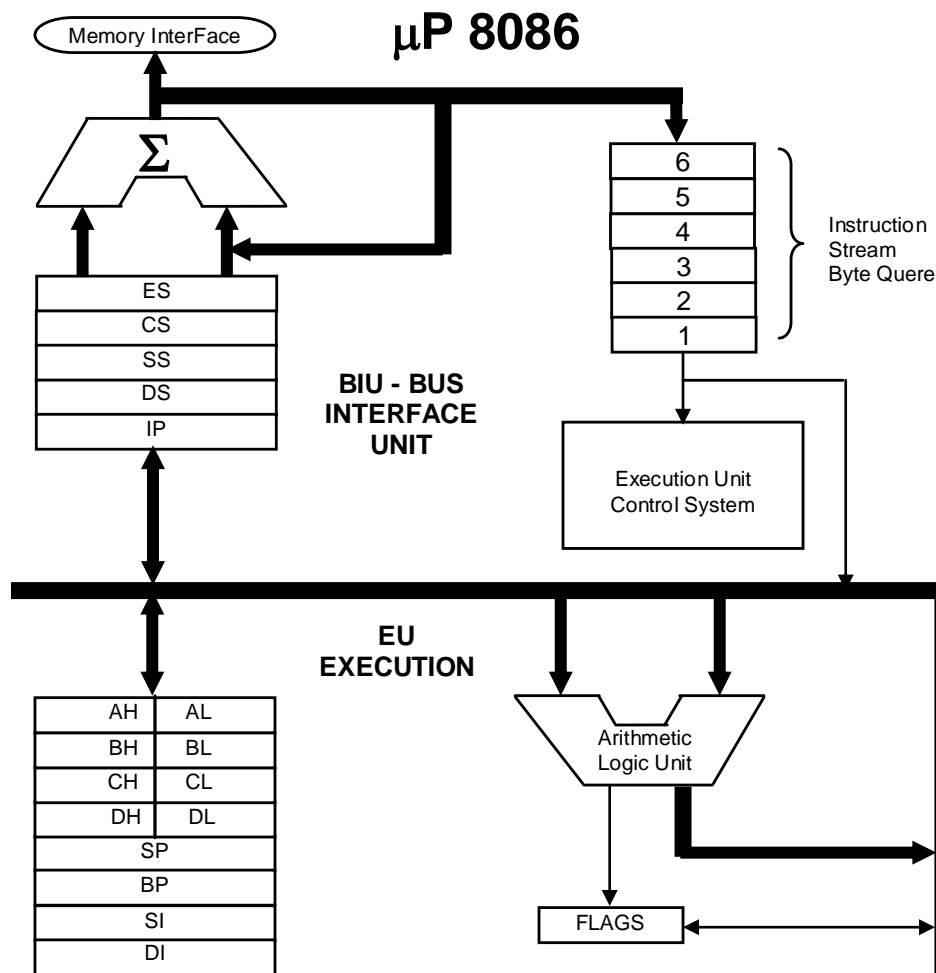
Các μP họ 80x86 được phát triển trên cơ sở công nghệ VLSI với các phần tử cơ bản là các transistor trường MOS có độ tiêu hao công suất rất nhỏ. Sơ đồ khối chức năng của $\mu P8086$ được thể hiện trên hình II.15, gồm hai thành phần chủ yếu là *đơn vị ghép nối BUS* (BIU – BUS Interface Unit), *đơn vị thực hiện lệnh* (EU – Execute Unit). Tất cả các thanh ghi và đường truyền dữ liệu trong EU đều có độ dài 16 bits. BIU thực hiện tất cả các nhiệm vụ giao tiếp với BUS cho EU: thiết lập khâu liên kết với BUS dữ liệu, BUS địa chỉ và BUS điều khiển. Dữ liệu được trao đổi giữa CPU với bộ nhớ khi EU có yêu cầu, song không được truyền trực tiếp tới EU mà thông qua một vùng nhớ RAM dung lượng nhỏ (6 bytes) được gọi là *hàng nhận lệnh trước* (Instruction Stream Byte Queue PQ - PreFetch Queue) rồi mới được truyền cho hệ thống *điều khiển EU* (Execution Unit Control System).

Khi EU đang thực hiện một lệnh thì BIU đã tìm và lấy lệnh sau đặt sẵn vào PQ. Đây là *cơ chế đường ống* (pipeline), một kỹ thuật tăng tốc độ cho CPU.

Kỹ thuật đường ống sử dụng một vùng nhớ RAM cực nhanh, làm tăng đáng kể tốc độ của bộ Vi xử lý thông qua việc truy tìm lệnh từ bộ nhớ chương trình thay cho sự liên hệ giữa CPU với bộ nhớ chương trình. Riêng với bộ xử lý Pentium, có hai đường ống, một cho các lệnh và một cho các dữ liệu.

Bảng sau cho ta vài thông số kỹ thuật cơ bản của các trung tâm Vi xử lý họ 80x86

Loại μP	Độ dài thanh ghi	Độ rộng BUS địa chỉ	Độ rộng BUS dữ liệu	Không gian nhớ	Tần số cực đại
8088	16 bits	20 bits	8 bits	1MByte	10 MHz
8086	16 bits	20 bits	16 bits	1Mbyte	10 MHz
80188	16 bits	20 bits	8 bits	1Mbyte	10 MHz
80186	16 bits	20 bits	16 bits	1Mbyte	10 MHz
80286	16 bits	24 bits	16 bits	16Mbytes	16 MHz
80386SX	32 bits	24 bits	16 bits	16Mbytes	20MHz
80386DX	32 bits	32 bits	32 bits	4Gbytes	40 MHz
i486	32 bits	32 bits	32 bits	4Gbytes	66 MHz
i486SX	32 bits	32 bits	32 bits	4Gbytes	25 MHz
Pentium (phiên bản đầu)	32 bits	32 bits	64 bits	4Gbytes	66 MHz

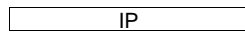
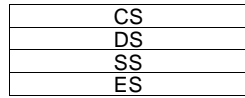
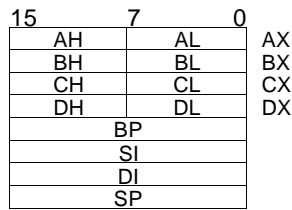
Hình II. 16 Cấu trúc các khối chức năng $\mu P8086$

II.2.3 Hệ thống thanh ghi trong các $\mu P80x86$

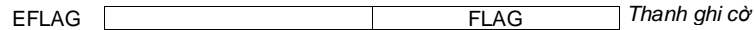
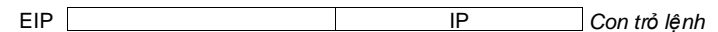
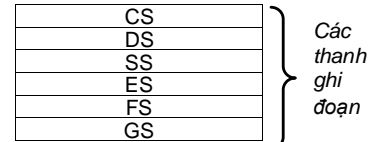
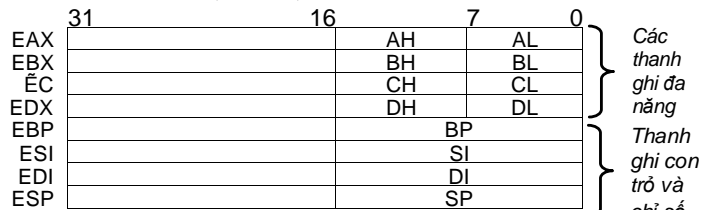
Có thể coi các thanh ghi của các trung tâm Vi xử lý như một bộ nhớ được đặt ngay bên trong CPU, có tốc độ truy cập cực kỳ nhanh, được dùng để lưu giữ các dữ liệu và các kết quả tạm thời của các quá trình tính toán, xử lý. Các thanh ghi trong họ $\mu P80x86$ có độ dài khác nhau, 16 bits với các trung tâm 8088/86, 80188/86 và 80286, 32 bits với các trung tâm 80386/486 trở đi và được mô tả trên Hình II.13

EU của $\mu P8086$ có 8 thanh ghi đa năng với tên gọi là AH, AL, BH, BL, CH, CL, DH, DL. Những thanh ghi này có thể sử dụng riêng rẽ cho việc lưu giữ các dữ liệu nhị phân 8 bits. Cũng có thể sử dụng chúng thành từng cặp thanh ghi có tên gọi là AX (AH-AL), BX (BH-BL), CX (CH-CL), và DX (DH-DL) để lưu giữ các dữ liệu nhị phân 16 bits.

8086 → 80286



i386, i486, Pentium



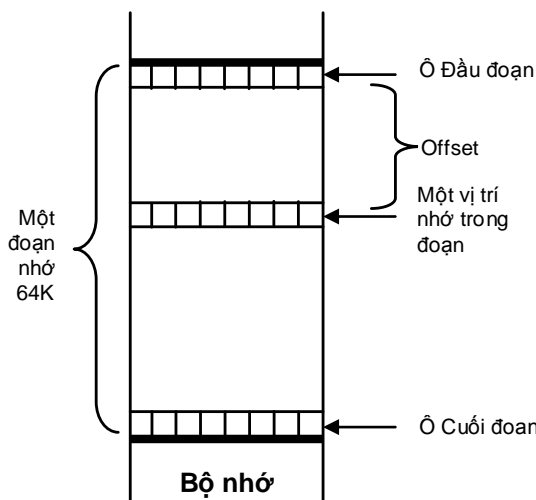
Hình II.17 Các thanh ghi trong các trung tâm Vi xử lý họ 80x86

1. Các thanh ghi đa năng:

Ưu điểm của việc sử dụng các thanh ghi này để lưu giữ tạm thời các dữ liệu là tốc độ truy cập của CPU với chúng nhanh hơn rất nhiều so với việc sử dụng các ô nhớ.

2. Các thanh ghi đoạn:

CPU đưa ra BUS địa chỉ 20 bits để quản lý một không gian nhớ 1Mbyte (1.048.576 Bytes) bộ nhớ vật lý. Tuy nhiên, các thanh ghi trong CPU lại chỉ có độ dài 16 bits, do vậy, không gian nhớ được chia thành từng đoạn (segment), mỗi đoạn dài 64Kbytes, địa chỉ của Byte đầu tiên được lấy làm địa chỉ đoạn. Hai đoạn nhớ kề cận cách nhau tối thiểu là 16 Bytes. Mỗi Byte nhớ trong đoạn sẽ được xác định bởi độ lệch (offset), tức là khoảng cách tính từ Byte nhớ đó đến đầu đoạn.



Hình II. 18 Về khái niệm địa chỉ đoạn và địa chỉ offset

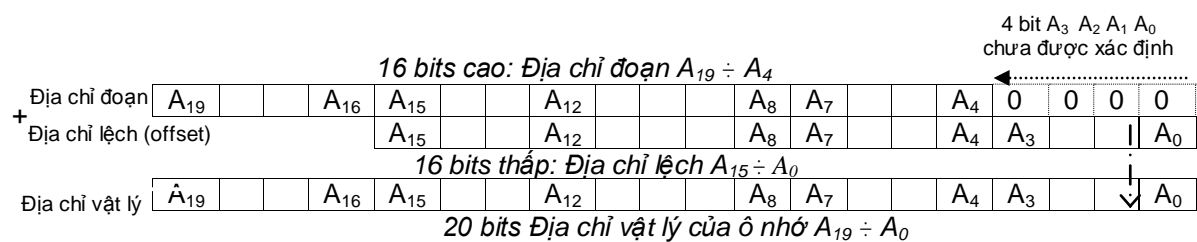
Như vậy, mỗi một cặp thông số bao gồm địa chỉ đoạn và độ lệch (segment : offset) sẽ xác định địa chỉ logic của một Byte nhớ vật lý trong bộ nhớ. Thanh ghi đoạn (Segment Register) chứa 16 bits cao, thanh ghi độ lệch (dùng thanh ghi đa năng hoặc các thanh ghi chỉ số, con trỏ) chứa 16 bit thấp của 20 bits địa chỉ. Địa chỉ vật lý của một vị trí nhớ do vậy sẽ được BIU tính theo công thức:

$$\text{Địa chỉ vật lý} = (\text{Segment}) \times 10H + (\text{offset})$$

µP8086 sử dụng 4 thanh ghi đoạn riêng biệt là: *Thanh ghi đoạn mã lệnh CS* (Code Segment), *thanh ghi đoạn ngăn xếp SS* (Stack Segment), *thanh ghi đoạn mở rộng ES* (Extra Segment) và *thanh ghi đoạn dữ liệu DS* (Data Segment).

- *Thanh ghi đoạn mã lệnh CS* là thanh ghi chứa địa chỉ bắt đầu của đoạn chương trình hiện hành trong bộ nhớ
- *Thanh ghi đoạn dữ liệu DS* là thanh ghi địa chỉ bắt đầu của đoạn chứa số liệu hiện hành trong bộ nhớ, hay còn gọi là nơi chứa các biến của chương trình
- *Thanh ghi đoạn ngăn xếp SS* là thanh ghi địa chỉ bắt đầu của đoạn ngăn xếp (Stack) trong bộ nhớ (ô nhớ do thanh ghi này chỉ đến còn được gọi là *đáy ngăn xếp*), nơi lưu giữ địa chỉ và dữ liệu khi thực hiện các chương trình con, lệnh gọi chương trình con hoặc thủ tục
- *Thanh ghi đoạn mở rộng ES* là thanh ghi địa chỉ bắt đầu của đoạn chứa các dữ liệu chuỗi, xâu ký tự
- Ngoài ra, trong các trung tâm i386/i486 còn có hai thanh ghi đoạn FS và GS.

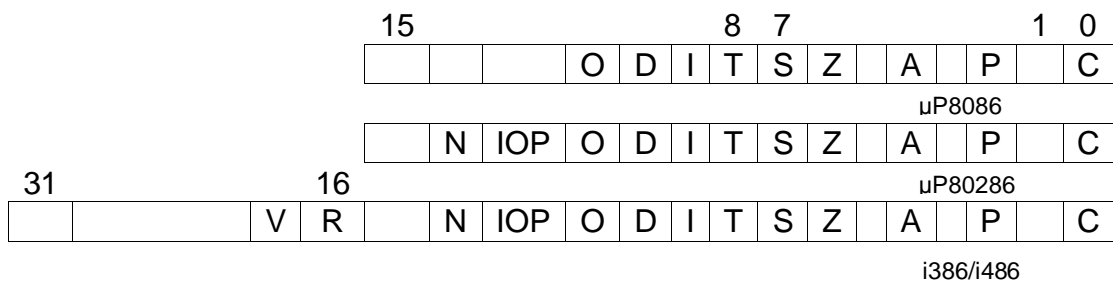
Các đoạn trong bộ nhớ có thể tách biệt nhau, nhưng cũng có thể gối chồng lên nhau, nhưng bao giờ cũng cách nhau tối thiểu 16 Bytes. Độ lệch 16Bytes này thực chất do 4 bit thấp nhất của địa chỉ từ A₃ đến A₀ chưa được xác định. Khi bộ cộng trong đơn vị địa chỉ tính địa chỉ vật lý để đưa ra BUS địa chỉ, nó lấy nội dung thanh ghi đoạn chèn thêm 4 số 0000_B cho 4 bits thấp nhất của 20 bits địa chỉ rồi mới cộng với 16 bits của phần địa chỉ offset. Điều này lý giải công thức tương trưng đã nêu trên. Phần địa chỉ bắt đầu của đoạn được lưu giữ trong thanh ghi đoạn cũng thường được gọi là *địa chỉ cơ sở* hay *địa chỉ nền*.



Hình II. 19 Mô tả cách tính địa chỉ vật lý của một vị trí nhớ

3. Thanh ghi cờ FLAG:

Chỉ có 9 trong số 16 bits của thanh ghi cờ (trong các bộ vi xử lý μ P8086 - μ P80286) và 11 trong số 32 bits của thanh ghi cờ (trong các bộ xử lý i386/i486) được sử dụng. Mỗi cờ có thể được lập (= "1") hay xoá (= "0") để biểu thị trạng thái kết quả của một phép xử lý trước đó hoặc trạng thái hiện tại của CPU. Các cờ IOP, N, R và V liên quan đến chế độ bảo vệ trong các bộ xử lý 80286 và i386/i486. Chín cờ còn lại gồm 6 cờ chỉ trạng thái và 3 cờ điều khiển.



Hình II.20 Vị trí các cờ trong thanh ghi cờ của họ Vi xử lý 80x86

Các cờ trạng thái gồm:

- Cờ nhớ CF (Carry Flag) được lập nếu một thao tác xảy ra hiện tượng *carry* hoặc *borrow* đối với toán hạng đích. CF có thể lập bởi lệnh STC và xoá bởi lệnh CLC.
- Cờ chẵn lẻ PF (Parity Flag) được lập nếu kết quả của một phép xử lý có số bit bằng "1" là số chẵn.
- Cờ mang phụ AF (Auxiliary Flag) được dùng cho xử lý các mã BCD và được lập nếu thao tác xử lý gây hiện tượng carry hoặc borrow cho 4 bits thấp của toán hạng
- Cờ zero ZF (Zero Flag) được lập nếu kết quả xử lý số liệu có kết quả bằng 0
- Cờ dấu SF (Sign Flag) dấu tương ứng với MSB của kết quả phép toán, được lập với kết quả dương và xoá với kết quả âm
- Cờ tràn OF (Overflow Flag) nếu kết quả phép toán là quá lớn cho toán hạng đích.

Các cờ điều khiển gồm:

- Cờ hướng DF (Direction Flag) xác định hướng của phép toán xử lý chuỗi ký tự, nếu được lập, chuỗi sẽ được xử lý từ địa chỉ cao tới địa chỉ thấp và ngược lại. Cờ được lập bởi lệnh STD và xoá bằng lệnh CLD

- *Cờ ngắt* IF (Interrupt enable Flag) nếu được lập, CPU sẽ chấp nhận yêu cầu ngắt cứng và phục vụ ngắt. Được lập bởi lệnh STI và xoá bằng lệnh CLI
- *Cờ bẫy* TF (Trap Flag) Dùng trong gỡ rối chương trình (Debugger) Không thể lập hay xoá trực tiếp bởi lệnh của máy.

4. Thanh ghi con trỏ lệnh IP

Thanh ghi con trỏ lệnh IP (Instruction Pointer) – thanh ghi 16 bits dùng để lưu giữ phần offset của địa chỉ lệnh kế tiếp sẽ được thực hiện trong tuần tự thực hiện chương trình. Kết hợp với CS, IP giống như thanh đếm chương trình PC trong $\mu P8085$, mỗi lần từ lệnh được đọc ra từ bộ nhớ, BIU sẽ thay đổi giá trị IP tùy theo độ dài của từ lệnh (số bytes của từ lệnh) sao cho nó chỉ đến từ lệnh kế tiếp trong bộ nhớ chương trình. Cũng cần nói thêm rằng khi gặp các lệnh rẽ nhánh hoặc lệnh gọi chương trình con, gọi thủ tục ..., các giá trị của CS:IP sẽ thay đổi đột ngột không theo quy luật trên. Các giá trị mới của CS:IP do người lập trình cung cấp thông qua địa chỉ của các nhãn (Label) trong chương trình hoặc giá trị cụ thể.

5. Các thanh ghi dữ liệu

Có 4 thanh ghi dữ liệu:

- *Thanh ghi tích lũy* AX (Accumulator register) thường dùng để lưu giữ các kết quả xử lý
- *Thanh ghi cơ sở* BX (Base register) thường dùng chỉ địa chỉ cơ sở (đáy) của một vùng nhớ trong bộ nhớ
- *Thanh ghi đếm* CX (Counter register) thường dùng để khai báo số lần một thao tác nào đó phải được thực hiện trong các vòng lặp, phép dịch, phép quay..., Giá trị của nội dung thanh ghi CX sẽ giảm đi một sau mỗi thao tác
- *Thanh ghi số liệu* DX (Data register) thường dùng để lưu giữ số liệu dùng làm thông số chuyển giao cho một chương trình. DX là thanh ghi duy nhất được dùng để chứa địa chỉ của các thiết bị vào/ra

6. Các thanh ghi con trỏ và chỉ số

Có 2 thanh ghi con trỏ và 2 thanh ghi chỉ số:

- *Thanh ghi con trỏ ngăn xếp* SP (Stack Pointer) chứa địa chỉ đỉnh ngăn xếp (vùng nhớ đặc biệt, hoạt động theo nguyên tắc LIFO –

Last In First Out – vào sau ra trước) sử dụng cho việc lưu giữ tạm thời các dữ liệu hay địa chỉ khi gọi chương trình con, khi phục vụ ngắt v.v... giá trị nội dung của SP luôn luôn là phần offset của địa chỉ ngăn xếp kế tiếp

- Thanh ghi con trỏ cơ sở BP (Base Pointer) có chức năng chứa giá trị offset tính từ địa chỉ SS nhưng còn được sử dụng để truy cập dữ liệu bên trong ngăn xếp
- Các thanh ghi chỉ số nguồn DI và thanh ghi chỉ số đích SI (Destination Index và Source Index) được dùng để lưu giữ các thành phần offset đối với những vùng dữ liệu được cất trong đoạn dữ liệu. Hai nội dung của hai thanh ghi này liên kết với nội dung thanh ghi đoạn DS để tạo ra địa chỉ nguồn và địa chỉ đích của vùng nhớ.

II.2.4 Các chế độ làm việc MIN/MAX

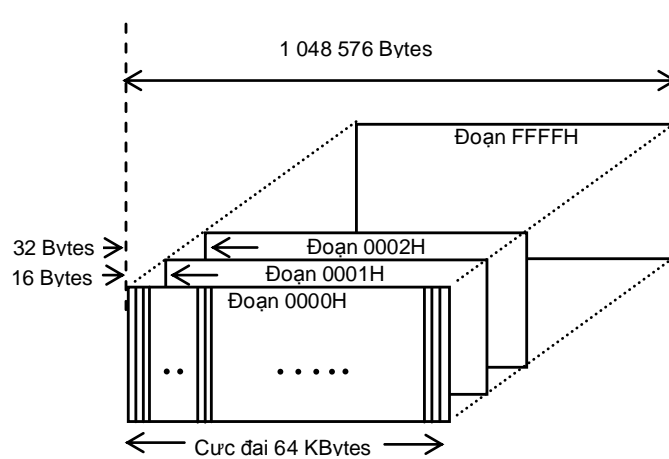
μ P8086 có hai chế độ làm việc: chế độ MIN và chế độ MAX. Chân số 33 của μ P8086 được coi như là *chân bẫy* (trap pin) cho μ P8086 trong việc định nghĩa chế độ làm việc. Những mạch phụ trợ cần thiết cho hai chế độ làm việc không thể thoả mãn với hệ thống 40 chân của CPU loại này, vì vậy một số chân sẽ đảm nhiệm những chức năng khác khi được xác định cho một chế độ, phụ thuộc vào cách nối chân MN/ \overline{MX} . Khi được nối với GND (mức điện áp 0V), μ P8086 chuyển đổi các chân từ 24 đến 31 sang chế độ MAX. Một mạch phụ điều khiển BUS 8288 sẽ giải mã các tín hiệu trạng thái $\overline{S_0}$, $\overline{S_1}$, $\overline{S_2}$ để tạo ra các tín hiệu định thời và các tín hiệu điều khiển tương thích với cấu trúc MULTIBUSTM trong các hệ thống máy tính. Khi được nối lên mức điện áp nguồn nuôi (mức Vcc +5V) tự μ P8086 tạo các tín hiệu điều khiển BUS trên các chân từ 24 đến 31 như được ghi trong ngoặc ở Hình II. 14.

II.2.5 Phương thức quản lý bộ nhớ, các mode địa chỉ

a. Phương thức quản lý bộ nhớ:

BUS địa chỉ của μ P8086 có độ dài 20 bits, do vậy có thể quản lý được $2^{20} = 1\text{M}$ ô nhớ (Mỗi tổ hợp “0” hoặc “1” của các bit trong 20 bits địa chỉ xác định vị trí của một ô nhớ). Vì một ô nhớ trong hệ Vi xử lý là 1 Byte, nên nói cách khác, *không gian nhớ* mà μ P8086 quản lý được là 1Mbyte.

Các thanh ghi của μ P8086 chỉ có độ dài 16 bits, nên nếu dùng một thanh ghi để đánh địa chỉ thì chỉ quản lý được 2^{16} ô nhớ, tức là 64KB. Để giải quyết vấn đề quản lý 1MByte, tức là 1.048.576 Bytes, μ P8086 sử dụng BUS địa chỉ có độ rộng 20 bits thông qua nội dung của hai thanh ghi 16 bits để đánh địa chỉ của bộ nhớ theo phương thức sau:

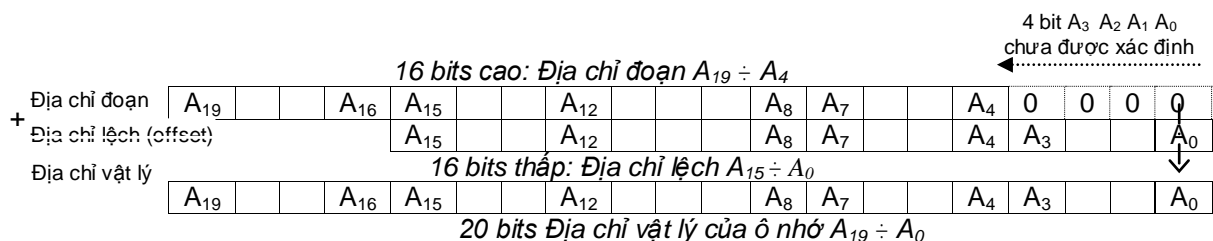


Hình II.21 Cách chia đoạn nhớ trong $\mu P8086$

Bằng cách lập chương trình, không gian địa chỉ được chia thành các *đoạn* (segment) nhớ với kích thước cố định là 64Kbytes gọi là một *đơn vị logic* của bộ nhớ. Mỗi đoạn gồm các ô nhớ liên tiếp, độc lập và được định vị tách rời nhau. Mỗi đoạn được người lập trình gán cho một *địa chỉ đoạn*, là địa chỉ ô nhớ đầu tiên của đoạn đó, còn được gọi là *địa chỉ nền*. Giá trị của các địa chỉ đoạn liền kề cách nhau tối thiểu là 16 Bytes. Các đoạn có thể kế cận, tách rời, phủ lấp nhau. Bên trong đoạn sẽ sử dụng các *giá trị lệch* (offset), tức là khoảng cách từ địa chỉ đoạn đến ô nhớ nằm trong đoạn. Một cặp giá trị địa chỉ đoạn và giá trị lệch, $[segment]:[offset]$, được gọi là *địa chỉ logic*. Địa chỉ logic cho phép định vị chính xác một Byte nhớ trong không gian địa chỉ. Địa chỉ đoạn được chứa trong các thanh ghi đoạn, giá trị dịch chuyển được chứa trong các thanh ghi đa năng, con trỏ hoặc chỉ số.

Về bản chất, thanh ghi đoạn chứa 16 bits cao của 20 bits địa chỉ, giá trị dịch chuyển là 16 bit thấp, và sự lệch nhau 4 bits đã được đơn vị địa chỉ của BIU giải quyết như trình bày trong hình II. 19: Dịch trái thanh ghi đoạn 4 bits (tương đương phép nhân với 16, cộng với giá trị dịch chuyển offset trong thanh ghi đa năng để tính địa chỉ vật lý của ô nhớ. Công thức tương ứng phép “dịch trái và cộng” có thể trình bày như sau:

$$\text{Địa chỉ vật lý} = 10H \times (\text{segment}) + (\text{offset})$$



Hình II.19 Mô tả cách tính địa chỉ vật lý của một vị trí nhớ

Thanh ghi đoạn là một thanh ghi 16 bits, có nhiệm vụ xác định đoạn của ô nhớ, còn thanh ghi đa năng cũng là một thanh ghi 16 bits. Vậy thanh ghi đoạn có thể định được $2^{16} = 65.536$ đơn vị (64K) đoạn nhớ và mỗi đoạn có 64Kbytes. Vậy Vi xử lý $\mu P8086$ có thể định địa chỉ tới $64K \times 64Kbytes = 4Gbytes$ nhớ.

Thanh ghi đoạn mã CS xác định đoạn nhớ chương trình mà lệnh kế tiếp sẽ được lấy để thực hiện, thanh ghi con trỏ IP chứa địa chỉ offset của lệnh kế tiếp. Cặp CS:IP tạo nên địa chỉ logic của lệnh kế tiếp trong tuần tự thực hiện chương trình. Các từ lệnh của họ 80x86 có thể có độ dài từ 1 byte đến tối đa là 15 bytes. Khi lệnh được thực hiện, giá trị của con trỏ IP do vậy sẽ tăng lên đúng bằng số Bytes của từ lệnh. Cần nhớ rằng *nội dung của thanh ghi con trỏ lệnh IP cùng với nội dung thanh ghi đoạn CS xác định địa chỉ của ô nhớ lệnh tiếp theo trong tuần tự thực hiện chương trình.*

b. Các mode đánh địa chỉ

1. Định vị thanh ghi (register addressing): Toán hạng được truy xuất nằm ngay trong thanh ghi của CPU.

Thí dụ MOV AX,BX ; chuyển nội dung của toán hạng nguồn (nội dung của thanh ghi) BX vào toán hạng đích AX. Nội dung thanh ghi BX vẫn được giữ nguyên.

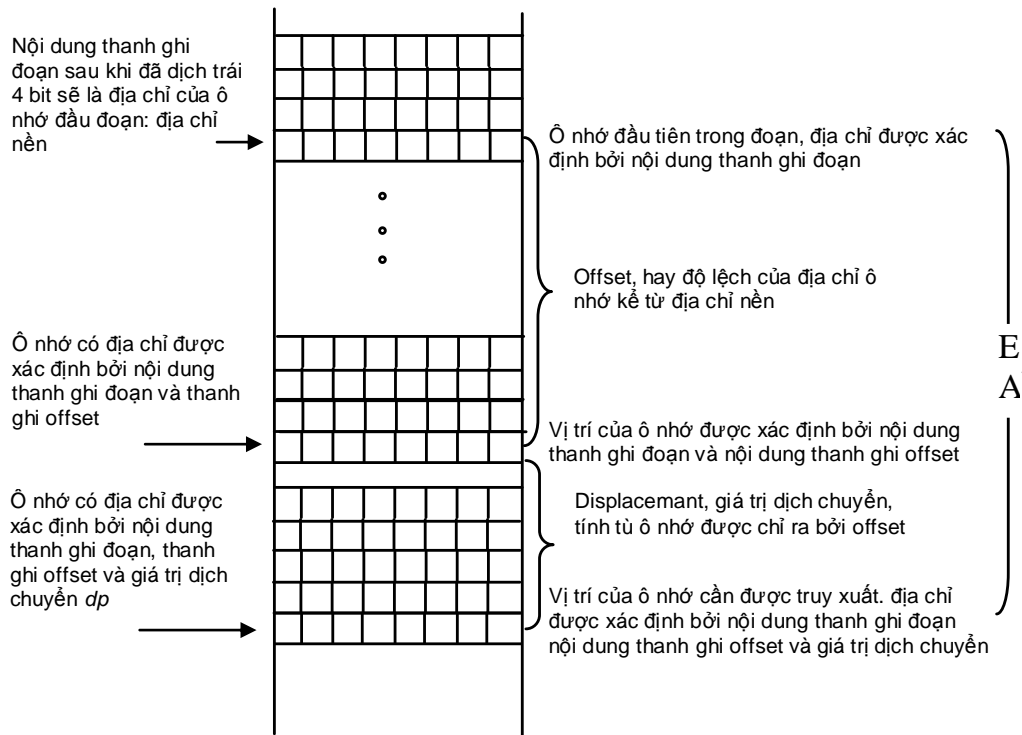
2. Định vị tức thời (immediate addressing): Toán hạng tức thời là dữ liệu 8 hay 16 bits nằm ngay trong lệnh, có thể dùng làm toán hạng nguồn hay hằng số. Toán hạng tức thời được lưu giữ ngay trong đoạn mã của bộ nhớ, ngay sau mã lệnh, nó được lấy ra cùng với lệnh và ghi vào hàng đợi lệnh PQ, do vậy được truy xuất nhanh hơn so với truy xuất toán hạng từ bộ nhớ.

Thí dụ MOV AL, 12H ; nạp giá trị 12H vào thanh ghi AL

3. Các kiểu định vị ô nhớ

Khác với hai kiểu định vị trên, toán hạng trong đoạn nhớ dữ liệu được CPU *truy xuất qua BUS dữ liệu*. Biết rằng, địa chỉ vật lý của ô nhớ được tính từ nội dung thanh ghi đoạn và offset theo cách trình bày trong Hình II. 19. Giá trị lệch mà đơn vị thực hiện lệnh EU tính cho một toán hạng trong đoạn nhớ kể từ địa chỉ đầu đoạn được gọi là *địa chỉ hiệu dụng* EA (effective address) của toán hạng. Đơn vị thực hiện lệnh có thể tính EA dựa vào cách mô tả địa chỉ trong phần toán hạng nguồn của lệnh. Ngoài giá trị trực tiếp, hoặc nội dung thanh ghi cơ sở hay thanh ghi chỉ số, khi cần còn có thể có một giá trị số

có độ dài 8 bits hay 16 bits được cộng thêm vào gọi là *giá trị dịch chuyển dp* (displacement). Xem Hình II.22



Hình II. 22 Mô tả cách xác định địa chỉ vật lý của ô nhớ cần truy xuất

Cụ thể như sau:

– *Định vị trực tiếp* (direct addressing): Toán hạng chứa địa chỉ là một số nằm ngay trong lệnh. Địa chỉ đoạn hiện tại nằm trong thanh ghi đoạn DS

Thí dụ `MOV CX,[1435H]` ; chuyển nội dung ô nhớ có địa chỉ offset bằng 1435H trong đoạn dữ liệu hiện tại vào thanh ghi CX

– *Định vị gián tiếp thanh ghi* (register indirect): địa chỉ hiệu dụng EA là nội dung của một trong các thanh ghi BX, BP, SI hoặc DI

Thí dụ `MOV AX, [SI]` ; chuyển nội dung của ô nhớ trong đoạn dữ liệu hiện tại có địa chỉ offset là nội dung thanh ghi SI

– *Định vị cơ sở* (based addressing): EA là tổng của nội dung thanh ghi BX hoặc BP và giá trị dịch chuyển dp nếu có

Thí dụ `MOV [BX] + dp, AL` ; chuyển nội dung thanh ghi AL và ô nhớ có địa chỉ offset bằng tổng của nội dung thanh ghi BX và giá trị dịch chuyển dp

– *Định vị chỉ số (indexed addressing)*: EA là tổng của nội dung thanh ghi SI hoặc DI và giá trị dịch chuyển *dp* nếu có

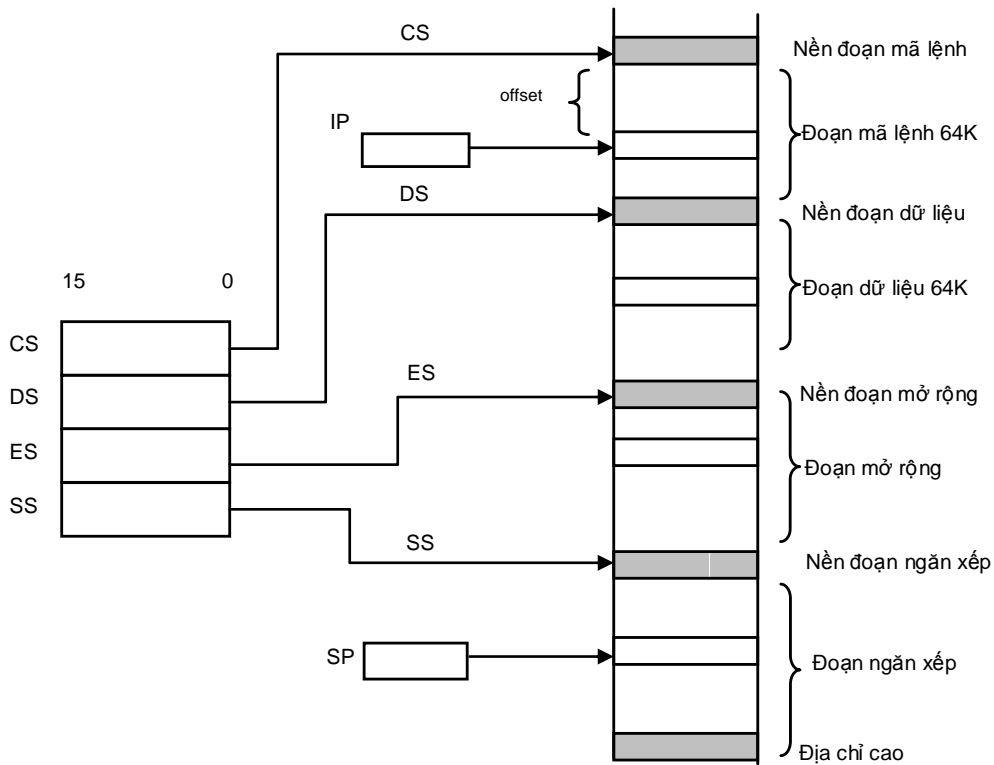
Thí dụ MOV AL,[SI] + dp ; chuyển nội dung ô nhớ có địa chỉ offset bằng tổng của nội dung thanh ghi SI và giá trị dịch chuyển dp vào thanh ghi AL

– *Định vị chỉ số và cơ sở (indexed addressing)*: EA là tổng của nội dung các thanh ghi cơ sở, thanh ghi chỉ số và giá trị dịch chuyển *dp* nếu có

Thí dụ MOV AH,[BX][SI] + dp ; chuyển nội dung ô nhớ có địa chỉ offset bằng tổng của nội dung thanh ghi BX, thanh ghi SI và giá trị dịch chuyển dp vào thanh ghi AH

– *Định vị chuỗi (string addressing)*: dùng riêng cho xử lý chuỗi. CPU sẽ tự động sử dụng các thanh ghi chỉ số nguồn SI và thanh ghi chỉ số đích DI để chỉ đến các byte kế tiếp

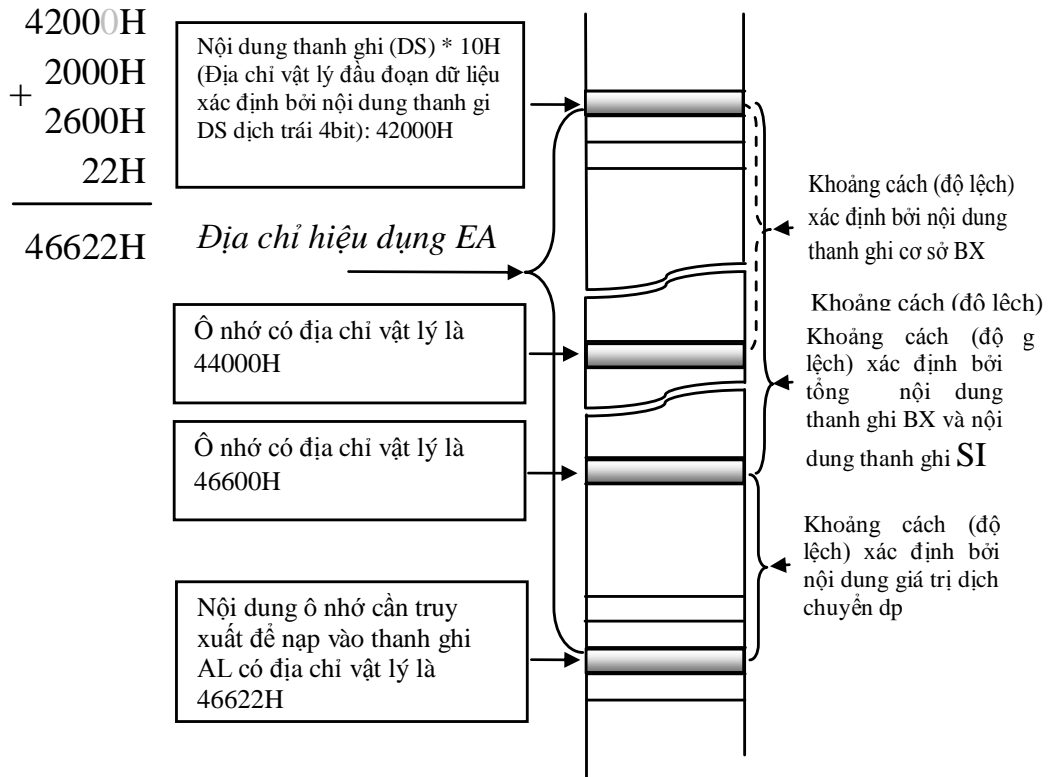
Thí dụ MOVS ; di chuyển chuỗi, nguồn tại vùng nhớ có địa chỉ đầu là DS : SI, đích là vùng nhớ có địa chỉ đầu DS : DI.



Hình II.23 Quản lý bộ nhớ theo đoạn (segment)

Ví dụ:

Nội dung thanh ghi DS là 4200H, nội dung thanh ghi BX là 2000H, nội dung thanh ghi SI là 2600H. Hãy viết lệnh chuyển dữ liệu ở ô nhớ có địa chỉ vật lý là 46622H vào thanh ghi AL.

Lời giải:

Thấy rằng địa chỉ vật lý đầu đoạn dữ liệu được xác định bởi nội dung thanh ghi đoạn DS và có giá trị là $4200\text{H} * 10\text{H} = 42000\text{H}$. Địa chỉ vật lý của ô nhớ cần truy xuất là 46622H, vậy độ lệch đối với địa chỉ nền (hay còn gọi là địa chỉ hiệu dụng) sẽ là $46622\text{H} - 42000\text{H} = 4622\text{H}$. Tổng nội dung thanh ghi BX và nội dung thanh ghi SI là $2000\text{H} + 2600\text{H} = 4600\text{H}$, như vậy, khoảng cách còn lại so với giá trị địa chỉ hiệu dụng là 22H, giá trị này có thể sử dụng như là *giá trị dịch chuyển dp*, vậy ta có thể sử dụng phương thức truy xuất bằng *địa chỉ cơ sở-chỉ số*, nghĩa là có thể sử dụng lệnh chuyển dữ liệu từ ô nhớ có địa chỉ vật lý là 46622H vào thanh ghi AL như sau:

```
MOV AL, [BX][SI]+22H
```

Giải thích lệnh: Nạp vào thanh ghi AL nội dung ô nhớ có địa chỉ vật lý được xác định bởi nội dung thanh ghi đoạn DS, nội dung thanh ghi cơ sở BX, nội dung thanh ghi chỉ số SI và giá trị dịch chuyển $dp = 22\text{H}$. $(AX) \leftarrow ((BX) + (SI) + 22\text{H})$

II.2.6 Phương thức đánh địa chỉ thiết bị ngoại vi

Có hai phương thức cơ bản đánh địa chỉ thiết bị ngoại vi:

– Định địa chỉ tách biệt (isolated I/O address): Các tín hiệu điều khiển phải được phân biệt đối với các thao tác ghi/đọc bộ nhớ và ghi/đọc thiết bị ngoại vi. Trong hệ Vi xử lý $\mu P8085$, tổ hợp các tín hiệu \overline{RD} , \overline{WR} và $\overline{IO/\overline{M}}$ sẽ được giải mã để tạo ra các tín hiệu đọc/ghi riêng cho bộ nhớ (\overline{MEMR} và \overline{MEMW}) và riêng cho thiết bị ngoại vi (\overline{IOR} và \overline{IOW}). Đối với họ 80x86, đó là việc sử dụng chip điều khiển BUS (BUS Controller 8288) để giải mã tổ hợp các tín hiệu nhịp đồng hồ CLK, các tín hiệu trạng thái $\overline{S2}$, $\overline{S1}$ và $\overline{S0}$ trong chế độ MAX thành các tín hiệu \overline{MRDC} , \overline{MWTC} , \overline{IORC} và \overline{IOWC} . Các mạch logic phụ trợ điều khiển truy nhập thiết bị ngoại vi trong hệ Vi xử lý có nhiệm vụ phát hiện các tín hiệu \overline{IORC} và \overline{IOWC} để thực hiện các thao tác vào/ra dữ liệu. Mạch logic này có nhiệm vụ giải mã địa chỉ thiết bị ngoại vi để tạo ra các tín hiệu cho phép truy nhập tới thiết bị cụ thể (thường được gọi là mạch giải mã địa chỉ thiết bị ngoại vi). Cũng cần nói thêm rằng địa chỉ thiết bị ngoại vi thực tế là địa chỉ của một thanh ghi trong thiết bị ngoại vi. Như vậy, việc trao đổi dữ liệu giữa CPU và thiết bị ngoại vi thực chất là trao đổi dữ liệu giữa CPU và thanh ghi trong không gian thiết bị ngoại vi. Các $\mu P80x86$ dành 16 dây địa chỉ thấp (A15 – A0) để quản lý một không gian 64K thiết bị ngoại vi.

– Định địa chỉ tuyến tính (Linear Addressing I/O), cũng còn gọi là định địa chỉ thiết bị ngoại vi theo bản đồ nhớ (Memory-Mapped I/O): Thanh ghi trong thiết bị ngoại vi được coi như một vị trí nhớ trong không gian nhớ, do vậy không sử dụng đến các tín hiệu điều khiển riêng cho việc trao đổi dữ liệu giữa CPU với thiết bị ngoại vi, mà sử dụng hoàn toàn chung cho bộ nhớ cũng như cho thiết bị ngoại vi. Đối với $\mu P8085$, tín hiệu phân biệt $\overline{IO/\overline{M}}$ không cần thiết nữa, cũng như không cần giả mã các tổ hợp tín hiệu $\overline{S2}$, $\overline{S1}$ và $\overline{S0}$ đối với các trung tâm 80x86. Mọi thao tác trao đổi dữ liệu giữa CPU và các thanh ghi thiết bị ngoại vi đều được tiến hành như với một ô nhớ trong bộ nhớ.

II.2.7 Các mạch Multiplexer, mạch Decoder, mạch PLA

Các mạch Multiplexer, mạch Decoder hay mạch PLA là những mạch phụ trợ không thể thiếu của một hệ Vi xử lý. Thông thường, các mạch Decoder và mạch PLA (Programmable Logic Array) được thiết kế sẵn trên một chip, được sử dụng nhiều trong các mạch giải mã các tín hiệu điều khiển, giải mã địa chỉ của vùng nhớ hay địa chỉ thiết bị ngoại vi.

– Các mạch Multiplexer (hoặc Coder)

Mạch Multiplexer (còn gọi là mạch Coder) thường được xây dựng theo mục đích sử dụng, có khi rất phức tạp. Một trong những ví dụ là mạch thu nhận và mã hoá bàn phím (keyboard), được xây dựng trên cơ sở một chip Vi xử lý chuyên dụng, bao gồm cả phần cứng lẫn chương trình. Sơ đồ khối chức năng của một mạch Multiplexer được thể hiện trên Hình II.24. Các tín hiệu vào riêng rẽ $x_1, x_2, x_3 \dots, x_n$, qua xử lý sẽ tạo ra một tổ hợp nhị phân đầu ra $\{y_m y_{m-1} \dots y_0\}$. Phân chuyển đổi từ một tín hiệu vào x_i thành tổ hợp ra $\{y_m y_{m-1} \dots y_0\}$ được thực hiện nhờ mạch tổ hợp logic hoặc kết hợp với phần mềm chuyên dụng.



Hình II. 24 Sơ đồ khối một mạch Multiplexer (Coder)

– Mạch giải mã (Decoder)

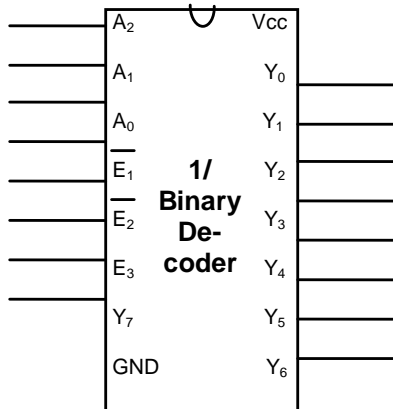
Các mạch giải mã thông thường 1/4, 1/8 được xây dựng như một chip phụ trợ trong các hệ Vi xử lý. Có thể kể đến như mạch giải mã 1/16 SN74154, mạch giải mã 1/8 74138 v.v... Bảng chân lý của mạch giải mã 1/8 như sau:

E ₃	E ₂	E ₁	A ₂	A ₁	A ₀	Y ₀	Y ₁	Y ₂	Y ₃	Y ₄	Y ₅	Y ₆	Y ₇
1	0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	0	0	1	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1	1	1	1	0	1	1
1	0	0	1	1	0	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0
0	x	x	x	x	x	1	1	1	1	1	1	1	1
x	1	x	x	x	x	1	1	1	1	1	1	1	1
x	x	1	x	x	x	1	1	1	1	1	1	1	1

Hình II. 25 Sơ đồ nối chân mạch giải mã nhị phân 1/8 và bảng chân lý

Vi mạch giải mã nhị phân 1/8 có sơ đồ nối chân như Hình II. 18. Khi vi mạch giải mã được “Enable”, ứng với tổ hợp các giá trị $E_3 E_2 E_1 = 100$, và với bất kỳ tổ hợp nào của các giá trị $A_2 A_1 A_0$ đều có một lối ra có giá trị LOW.

Ứng với lối ra này sẽ là một vị trí hoặc một vùng nhớ được chọn, hoặc một thiết bị ngoại vi. Đối với các vi mạch có chân \overline{CS} (chip select), đây là tín hiệu chọn vô thích hợp.



– Mạch PLA (Programmable Logic Array)

Mạch PLA thực chất là một chip nhớ ROM được ghi sẵn theo một quy luật nào đó theo phương thức giải mã một tổ hợp nhị phân ở đầu vào. Có nghĩa là ứng với một ô nhớ là một tổ hợp giá trị theo một quy luật giải mã đầu vào, mà đầu vào đây chính là địa chỉ của ô nhớ đó. Các mạch PLA thích hợp với những nơi cần sử dụng bộ giải mã với số lượng đầu vào lớn hơn 3.

II.2.8 Vài nét về lập trình hợp ngữ

Hợp ngữ (Assembler) là một công cụ rất mạnh được sử dụng trong việc phát triển mã lệnh của các hệ Vi xử lý và máy vi tính. Hợp ngữ là chương trình dịch các lệnh gợi nhớ (Mnemonics) và các ký hiệu (symbols) thành mã máy cho các hệ vi xử lý và máy vi tính thực hiện. Cần phân biệt rằng hợp ngữ là một chương trình, chứ không phải là một phần của phần cứng.

Dữ liệu vào của hợp ngữ là tập các lệnh gợi nhớ, và dữ liệu ra của hợp ngữ chính là các tập các byte mã máy nhị phân, mã thực thi được đánh địa chỉ chính xác trong không gian nhớ chương trình.

Dữ liệu vào được gọi là mã nguồn (source code), dữ liệu ra được gọi là mã thực thi hoặc mã đối tượng (object code). Quá trình mã nguồn được dịch thành mã đối tượng được gọi là assembly. Công cụ phần mềm thực hiện quá trình này gọi là hợp ngữ (assembler). Có thể thấy rất dễ dàng rằng: viết một lệnh MOV A,M dễ nhớ hơn rất nhiều so với mã hexa của lệnh này: 7E_H hoặc mã nhị phân 0111 1110_B.

Hiện có hai loại chương trình hợp ngữ đang được sử dụng rộng rãi: Hợp ngữ thường trú (Resident Assemblers) - được cài đặt ngay trong hệ thống, và hợp ngữ chuyển đổi (Cross Assemblers) không được cài đặt ngay trong hệ thống, mà là trong một máy chủ khác. Mã chương trình do máy chủ tạo ra từ hợp ngữ không thể chạy được trên máy chủ.

Ngoài ra còn tồn tại hai loại hợp ngữ khác là hợp ngữ tuyệt đối (Absolute Assembler), và hợp ngữ tái định vị (Relocatable Assemblers), sẽ được giới thiệu trong các trình học hợp ngữ sau này.

II.3 Cấu trúc và tính năng của một số chip Vi xử lý hiện đại.

Trải qua mấy thập kỷ phát triển, công nghệ chế tạo các chip Vi xử lý đã có những bước tiến vũ bão. Đã xuất hiện nhiều kiểu cấu trúc chip Vi xử lý như CISC (Complete Instruction-Set Computer), RISC (Reduced Instruction-Set Computer), bộ xử lý scalar hay superscalar, Vi xử lý VLIW (Very Long Instruction Word), Vi xử lý Superpipelined, Vi xử lý Vector, và Vi xử lý biểu tượng (Symbolic μ P), nhằm đáp ứng nhu cầu tạo nên những máy tính cực mạnh, những siêu máy tính, mainfram phục vụ những công việc tính toán lớn hay tạo ra các máy tính xử lý song song.

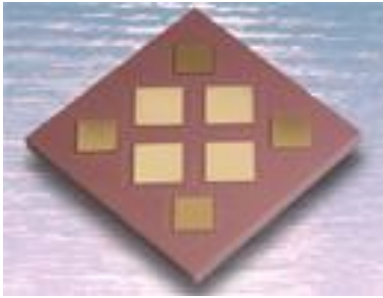
Đối với họ x86, đã có các trung tâm i486 với kiến trúc RISC, tập lệnh rút gọn với tốc độ xử lý tăng nhanh đáng kể. Đó là những trung tâm xử lý 32 bits thực sự. Không gian địa chỉ vật lý và không gian bộ nhớ ảo được quản lý bởi 32 bits địa chỉ, lên đến 4Gbytes. Ngoài ra, các bộ đồng xử lý toán cũng được tích hợp tạo nên sức mạnh đáng kể. Các trung tâm này đã được sử dụng để tạo nên những máy tính xử lý song song với cấu trúc hiện đại và khả năng tính toán lớn cho phép giải những bài toán rất phức tạp.

Năm 1992, chip xử lý Pentium MMX ra đời, trong cấu trúc có tới hai đường ống song song (superscalar), hai khối số học và logic (ALU) cho phép thi hành hai lệnh máy trong một chu kỳ. BUS nội bộ của Pentium MMX là BUS 64 bits và 128 bits, tốc độ trao đổi dữ liệu với bộ nhớ do vậy được nâng cao đáng kể. Đặc biệt, chip Pentium có một vùng nhớ gọi là vùng đệm đích rẽ nhánh BTB (Branch Target Buffer) đối với 256 lệnh rẽ nhánh mới. Pentium cũng được tích hợp một bộ đồng xử toán học (*Mathematical Coprocessor*) với hiệu suất rất cao nhờ giải thuật nhanh hơn. Dù là loại Vi xử lý CISC, nhưng Pentium đã ứng dụng giống như các loại Vi xử lý RISC tốc độ cao: xử lý đường ống, cấu trúc superscalar và dự đoán rẽ nhánh.

Năm 1999, chip Pentium PIII ra đời với cấu trúc có thêm 70 lệnh cho truyền thông đa phương tiện, tốc độ xung nhịp đã vượt qua ngưỡng 1GHz. Tiếp theo đó là các chip Pentium PIV với tốc độ cao hơn hẳn và cấu trúc đa phân luồng.

IBM cho biết họ đang áp dụng những phương pháp sản xuất mới nhằm cho phép thiết bị xử lý dành cho máy chủ Power6 có thể chạy nhanh gấp hai lần so với hiện nay nhưng vẫn đảm bảo yêu cầu về nhiệt độ.

Theo thông tin từ Giám đốc kỹ thuật của IBM, họ không chỉ thu nhỏ kích cỡ các bóng bán dẫn (transistor) mà còn thay đổi phương thức hoạt động của silicon khi đặt lớp cách điện phía dưới một lớp silicon gồm khoảng 500 nguyên tử.



Chip IBM Power5. (IBM)

Những cải tiến đó khiến Power6, sản xuất theo công nghệ 65 nm và dự định ra mắt giữa năm 2007, đạt xung nhịp tới 4 - 5 GHz. IBM khẳng định Power6 sẽ cạnh tranh trực tiếp với sản phẩm của các đối thủ như Intel, AMD và Sun Microsystems.

Ngay khi Intel giới thiệu công nghệ 90 nm năm 2003, người ta nhận thấy nó gây ra tình trạng thất thoát năng lượng nghiêm trọng hơn những phương pháp sản xuất trước đó, khiến chip tỏa nhiệt ngay cả khi chúng không chạy hết công suất. Một trong những biện pháp khắc phục là tích hợp 2 lõi xử lý trên một chip đơn và giảm xung nhịp hoạt động để tăng khả năng vận hành cũng như tránh rắc rối do nhiệt độ cao. Ngược lại, Power6 được xây dựng để hoạt động ở xung nhịp cao chưa từng thấy nhưng vẫn tiêu thụ điện năng hiệu quả.

Hiện nay, Intel cũng đang nghiên cứu hai kỹ thuật sản xuất và thiết kế mới nhằm giảm lượng điện tiêu thụ trên bảng mạch hệ thống. Phương pháp thứ nhất cung cấp nguồn điện áp cho cả CPU và bộ nhớ đệm (cache) còn cách thứ hai sẽ tích hợp bộ điều hòa điện áp trên các transistor.

IBM phát triển chip 500 GHz



Chip SiGe ở chế độ đông lạnh có xung nhịp gấp 250 lần so với chip ĐTĐĐ thông thường.

Ảnh: Marketwire

"Big Blue" và Công ty Georgia Tech đã cùng chế tạo một chip có xung nhịp cao hơn 100 lần so với kỷ lục của thiết bị xử lý máy tính hiện nay, với điều kiện nó phải hoạt động ở nhiệt độ nghe có vẻ phi thực tế - 268,5°C.

Ở nhiệt độ trong phòng, chip này vẫn đạt tốc độ 350 GHz, tương đương 350 tỷ vòng/giây, nhanh hơn nhiều so với thiết bị xử lý máy tính tại thời điểm này (dao động từ 1,8 GHz đến 3,8 GHz).

Đây là một phần dự án khám phá tốc độ tối đa của các chip silicon-germani (SiGe). SiGe cũng giống như công nghệ chip silicon khác nhưng được tăng cường nguyên tố germani để nâng hiệu suất và giảm lượng điện tiêu thụ. Trên lý thuyết, SiGe có thể mở rộng tốc độ lên 1 terahertz (THz), tức một nghìn tỷ vòng mỗi giây.

Tuy nhiên, thêm thành phần germani đồng nghĩa với chi phí sản xuất tấm wafer tăng cao, do đó SiGe rất kén chọn thị trường. IBM đã bán ra hàng trăm triệu chip này từ năm 1998, nhưng chưa thể địch nổi khi so với con số hàng tỷ chip silicon mỗi năm nhờ sản lượng điện thoại di động.

Chip SiGe hiệu suất lớn sẽ được ứng dụng trong các hệ thống phòng thủ, phương tiện khám phá vũ trụ và thiết bị cảm ứng từ xa.

Một số mốc phát triển thương hiệu chip của Intel

Năm 2003: Những máy tính xách tay đầu tiên sử dụng Centrino đã giới thiệu khái

niệm hoàn toàn mới về một nền tảng máy tính tích hợp với hiệu suất hoạt động mạnh mẽ, kết nối không dây và thời gian sử dụng pin lâu.

Năm 2004: Bộ vi xử lý di động Pentium M là trái tim của những hệ thống sử dụng công nghệ Intel Centrino với tốc độ xử lý lên tới 2,13 GHz và tích hợp sẵn khả năng kết nối Wi-Fi.

Năm 2005: Intel giới thiệu thế hệ tiếp theo của vi xử lý Centrino, giúp tăng cường hiệu suất và nâng cao bảo mật cho người dùng doanh nghiệp, đồng thời mang lại công nghệ đồ họa, âm thanh và hình ảnh chất lượng cao.

Năm 2006: Intel Core 2 Duo ra đời, mang đến hiệu suất hoạt động nhanh gấp 2 lần và khả năng tiết kiệm điện năng cao.

Năm 2007: Vi xử lý Centrino Pro bảo đảm khắc phục sự cố nhanh, chi phí thấp và tăng cường bảo mật.

Năm 2008: Thế hệ mới của Centrino được tiếp thêm sức mạnh với bộ vi xử lý 45 nanomet (nm) tiên tiến của Intel là Core 2 Duo.

II.3.1 Cấu trúc chip Vi xử lý Pentium

Pentium là loại đơn vị xử lý trung tâm 32 bit và là loại đơn vị xử lý trung tâm đầu tiên sử dụng kỹ thuật ILP (Instruction Level Pararellism), kỹ thuật xử lý lệnh song song.

Kỹ thuật đường ống và kỹ thuật xử lý lệnh song song ILP

Một lệnh thường được xử lý qua năm giai đoạn:

1. Nhập lệnh (FI Fetch the Instruction)
2. Giải mã lệnh (DI Decode the Instruction)
3. Tạo địa chỉ toán hạng (GOA Generate Operand Address)
4. Nhập toán hạng (FO Fetch Operands)
5. Thực hiện lệnh (EI Execute Instruction)

Với kỹ thuật xử lý lệnh thông thường, đơn vị xử lý trung tâm phải tuần tự thực hiện xong tất cả các giai đoạn thực hiện một lệnh, thường là sau 5 chu kỳ máy, rồi mới chuyển sang nhập và thực hiện lệnh tiếp theo. Để tăng tốc độ xử lý lệnh mà không nhất thiết phải tăng tần số nhịp của máy, người ta sử dụng các kỹ thuật khác như kỹ thuật xử lý lệnh kiểu đường ống (Pipeline) và kỹ thuật xử lý lệnh song song (ILP).

- Kỹ thuật xử lý lệnh kiểu đường ống (Pipeline)

Kỹ thuật xử lý lệnh kiểu đường ống được sử dụng trong các đơn vị xử lý trung tâm từ đời đơn vị xử lý trung tâm Intel 8086.

Chu kỳ máy

1	2	3	4	5	6	7	8	9
FI								
	DI							
		GOA						
			FO					
				EI				
	FI							
		DI						
			GOA					
				FO				
					EI			
		FI						
			DI					
				GOA				
					FO			
						EI		

Đường ống tương tự như dây chuyền sản xuất nhiều công đoạn. ở dây chuyền sản xuất, mỗi công đoạn thực hiện một thao tác sản xuất. Sản phẩm được chuyển và hình thành dần sau mỗi công đoạn sản xuất, cho đến khi được hoàn thành ở công đoạn cuối cùng. Trong kỹ thuật xử lý lệnh theo kiểu đường ống, việc thực hiện lệnh cũng được thực hiện qua 5 thao tác, mỗi thao tác được thực hiện trong một giai đoạn, giai đoạn nọ tiếp sau giai đoạn kia, cho đến khi thực hiện xong lệnh. Với một đường ống 5-giai đoạn, tại mỗi chu

kỳ máy có 5 bộ dữ liệu thuộc 5 giai đoạn xử lý được gửi vào đường ống và 5 thao tác được thực hiện đồng thời, nhờ vậy mà sau mỗi chu kỳ máy lại có một lệnh được hoàn thành và một lệnh mới được nhập. Kỹ thuật đường ống với đường ống 5-giai đoạn cho phép tăng tốc độ thực hiện lệnh lên gấp 5 lần.

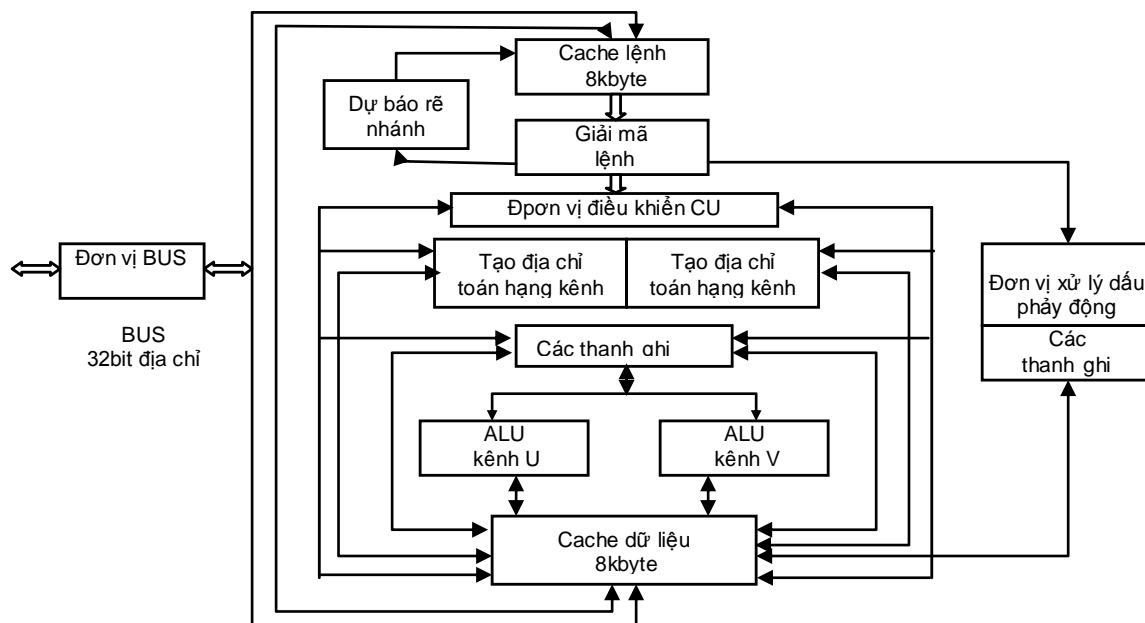
- Kỹ thuật ILP (xử lý lệnh song song)

Kỹ thuật ILP là kỹ thuật thiết kế đơn vị xử lý trung tâm và chương trình dịch nhằm làm tăng tốc độ các thao tác máy (như ghi-đọc bộ nhớ) và thực hiện các phép tính. Trong các kỹ thuật ILP có kỹ thuật *superscalar*, trong đó tại một chu kỳ máy nhiều lệnh được nhập và được thực hiện đồng thời trên nhiều đường ống khác nhau.

Pentium là loại đơn vị xử lý trung tâm được thiết kế theo kỹ thuật superscalar, trong đó hai lệnh được nhập và giải mã đồng thời. Pentium có hai đường ống thực hiện lệnh song song U và V. Quá trình thực hiện lệnh được mô tả như sau :

Chu kỳ máy	1	2	3	4	5	6	7
FI		DI	GOA	FO	EI		
			GOA	FO	EI		
		FI	DI	GOA	FO	EI	
			GOA	FO	EI		
			FI	DI	GOA	FO	EI
				GOA	FO	EI	

Cấu trúc của Pentium



Hình II.26 Cấu trúc trung tâm Vi xử lý Pentium

Pentium là đơn vị xử lý trung tâm loại 32bit và là đơn vị xử lý trung tâm loại CISC (Complex Instruction Set Computer) với đặc điểm: hệ lệnh phức tạp, nhiều kiểu xác định địa chỉ, nhiều khuôn dạng lệnh và nhiều kích thước lệnh khác nhau.

Pentium có BUS địa chỉ trong và ngoài 32 bit, BUS dữ liệu trong và ngoài 64 bit. Pentium có hai cache 8 Kbyte độc lập : một cache 8 Kbyte 2 công dành cho dữ liệu và một cache 8 Kbyte chứa lệnh. Pentium có hai đường ống thực hiện lệnh song song U và V, và 2 đơn vị số học-logic ALU. Pentium có một đường ống riêng thực hiện các lệnh dấu phẩy động và một bộ đồng xử lý dấu phẩy động FPU được tích hợp trong chip.

Pentium có các thanh ghi sau :

Các thanh ghi hệ thống :

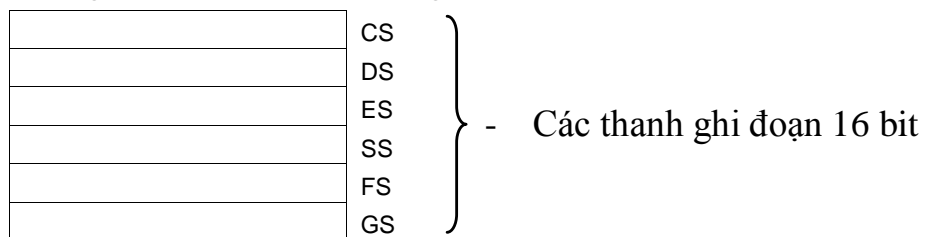
- Các thanh ghi điều khiển 32 bit: CR0, CR1, CR2, CR3
- Các thanh ghi hệ thống quản lý bộ nhớ :

GDTR: Thanh ghi bảng mô tả toàn cục (Global Descriptor Table Register)

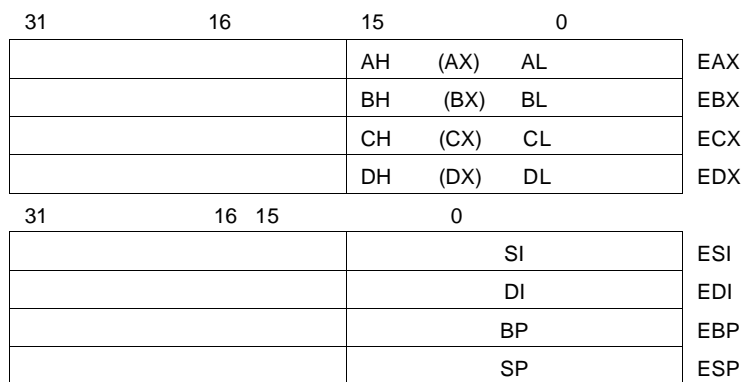
LDTR: Thanh ghi bảng mô tả cục bộ (Local Descriptor Table Register)

IDTR: Thanh ghi bảng mô tả ngắt (Interrupt Descriptor Table Register)

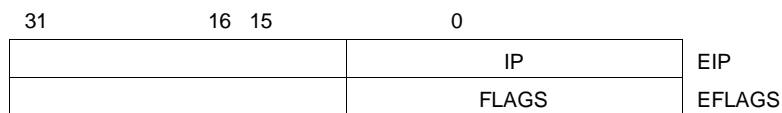
TR: Thanh ghi nhiệm vụ (Task Register)



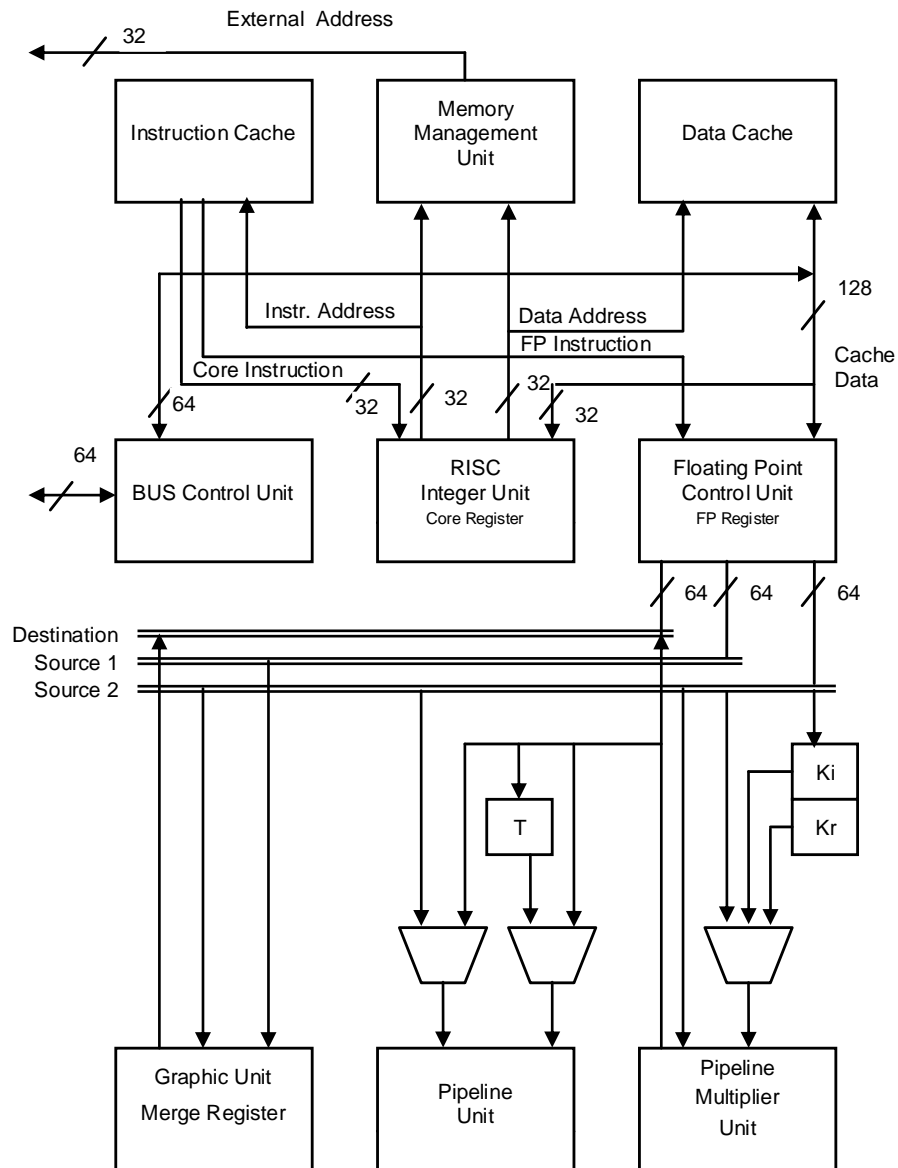
- Các thanh ghi đa năng 32 bit :



- Con trỏ lệnh EIP và thanh ghi cờ EFLAGS 32 bit :



Hình II.27 Các thanh ghi và cấu trúc bên trong trung tâm Vi xử lý Pentium

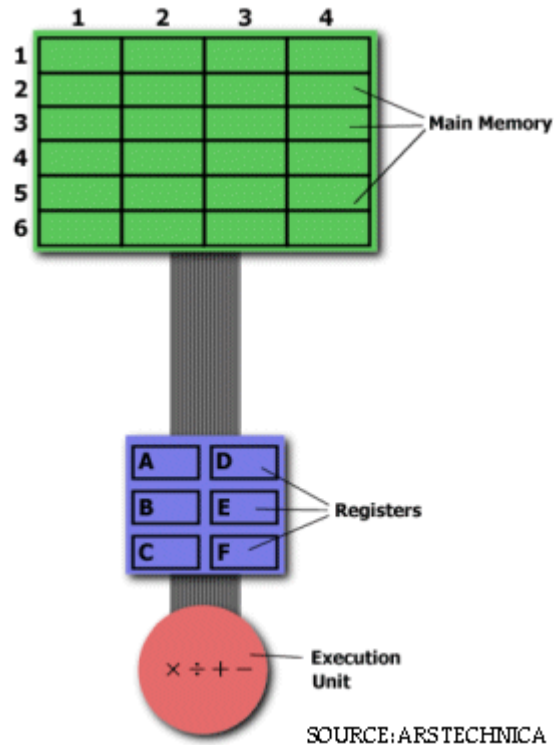


Hình II. 28. Cấu trúc CPU Pentium

II.3.2 Kiến trúc RISC, CISC

RISC: (Reduced Instruction-Set Computer)

CISC: (Complete Instruction-Set Computer)



Hình II.29 Tổ chức máy tính

Cách đơn giản nhất để có thể khảo sát những ưu nhược điểm của kiến trúc RISC là so sánh việc thực hiện một phép toán đối với loại kiến trúc CISC trước đây. Giả sử ta phải thực hiện một lệnh nhân hai toán hạng được lưu giữ trong bộ nhớ.

Hình II. ... mô tả tổ chức của một máy tính. Bộ nhớ được tạo từ các ô nhớ từ 1:1 (hàng 1: cột 1) đến 6:4 (hàng 6 : cột 4). Khối thực hiện lệnh có nhiệm vụ thực hiện các lệnh tính toán x (nhân), ÷ (chia), + (cộng) và - (trừ). Tất nhiên, khối thực hiện tính toán chỉ có thể làm việc với các dữ liệu (toán hạng) đã được chứa sẵn ở một trong các thanh ghi A,B,C,D,E hoặc F. Giả sử ta phải tìm tích 2 số, số thứ nhất được chứa ở ô 2:3 và số thứ hai ở ô 5:2, kết quả sẽ được lưu lại vào ô 2:3. Bây giờ ta sẽ tiếp cận cách giải quyết vấn đề trên hai loại CPU, CISC và RISC.

Trên CPU CISC: ưu tiên hàng đầu của loại CPU này là hoàn thiện một công việc với ít lệnh nhất có thể. Điều này có thể thực hiện nhờ vào việc xây dựng một phần cứng CPU có khả năng *hiểu được và thực hiện được* một chuỗi các tác nghiệp. Trong trường hợp cụ thể này, CISC sẽ có một lệnh xác định duy nhất, tạm gọi là MULT, mà khi thực hiện, lệnh sẽ nạp hai giá trị toán hạng vào 2 thanh ghi sau đó thực hiện phép nhân rồi ghi kết quả vào một

thanh ghi tương ứng. Như vậy công việc sẽ được thể hiện bằng một lệnh như sau:

MULT 2:3,5:2

Lệnh MULT là một lệnh hoàn thiện (complex). Lệnh làm việc trực tiếp trên băng nhớ của máy tính, chứ không cần người lập trình phải dùng lệnh gọi hay nạp nội dung, ghi nội dung vào ô nhớ. Lệnh rất gần với ngôn ngữ bậc cao. Giả sử ta gọi “ a ” là giá trị của toán hạng trong ô nhớ 2:3 và “ b ” là giá trị toán hạng trong ô nhớ 5:2, thì lệnh tương ứng trong ngôn ngữ C là “ $a = a*b$ ”.

Ưu điểm lớn nhất của hệ thống CISC là chương trình dịch phải làm rất ít việc khi dịch một chương trình, hay một lệnh của ngôn ngữ bậc cao sang ngôn ngữ máy. Vì độ dài của mã lệnh rất nhỏ, nên hệ thống cũng cần ít RAM hơn để ghi nhớ lệnh. Dĩ nhiên, việc thiết kế cấu trúc loại CISC đặc biệt sẽ phải tích hợp các lệnh hoàn thiện bằng phần cứng.

Trên CPU RISC: Các CPU loại RISC chỉ sử dụng các lệnh (Instruction) có thể thực hiện được trong một chu kỳ xung nhịp. Như vậy lệnh MULT được mô tả ở phần trên phải được chia thành 3 lệnh nhỏ hơn: “LOAD” chuyển dữ liệu (toán hạng) từ ô nhớ vào thanh ghi; “PROD” thực hiện phép nhân hai toán hạng được lưu giữ trong các thanh ghi, và lệnh “STORE” sẽ thực hiện việc chuyển kết quả tính toán ghi vào ô nhớ. Để thực hiện được phép nhân hai toán hạng, người lập trình phải mã hoá thành 4 lệnh như sau:

LOAD A, 2:3

LOAD B, 5:2

PROD A, B

STORE 2:3, A

Có thể thấy rằng với kiến trúc RISC, không thuận lợi lắm cho hoàn thành phép toán nhân hai số vì phải viết nhiều dòng lệnh hơn, cần nhiều RAM hơn để lưu giữ các lệnh mức assembly. Chương trình dịch cũng phải thực hiện nhiều việc hơn để chuyển đổi các lệnh của ngôn ngữ bậc cao sang mã máy.

Thế nhưng, chiến lược RISC đã mang đến nhiều thuận lợi quan trọng. Vì mỗi lệnh chỉ cần một chu kỳ xung nhịp để thực hiện, toàn bộ chương trình cũng sẽ chỉ cần số chu kỳ xung nhịp như khi thực hiện lệnh MULT ở hệ thống CISC. Nhưng kiến trúc RISC với hệ lệnh rút gọn cần ít linh kiện và không gian cho mạch tích hợp, bỏ qua được các thanh ghi đa năng. Hơn nữa, mỗi lệnh chỉ thực thi trong một chu kỳ xung nhịp nên việc tổ chức đường ống cũng đơn giản hơn nhiều.

Việc tách lệnh “LOAD” và lệnh “STORE” đã đơn giản hoá đáng kể khối lượng công việc CPU phải thực hiện. Sau khi thực hiện lệnh MULT ở cấu trúc CISC, CPU tự động xoá nội dung các thanh ghi. Nếu một toán hạng nào đó còn tiếp tục được sử dụng cho lệnh tiếp theo, CPU phải nạp lại. Ở kiến

trúc RISC, nội dung của toán hạng vẫn được giữ lại cho đến khi một giá trị mới được nạp vào.

Cuối cùng, để so sánh một cách toàn diện hơn, công thức sau được dùng để đánh giá khả năng tính toán, xử lý của các loại CPU:

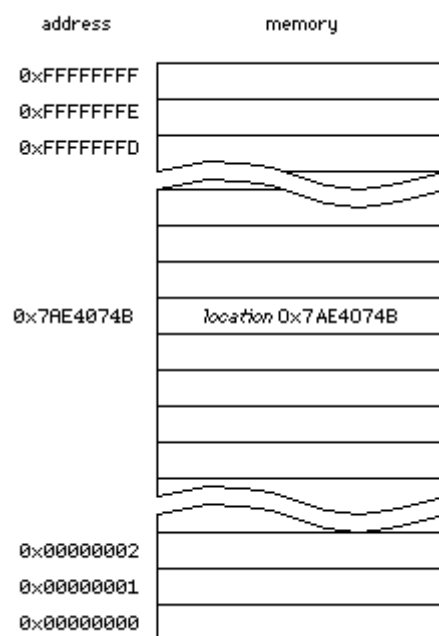
$$\frac{\text{time}}{\text{program}} = \frac{\text{time}}{\text{cycle}} \times \frac{\text{cycles}}{\text{instruction}} \times \frac{\text{instructions}}{\text{program}}$$

CISC cố gắng giảm số lệnh trong một chương trình, hy sinh số chu kỳ thực hiện một lệnh trong khi RISC theo chiến lược ngược lại.

Một số thông tin thú vị cho độc giả: Chỉ các chip họ x86 vẫn trung thành với kiến trúc CISC, dĩ nhiên không được thông dụng lắm vì những lý do khác: Trước hết do sự phát triển vũ bão của công nghệ tích hợp mạch, trong công nghệ sản xuất linh kiện điện tử. Sự giảm giá đến mức khó hiểu của bộ nhớ RAM cũng làm đảo lộn cách nhìn nhận những nhược điểm của các CPU theo kiến trúc RISC. Giá 1Mbyte RAM năm 1977 là khoảng 5000USD, nhưng đến năm 1994 là khoảng 6USD, còn tại thời điểm này (2005) là khoảng hơn 0,2 USD! Công nghệ chương trình dịch (compiler technology) cũng trở nên hoàn thiện hơn nên CPU loại RISC cùng với bộ nhớ RAM dung lượng lớn và công nghệ phần mềm đã trở thành lý tưởng hơn nhiều đối với các hãng sản xuất máy tính.

II.3.3 Quản lý bộ nhớ

Địa chỉ (address) là phương thức duy nhất để “xác định vị trí (location)” của một ô nhớ trong “không gian địa chỉ” (address space).



Địa chỉ được thể hiện bằng một số nguyên nhị phân không dấu và được lưu giữ trong các thanh ghi chuyên dụng và thanh ghi đa năng với những kỹ thuật hoàn thiện. Địa chỉ được giải mã bằng phần cứng để truy xuất đến một vị trí nhớ trong các khối nhớ vật lý, ví dụ bộ nhớ RAM hoặc ROM hay trong một nguồn nhớ được bản đồ hoá (memory mapped resource).

Hình bên biểu diễn cách nhìn tổng quát về địa chỉ, không

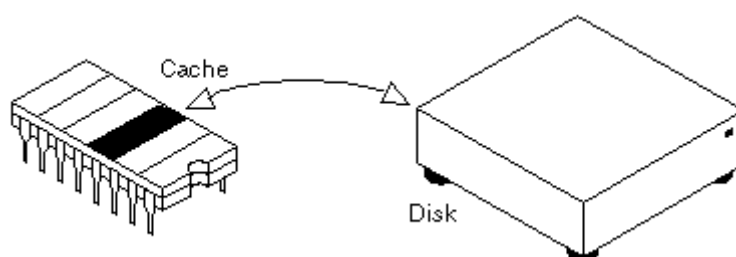
gian địa chỉ và vị trí nhớ trong kiến trúc máy tính 32 bit. Có thể thấy địa chỉ như là một con trỏ (pointer), một số nguyên nhị phân tham chiếu đến một đối tượng hay một vị trí nhớ (ô nhớ). Dĩ nhiên, để tạo ra được một con trỏ, các kỹ thuật như phân đoạn (segment), sử dụng độ lệch (offset) và giá trị dịch chuyển (displacement) được sử dụng và được tạo nhờ đơn vị giao diện BUS (BIU) trong các CPU.

Không gian địa chỉ là tập tất cả các địa chỉ, cũng có thể hình dung như là một hàm riêng tham chiếu đến các ô nhớ. Thông thường, địa chỉ bắt đầu từ 0 (zero) cho đến 2^N-1 , trong đó N là độ rộng của BUS địa chỉ (16, 20, 24, 32 hoặc 64). Không gian này có thể không chính xác với kiến trúc phân đoạn.

Trong các hệ thống hiện đại, phần lớn không gian địa chỉ có thể được dữ trữ nhờ kiến trúc của hệ điều hành, hoặc tạm thời không được bản đồ hoá. Những vấn đề liên quan độc giả có thể tìm thấy trong các tài liệu về không gian bộ nhớ ảo và không gian bộ nhớ vật lý.

II.3.4 Bộ nhớ cache

Cache là cơ chế nhớ tốc độ cao đặc biệt. Cache có thể sử dụng như một vùng nhớ dữ trữ trong bộ nhớ chính nhưng với những chip nhớ tốc độ cao. Có hai loại bộ nhớ cache được sử dụng chung trong máy PC, *memory caching* và *disk caching*.



Memory cache còn được gọi bộ nhớ cache hay RAM cache, sử dụng RAM tĩnh (SRAM) tốc độ cao. Rất hiệu quả vì nhiều chương trình truy nhập các dữ liệu hoặc lệnh thông qua vùng nhớ này. Bằng cách lưu giữ dữ liệu và lệnh trong cache, tốc độ truy nhập bộ nhớ được nâng cao. Cũng có một loại memory cache được tích hợp trực tiếp trong CPU như ở các CPU 80486 (8KB), ở Pentium là 16KB. Chúng được gọi là cache nội bộ (*Internal cache*), hay cache mức 1 (L1). Các PC còn hỗ trợ cache ngoài (*External cache*), còn gọi là cache L2, là bộ nhớ được dùng trung gian giữa CPU và bộ nhớ chính DRAM.

Disk cache làm việc giống như nguyên lý của cache nhớ, nhưng thay vì sử dụng SRAM, cache đĩa sử dụng DRAM như bộ nhớ chính. Phần lớn dữ liệu được truy xuất từ đĩa được lưu giữ trong các vùng nhớ đệm. Mỗi khi

chương trình truy xuất đĩa, thông thường nó kiểm tra xem, các dữ liệu đó đã được lưu vào vùng cache đĩa hay chưa. Cache đĩa đóng vai trò rất quan trọng trong việc nâng cao tốc độ truy xuất, vì truy xuất một byte dữ liệu trong RAM có thể nói nhanh hơn gấp ngàn lần truy xuất một byte dữ liệu từ các ổ đĩa. Khi dữ liệu được tìm thấy trong bộ nhớ cache, tức là *cache hit*, và hiệu suất của cache được đánh giá bằng *hit rate*. Hầu hết các hệ thống cache đều sử dụng kỹ thuật *smart caching*, có nghĩa là hệ thống luôn luôn ghi nhận một số loại dữ liệu thường được sử dụng nhất. Chiến lược xác định các thông tin nào được lưu giữ vào trong bộ nhớ cache là vấn đề được đặc biệt quan tâm trong khoa học máy tính.

II.4 Single-Chip MicroComputer μ C8051

II.4.1 Tổng quan

Ngoài các trung tâm vi xử lý họ x86, Intel[®] còn thiết kế và sản xuất các trung tâm Vi xử lý chuyên dụng phục vụ các mục đích đo lường và điều khiển tự động, phục vụ các ứng dụng đơn giản nhưng rất phổ biến khác, đặc biệt là trong các *Hệ thống nhúng (Embedded Systems)*. Các chip Vi xử lý loại này đã vượt ra ngoài khuôn khổ của một *trung tâm Vi xử lý đơn thuần*, trở thành một *máy vi tính (MicroComputer)*. Cũng có thể nhìn nhận rằng, các trung tâm Vi xử lý họ này là một *máy vi tính* thực thụ, nếu nhìn nhận chip này theo quan điểm kiến trúc máy tính của Von Neumann: Chip được trang bị thêm bộ nhớ chương trình (ROM hoặc EPROM) và bộ nhớ dữ liệu, cũng như các cổng vào/ra nối tiếp, vào/ra song song, v.v...

MCS - 51 là họ vi điều khiển của Intel[®]. Các nhà sản xuất khác như Siemens, AMD - Advanced Micro Device, Fujitsu và Philips được cấp phép làm các nhà cung cấp các chip của họ MCS - 51.

Vi mạch chủ yếu của họ MCS - 51 là chip μ C8051, linh kiện đầu tiên của họ này được đưa ra thị trường. Chip μ C8051 có các đặc trưng được tóm tắt như sau:

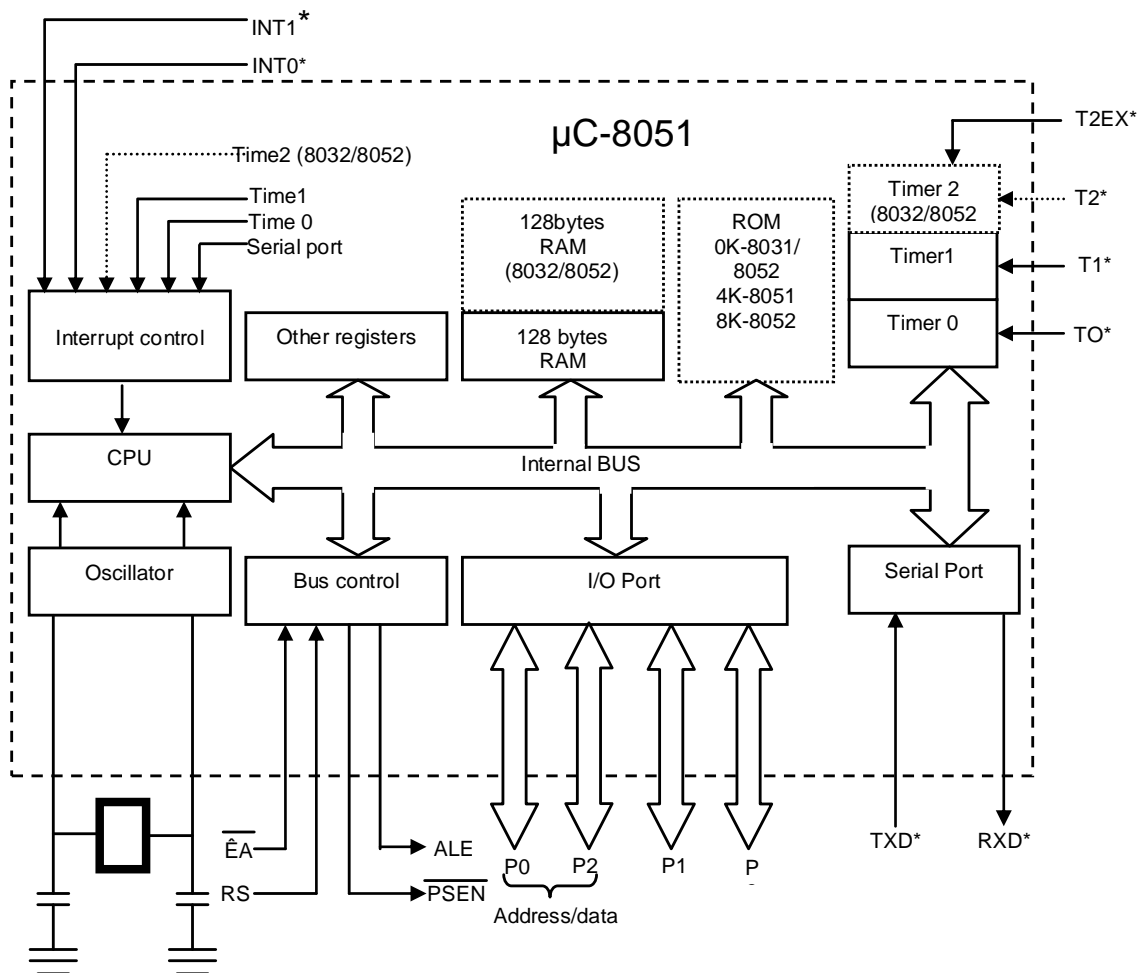
- 4 KB ROM và 128 byte RAM
- 4 port 8- bit, 32 lối vào/ra
- 2 bộ định thời (Timmer) 16 bit
- Mạch giao tiếp nối tiếp
- Không gian nhớ chương trình ngoài (mở rộng) 64K
- Không gian nhớ dữ liệu ngoài 64K
- Bộ xử lý bit (thao tác trên các bit riêng rẽ)
- 210 vị trí bit nhớ được định địa chỉ

– Thực hiện phép toán Nhân, Chia trong thời gian 4 μ s.

Các thành viên khác của họ MCS-51 có các tổ hợp ROM (EPROM), RAM trên chip với dung lượng khác nhau, bộ biến đổi tín hiệu tương tự-số và số-tương tự, và có thể có thêm bộ định thời thứ ba. Mỗi một chip của họ MCS-51 đều có phiên bản CMOS tiêu thụ công suất thấp.

Dưới đây là thông số cơ bản của một số μ C họ MCS-51:

Chip	Bộ nhớ chương trình trên chip	Bộ nhớ dữ liệu trên chip	Các bộ nhớ định thời
8051	4 K ROM	128 byte	2
8031	0 K	128 byte	2
8751	4 K EPROM	128 byte	2
8052	8 K ROM	256 byte	3
8032	0 K	256 byte	3
8752	8 K EPROM	256 byte	3



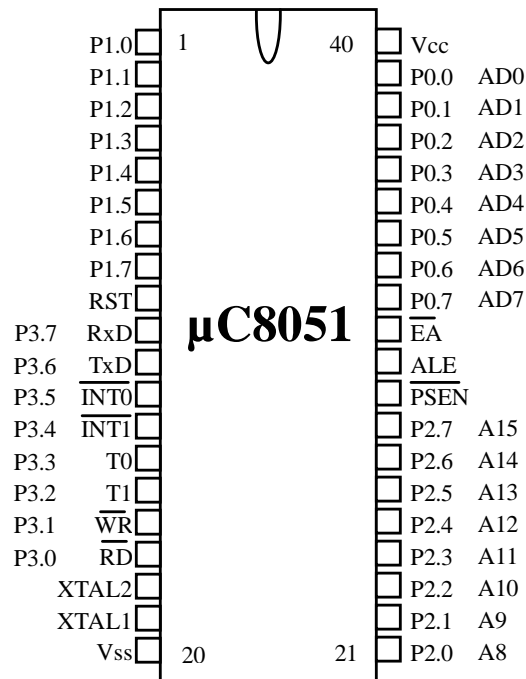
Hình II.29 Sơ đồ cấu trúc của chip μ C8051

Port 0 (các chân từ 32 đến 39) có hai công dụng. Trong các thiết kế tối thiểu, *port 0* được sử dụng làm nhiệm vụ vào/ra. Trong các thiết kế lớn hơn có bộ nhớ ngoài, *port 0* trở thành bus địa chỉ và bus dữ liệu dồn kênh.

Port 1 chỉ có một công dụng là vào/ra (các chân từ 1 đến 8), và dùng để giao tiếp với các thiết bị ngoại vi hoặc làm cổng vào/ra hoặc làm các lối vào cho mạch định thời thứ ba.

Port 2 (các chân từ 21 đến 28) có hai công dụng, hoặc làm nhiệm vụ vào/ra hoặc là byte địa chỉ cao của bus địa chỉ 16-bit cho các thiết kế có bộ nhớ chương trình ngoài hoặc các thiết kế có nhiều hơn 256 byte bộ nhớ dữ liệu ngoài.

Port 3 (các chân từ 10 đến 17) có hai công dụng. Khi không hoạt động vào/ra, các chân của *port 3* có nhiều chức năng riêng (mỗi chân có chức năng riêng liên quan đến các đặc trưng cụ thể của 8051).



Hình II.30 Sơ đồ nối chân Chip $\mu C8051$

II.4.2 Mô tả cấu trúc và chức năng

Hình II.28 cho ta sơ đồ chân của chip 8051. Chức năng tóm tắt của từng chân như sau : 32 trong số 40 chân của 8051 có công dụng vào/ra, tuy nhiên 24 trong 32 chân này có hai công dụng, mỗi chân có thể hoạt động vào/ra hoặc hoạt động như một chân của tín hiệu điều khiển hoặc như một chân địa chỉ/dữ liệu của bus địa chỉ/dữ liệu dồn kênh.

32 chân hình thành 4 *port* 8-bit. Với các thiết kế yêu cầu tối thiểu bộ nhớ ngoài hoặc các thành phần bên ngoài khác ta có thể sử dụng *port* này làm nhiệm vụ vào/ra. 8 chân của mỗi *port* có thể được sử dụng như một đơn vị giao tiếp song song với các thiết bị ngoại vi.

+ **Port 0**

Port 0 (các chân từ 32 đến 39) có hai công dụng. Trong các thiết kế tối thiểu, *port 0* được sử dụng làm nhiệm vụ vào/ra. Trong các thiết kế lớn hơn có bộ nhớ ngoài, *port 0* trở thành bus địa chỉ và bus dữ liệu dồn kênh.

+ **Port 1**

Port 1 chỉ có một công dụng là vào/ra (các chân từ 1 đến 8), và dùng để giao tiếp với các thiết bị ngoại vi hoặc làm chân vào/ra, hoặc làm các chân nhận tín hiệu vào của mạch định thời thứ ba.

+ **Port 2**

Port 2 (các chân từ 21 đến 28) có hai công dụng, hoặc làm nhiệm vụ vào/ra, hoặc 8 chân địa chỉ cao của bus địa chỉ 16-bit cho các thiết kế có bộ nhớ chương trình ngoài hoặc các thiết kế có nhiều hơn 256 byte bộ nhớ dữ liệu ngoài.

+ **Port 3**

Port 3 (các chân từ 10 đến 17) có hai công dụng. Khi không sử dụng cho mục đích vào/ra, các chân của *port 3* có các chức năng riêng (mỗi chức năng riêng liên quan đến các đặc trưng cụ thể của 8051).

+ **Chân cho phép truy nhập bộ nhớ chương trình PSEN**

8051 cung cấp 4 tín hiệu điều khiển bus. Tín hiệu PSEN (*Program store enable*) là tín hiệu ra trên chân 29. Đây là tín hiệu điều khiển cho phép truy xuất bộ nhớ chương trình ngoài, chân này thường nối với chân cho phép ra OE (*output enable*) của EPROM (hoặc ROM), cho phép đọc các byte lệnh.

Tín hiệu PSEN ở logic 0 trong suốt thời gian tìm nạp lệnh (Instruction Fetch). Các mã nhị phân của chương trình hay *opcode* (mã thao tác) được đọc từ EPROM, qua bus dữ liệu và được chốt vào thanh ghi lệnh IR của 8051 để được giải mã.

Khi thực thi một chương trình chứa ở ROM nội, PSEN được duy trì ở mức logic không tích cực (logic 1).

+ **Chân cho phép chốt địa chỉ ALE**

8051 sử dụng chân 30, chân cho phép chốt địa chỉ ALE (*address latch enable*) để giải dòn kênh (*demultiplexing*) bus dữ liệu và bus địa chỉ. Khi *port 0* được sử dụng làm bus địa chỉ/dữ liệu dòn kênh, chân ALE đưa ra tín hiệu để chốt địa chỉ (byte thấp của địa chỉ 16-bit) vào một thanh ghi ngoài trong suốt 1/2 đầu của chu kỳ bộ nhớ (memory cycle). Sau khi địa chỉ đã được chốt các chân của *port 0* sẽ làm nhiệm vụ vào/ra dữ liệu trong 1/2 thứ hai của chu kỳ bộ nhớ.

+ **Chân truy xuất ngoài EA**

Lối vào EA (chân 31) có thể nối với 5V (logic 1) hoặc nối với GND (logic 0). Nếu chân này nối lên 5V, 8051/8052 thực thi chương trình trong ROM nội (Chương trình thường nhỏ hơn 4K/8K). Nếu chân này nối với GND (chân PSEN ở logic 0), CPU thực thi chương trình chứa ở bộ nhớ ngoài. Đối với 8031/8032 chân EA phải ở logic 0 vì chúng không có bộ nhớ chương trình trên chip. Nếu chân EA ở logic 0 đối với 8051/8052, ROM nội ở bên trong chip được vô hiệu hoá và chương trình thực thi chứa ở ROM bên ngoài.

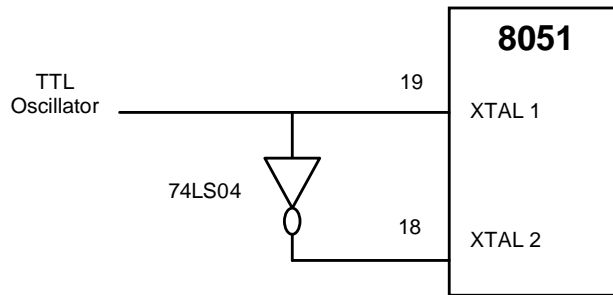
Các phiên bản EPROM của 8051 còn sử dụng chân EA làm chân nhận xung điện áp 21V cho việc lập trình EPROM nội.

+ **Chân RESET (RST)**

Lỗi vào RST (chân 9) là lỗi vào tái khởi động (master reset) của 8051 dùng để thiết lập trạng thái ban đầu cho toàn hệ thống. Khi lỗi này được treo ở logic 1 tối thiểu hai chu kỳ máy, các thanh ghi bên trong của 8051 được nạp các giá trị thích hợp cho việc khởi động lại hệ thống.

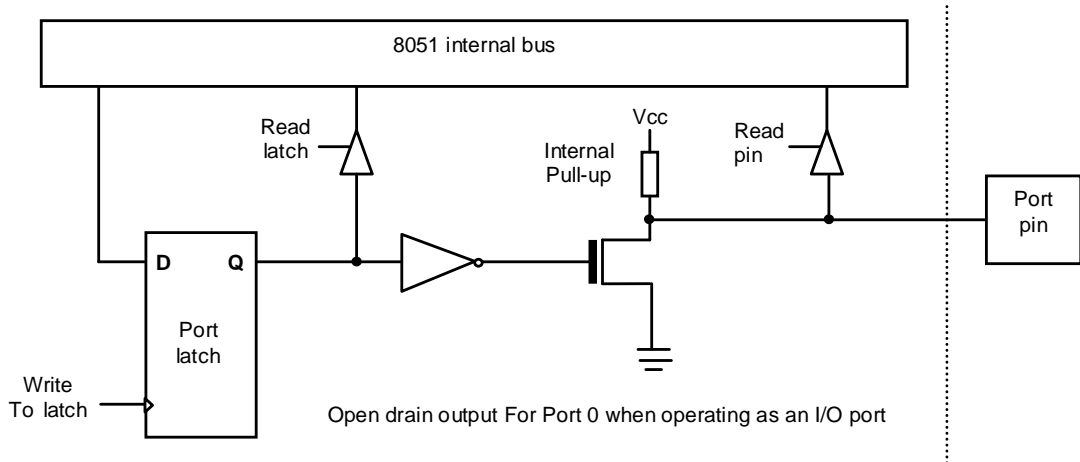
+ **Các chân XTAL1&XTAL2**

Mạch dao động bên trong chip 8051 được ghép với thạch anh bên ngoài ở hai chân XTAL1 và XTAL2 (chân 18 và 19). Tần số danh định của thạch anh là 12MHz cho hầu hết các chip của họ MCS-51 (Riêng 80C31BH-1 sử dụng thạch anh 16MHz bên trong, mạch dao động trong chip không cần thạch anh bên ngoài). Một nguồn xung clock TTL có thể được nối với chân XTAL1 và XTAL2.



Hình 2.31 8051 ghép với mạch dao động TTL bên ngoài

+ **Cấu trúc của Port vào/ra.**



Hình II.32 Cấu trúc của port vào ra

8051 internal bus: Bus nội của 8051

Read pin: chân port

Read latch: bộ chốt phục vụ đọc

Port latch: Bộ chốt của port

Internal pull-up: Mạch pull-up

Write to latch: Ghi vào bộ chốt

Sơ đồ mạch điện bên trong của port vào/ra được vẽ đơn giản như hình II.23, việc ghi dữ liệu đến 1 chân của Port thực chất là nạp dữ liệu vào bộ

chốt của *port*, lối ra Q của bộ chốt điều khiển một Transistor trường và Transistor này nối với chân của port. Khả năng Fanout của các port 1, 2 và 3 là 4, tải của vi mạch TTL loại Schottky công suất thấp (LS).

Ta thấy có 2 khả năng: "đọc bộ chốt" và "đọc chân *port*". Các lệnh yêu cầu thao tác đọc-sửa-ghi đọc bộ chốt để tránh nhầm lẫn mức điện áp do sự kiện dòng tải tăng. Các lệnh nhập 1 bit của *port* (như MOV C,P1.5) đọc chân *port*.

+ **Tổ chức bộ nhớ.**

Hầu hết các bộ vi xử lý (CPU) đều có không gian nhớ chung cho dữ liệu và chương trình, các chương trình được lưu trên đĩa và nạp vào RAM để thực thi vậy thì cả hai dữ liệu và chương trình đều lưu trữ trong RAM.

Các chip vi điều khiển hiếm khi được sử dụng giống như các CPU trong các hệ máy tính, thay vào đó chúng được làm thành phần trung tâm trong các thiết kế hướng điều khiển, trong đó có bộ nhớ dung lượng giới hạn, không có ổ đĩa và hệ điều hành. Chương trình điều khiển thường trú trong ROM.

8051 có không gian và bộ nhớ riêng cho chương trình và dữ liệu được tích hợp ở bên trong chip, ta có thể mở rộng bộ nhớ chương trình và bộ nhớ dữ liệu bằng cách sử dụng các chip nhớ bên ngoài với dung lượng tối đa là 64K cho bộ nhớ chương trình và 64K cho bộ nhớ dữ liệu.

Bộ nhớ tích hợp trong chip bao gồm ROM (Chỉ có ở 8051/8052) và RAM. RAM trên chip bao gồm vùng RAM đa chức năng và vùng RAM với từng bit được định địa chỉ (gọi là vùng RAM định địa chỉ bit, các Bank) và các thanh ghi chức năng đặc biệt SFR (Special Function Register). Hai đặc tính đáng lưu ý là:

(a) Các thanh ghi và các *port* vào/ra được định địa chỉ theo kiểu ánh xạ bộ nhớ (memory mapped) và được truy xuất như một vị trí trong bộ nhớ.

(b) Vùng *stack* thường trú trên RAM trong chip (RAM nội) thay vì ở trong RAM ngoài đối với các bộ vi xử lý.

+ **Vùng RAM đa mục đích.**

Vùng RAM đa mục đích có 80 byte đặt ở địa chỉ từ 30H đến 7FH, bên dưới vùng này từ địa chỉ 00H đến 2FH là vùng nhớ có thể được sử dụng tương tự. Bất kỳ vị trí nhớ nào trong Vùng RAM đa mục đích đều có thể được truy xuất tự do bằng cách định địa chỉ trực tiếp hoặc gián tiếp.

+ **Vùng RAM định địa chỉ bit.**

8051 chứa 210 bit được định địa chỉ trong đó 128 bit chứa trong các byte ở địa chỉ 20H đến 2FH (16 byte x 8 bit = 128 bit) phần còn lại chứa trong các thanh ghi đặc biệt. Truy xuất các bit riêng rẽ thông qua phần mềm là một đặc trưng mạnh của hầu hết các bộ vi điều khiển. Các bit có thể được

Set, Reset, AND, OR, v.v... bằng một lệnh. Hầu hết các bộ vi xử lý có một chuỗi lệnh đọc-sửa-ghi để nhận được cùng một kết quả. Ngoài ra 8051 còn có các *port* vào/ra có thể định địa chỉ từng bit, điều này làm đơn giản việc giao tiếp bằng phần mềm với các thiết bị vào/ra đơn bit.

+ **Các thanh ghi.**

32 vị trí thấp nhất của bộ nhớ nội được sử dụng như những thanh ghi. Các lệnh của 8051 hỗ trợ 8 thanh ghi từ R0 đến R7 thuộc băng 0 (bank 0) đây là băng mặc định sau khi reset hệ thống. Các thanh ghi này được đánh địa chỉ từ 00H đến 07H.

Các lệnh sử dụng các thanh ghi từ R0 đến R7 là các lệnh ngắn và thực hiện nhanh hơn so với các lệnh tương đương sử dụng kiểu định địa chỉ trực tiếp. Các giá trị dữ liệu được sử dụng nhiều nên chứa ở một trong các thanh ghi này. Băng thanh ghi đang được sử dụng được gọi là băng thanh ghi tích cực. Băng thanh ghi tích cực có thể được thay đổi bằng cách thay đổi các bit chọn băng trong từ trạng thái chương trình PSW .

+ **Các thanh ghi chức năng đặc biệt (SFR)**

Các thanh ghi của hầu hết các bộ vi xử lý đều được truy xuất tường minh bởi một tập lệnh. Thao tác được xác định rõ ràng trong *opcode* của lệnh.

Các thanh ghi của 8051 được cấu thành từ một phần của RAM trên chip, do vậy mỗi thanh ghi cũng có một địa chỉ. Cũng như các thanh ghi từ R0 đến R7, ta có 21 thanh ghi chức năng đặc biệt SFR chiếm phần trên của RAM nội từ địa chỉ 80H đến FFH .

+ **Cờ nhớ.**

Cờ nhớ CY (CARRY FLAG). Công dụng truyền thống trong các phép toán số học, được set bằng 1 nếu có số nhớ từ phép cộng bit 7 hoặc có số mượn mang đến bit 7.

+ **Cờ nhớ phụ.**

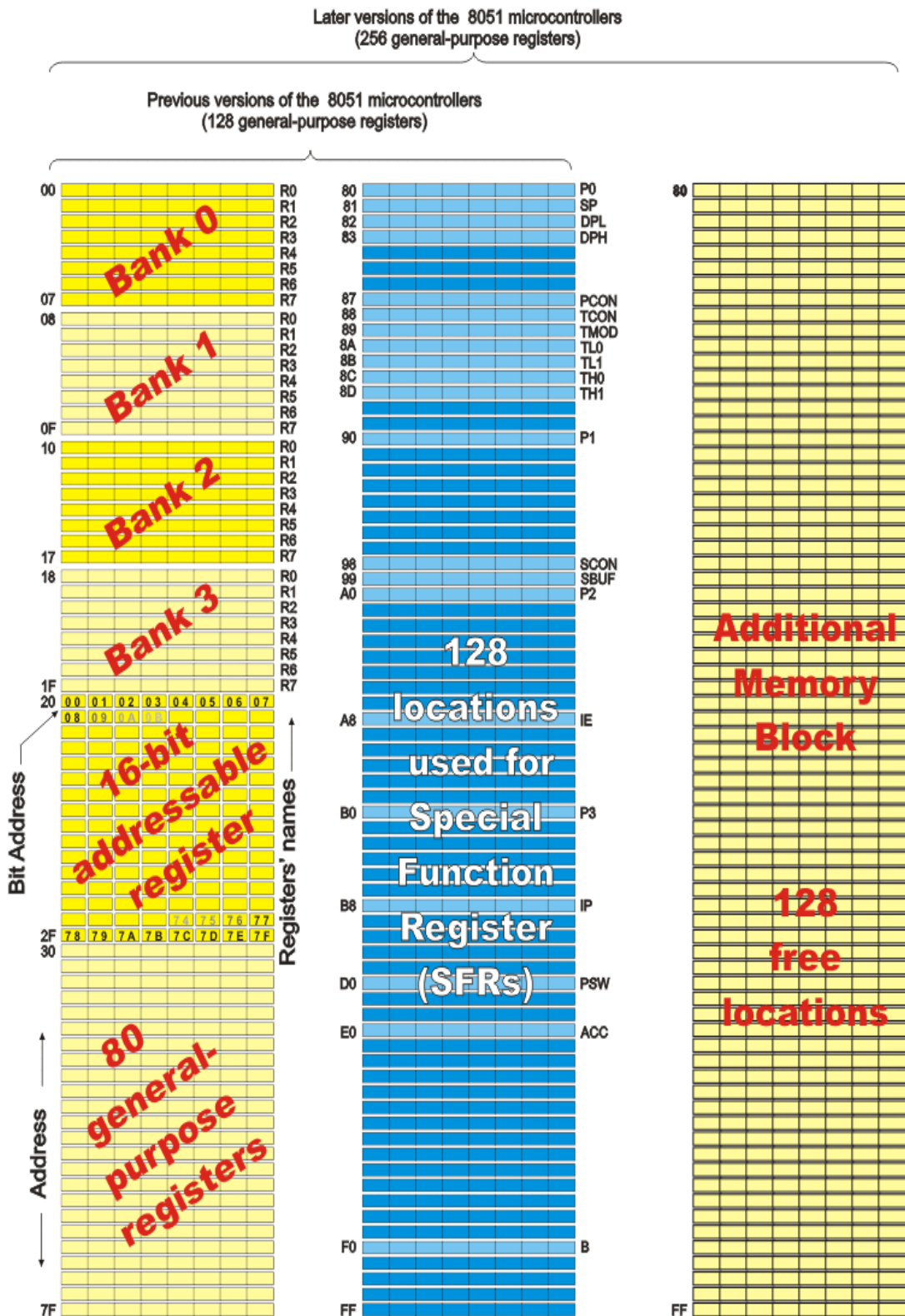
Khi cộng các giá trị BCD, cờ nhớ phụ AC (auxiliary carry Flag) được set bằng 1 nếu có một số nhớ được tạo từ bit 3 chuyển sang bit 4 hoặc nếu kết quả trong 4 bit thấp nằm trong vùng từ 0AH đến 0FH. Nếu các giá trị được cộng là giá trị BCD, lệnh cộng phải được tiếp theo bởi lệnh DA A (hiệu chỉnh thập phân thanh chứa A) để đưa các giá trị kết quả lớn hơn 9 về giá trị đúng.

+ **Cờ 0**

Đây là cờ có nhiều mục đích dành cho các ứng dụng của người lập trình.

+ **Các bit chọn dãy thanh ghi**

Các bit chọn dãy thanh ghi RS0, RS1 dùng để xác định dãy thanh ghi tích cực. Các bit này được xoá sau khi có thao tác reset hệ thống và đổi mức logic bởi phần mềm khi cần.



Hình II.33. Tổ chức bộ nhớ RAM trong họ $\mu C8051$

+ **Cờ tràn**

Cờ tràn OV (overFlow Flag) được reset bằng 1 sau phép toán cộng hoặc trừ nếu có xuất hiện một sự tràn số học. Khi các số có dấu được cộng hoặc được trừ, phần mềm có thể kiểm tra bit tràn OV để xác định xem kết quả có nằm trong khoảng cho phép hay không.

+ **Cờ chặn lẻ**

Bit chặn lẻ P tự động được set bằng 1 hay xoá bằng 0 ở mỗi chu kỳ máy để thiết lập kiểm tra chặn lẻ cho thanh chứa A. Số các bit 1 trong thanh chứa cộng với bit P luôn là số chẵn. Nếu thanh chứa có nội dung 10101101B, bit P sẽ là 1 để có số bit 1 là 6. Bit chặn lẻ được sử dụng nhiều để kết hợp với các chương trình vào/ra nối tiếp trước khi truyền dữ liệu hoặc kiểm tra chặn lẻ sau khi truyền dữ liệu.+

+ **Từ trạng thái chương trình PSW**

Bit	Ký hiệu	Địa chỉ	Mô tả bit
PSW.7	CY	D7H	Cờ nhớ
PSW.6	AC	D7H	Cờ nhớ phụ
PSW.5	F0	D6H	Cờ 0
PSW.4	RS1	D5H	Chọn dãy thanh ghi (bit 1)
PSW.3	RS0	D4H	Chọn dãy thanh ghi bit 0
			00 = Bank 0: địa chỉ từ 00H đến 07H
			01 = Bank 1: địa chỉ từ 08H đến 0FH
			10 = Bank 2: địa chỉ từ 10H đến 17H
			11 = Bank 3: địa chỉ từ 18H đến 1FH
PSW2	OV	D3H	Cờ tràn
PSW1	-	D1H	Dự trữ
PSW0	P	D0H	Kiểm tra chặn lẻ

+ **Thanh ghi B**

Thanh ghi B ở địa chỉ F0H được dùng chung với thanh chứa A trong các phép toán nhân, chia. Lệnh MUL AB nhân 2 số 8-bit không dấu chứa trong A và B và chứa kết quả vào cặp Thanh ghi B:A (Thanh chứa A chứa byte thấp và thanh chứa B chứa byte cao của tích số).

Lệnh chia DIV AB chia A bởi B, thương số chứa trong thanh chứa A và số dư chứa trong Thanh ghi B. Thanh ghi B còn được xử lý như một thanh ghi nháp. Các bit được định địa chỉ của thanh ghi B có địa chỉ từ F0H đến F7H.

+ **Con trỏ Stack**

Con trỏ *stack* SP (stack pointer) là một thanh ghi 8-bit ở địa chỉ 81H. SP chứa địa chỉ của dữ liệu hiện đang ở đỉnh của *stack*. Các lệnh liên quan đến *stack* bao gồm dữ liệu cất vào *stack* và lệnh lấy dữ liệu ra khỏi *stack*. Việc cất vào *stack* làm tăng SP trước khi ghi dữ liệu và việc lấy dữ liệu ra khỏi *stack* sẽ giảm SP. Vùng *stack* của 8051 được giữ trong RAM nội và được giới hạn đến các địa chỉ truy xuất được bởi kiểu định địa chỉ gián tiếp.

+ ***Con trỏ dữ liệu DPTR***

Con trỏ dữ liệu DPTR (data pointer) dùng để truy xuất bộ nhớ bên ngoài hoặc bộ nhớ dữ liệu ngoài. DPTR là một thanh ghi độ dài là 16 bit có địa chỉ là 82H (DPL, byte thấp) và 83H (DPH, byte cao).

+ ***Các thanh ghi port.***

Các *port* vào/ra của 8051 bao gồm *port* 0 tại địa chỉ 80H, *port* 1 tại địa chỉ 90H, *port* 2 tại địa chỉ A0H, và *port* 3 tại địa chỉ B0H, các *port* 0, 2 và 3 không được dùng để vào/ra nếu ta sử dụng thêm bộ nhớ ngoài hoặc nếu có một số đặc tính của 8051 được sử dụng như là ngắt, *port* nối tiếp,... P1.2 đến P1.7, ngược lại, luôn là các đường vào/ra đa mục đích hợp lệ.

Tất cả các *port* đều được định địa chỉ từng bit nhằm cung cấp khả năng giao tiếp mạnh.

+ ***Các thanh ghi định thời.***

8051 có 2 bộ đếm định thời (time/counter) 16-bit để định các khoảng thời gian hoặc để đếm các sự kiện. Bộ định thời 0 có địa chỉ là 8AH (TL0, byte thấp) và 8BH (TL1, byte cao); bộ định thời 1 có địa chỉ 8CH (TH0, byte thấp) và 8DH (TH1, byte cao).

Hoạt động của bộ định thời được thiết lập bởi thanh ghi chế độ định thời TMOD (time mode register) ở địa chỉ 89H và thanh ghi điều khiển định thời TCON (time control register) ở địa chỉ 88H. Chỉ có TCON được định địa chỉ từng bit.

+ ***Các thanh ghi của port nối tiếp.***

Bên trong 8051 có một *port* nối tiếp để truyền thông với các thiết bị nối tiếp như các thiết bị đầu cuối hoặc modem, hoặc để giao tiếp với các IC khác có mạch giao tiếp nối tiếp (như các thanh ghi dịch).

Một thanh ghi được gọi là bộ đệm dữ liệu nối tiếp SBUF (serial data buffer) ở địa chỉ 99H lưu dữ liệu truyền đi và dữ liệu nhận về. Việc ghi lên SBUF sẽ nạp dữ liệu để truyền và việc đọc SBUF sẽ lấy dữ liệu đã nhận được.

Các chế độ hoạt động khác nhau được lập trình thông qua thanh ghi điều khiển *port* nối tiếp SCON (serial port control register) ở địa chỉ 98H, thanh ghi này được định địa chỉ từng bit.

+ Các thanh ghi ngắt

8051 có một cấu trúc ngắt với 2 mức ưu tiên và 5 nguyên nhân ngắt. (5 source, 2 priority level interrupt structure). Các ngắt bị vô hiệu hoá sau khi reset hệ thống và sau đó được cho phép bằng cách ghi vào thanh ghi cho phép ngắt IE (interrupt enable register) ở địa chỉ A8H. Mức ưu tiên của các ngắt được thiết lập qua thanh ghi ưu tiên ngắt IP (interrupt priority register) ở địa chỉ B8H. Cả 2 thanh ghi này đều được định địa chỉ từng bit.

+ Thanh ghi điều khiển nguồn

Thanh ghi điều khiển nguồn PCON (power control register) có địa chỉ 87H chứa các bit điều khiển.

Bit SMOD tăng gấp đôi tốc độ baud của *port* nối tiếp. Khi *port* này hoạt động ở các chế độ 1, 2 hoặc 3, các bit 4, 5 và 6 của PCON không được định nghĩa. Các bit 2 và 3 là các bit cờ đa mục đích dành cho các ứng dụng của người sử dụng.

Các bit điều khiển nguồn, nguồn giảm PD và nghỉ IDL, hợp lệ cho tất cả chip thuộc họ MCS-51, nhưng chỉ được thực hiện trong các phiên bản CMOS của MCS-51. TCON không được định địa chỉ bit.

Chế độ nguồn giảm.

Lệnh thiết lập bit PD bằng 1 sẽ là lệnh sau cùng được thực hiện trước khi đi vào chế độ nguồn giảm:

- (1) Mạch giao động trên chip ngừng hoạt động
- (2) Mọi chức năng ngừng hoạt động
- (3) Nội dung của RAM trên chip được duy trì
- (4) Các phần port duy trì mức logic hiện tại của chúng
- (5) ALE và PSEN giữ được ở mức thấp. Chỉ ra khỏi chế độ này bằng cách reset hệ thống.

Trong suốt thời gian ở chế độ nguồn giảm, Vcc có điện áp là 2v. Cần phải giữ cho Vcc không thấp hơn sau khi đạt được chế độ nguồn giảm và cần phục hồi Vcc bằng 5V tối thiểu 10 chu kỳ dao động trước khi chân RST đạt mức thấp lần nữa.

Bit	Ký hiệu	Mô tả
7	SMOD	bit tăng gấp đôi tốc độ baud, bit này khi set làm cho tốc độ baud tăng 2 ở các chế độ 1,2 và 3 của <i>port</i> nối tiếp
6	-	Không định nghĩa
5	-	Không định nghĩa
4	-	Không định nghĩa

3	CF1	Bit cờ đa mục đích 1
2	CF0	Bit cờ đa mục đích 2
1	FD	Nguồn giảm; thiết lập để tích cực chế độ nguồn giảm, chỉ ra khỏi chế độ bằng reset
0	IDL	Chế độ nghỉ; thiết lập để tích cực chế độ ghi, chỉ ra khỏi chế độ bằng một ngắt hoặc reset hệ thống.

Chế độ nghỉ.

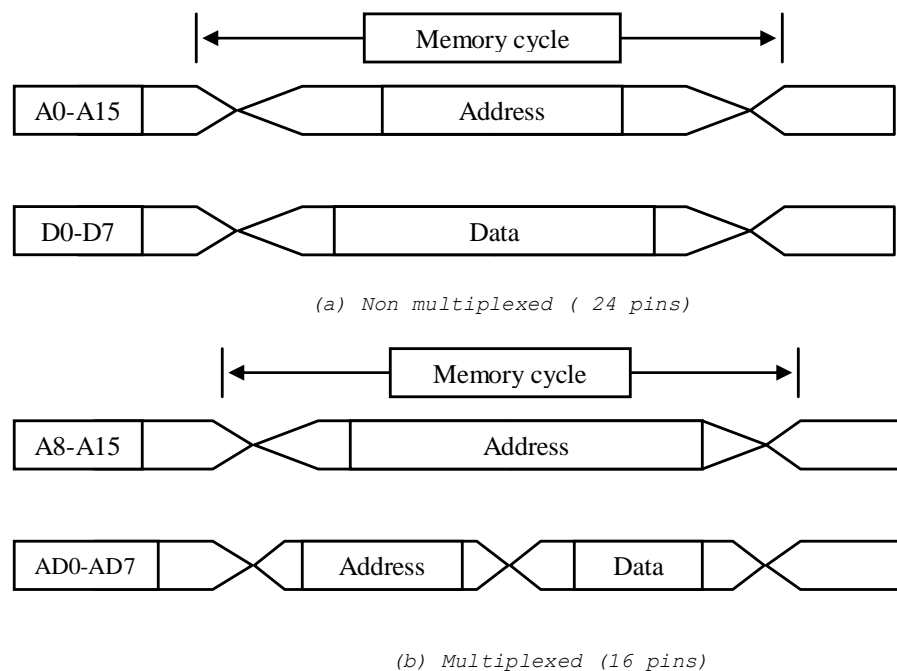
Lệnh thiết lập bit IDL bằng 1 sẽ là lệnh sau cùng được thực thi trước khi đi vào chế độ nghỉ. Ở chế độ nghỉ, tín hiệu *clock* nội được khoá không cho đến CPU nhưng không khoá đối với các chức năng ngắt, định thời và *port* nối tiếp. Trạng thái của CPU được duy trì và nội dung của tất cả các thanh ghi cũng được giữ không thay đổi.

Các chân *port* cũng được duy trì các mức logic của chúng. ALE và PSEN được giữ ở mức cao.

Chế độ nghỉ kết thúc bằng cách cho phép ngắt hoặc bằng cách reset hệ thống. Cả hai cách vừa nêu đều xoá bit IDL.

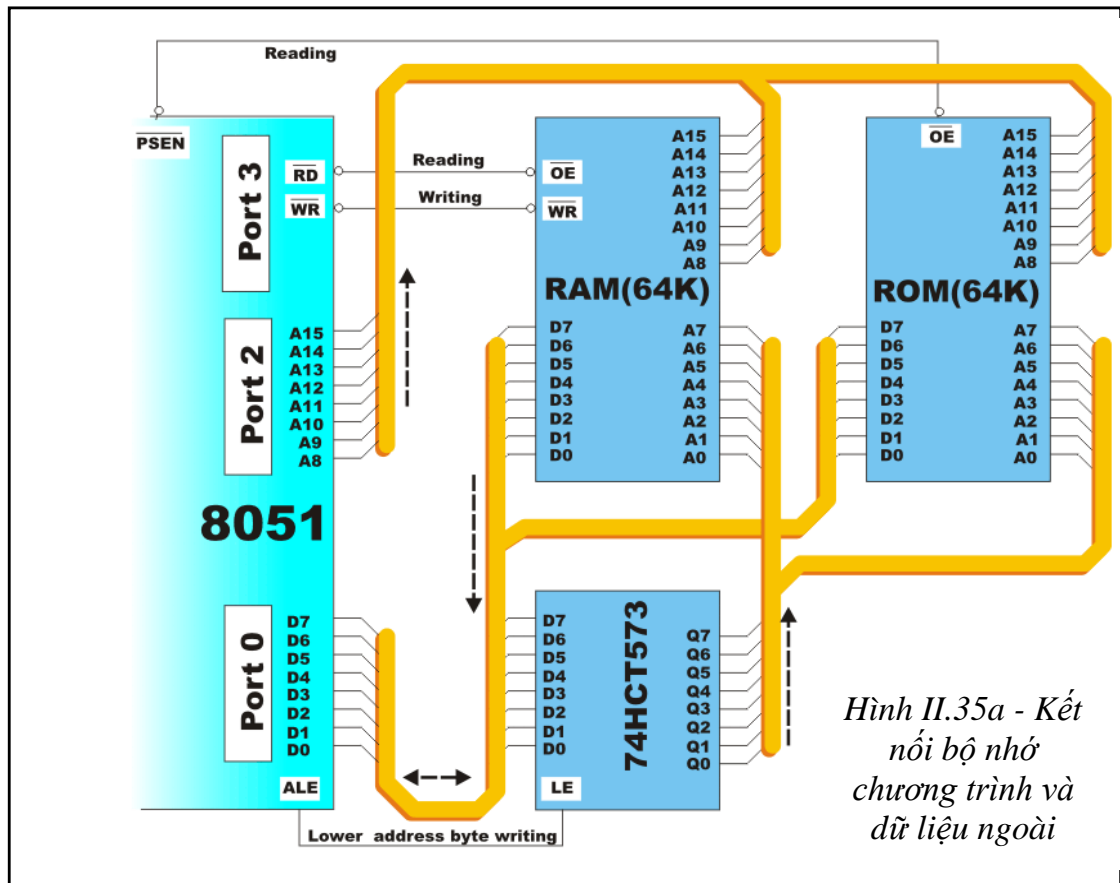
+ Bộ nhớ ngoài

Các bộ vi xử lý họ MCS-51 có khả năng mở rộng các tài nguyên trên chip (bộ nhớ, I/O, v.v...) để tránh hiện tượng cổ chai trong thiết kế. Cấu trúc của MCS-51 cho khả năng mở rộng không gian bộ nhớ chương trình đến 64K ROM ngoài và không gian bộ nhớ dữ liệu đến 64K RAM ngoài khi cần thiết.



Hình II.34 Đa hợp địa chỉ (byte thấp) và Bus dữ liệu
(a) không dồn kênh (24 chân), (b) dồn kênh (16 chân)

Các IC giao tiếp ngoại vi cũng có thể được thêm vào để mở rộng khả năng vào/ra. Chúng trở thành một phần của không gian bộ nhớ dữ liệu ngoài bằng cách sử dụng cách định địa chỉ kiểu I/O ánh xạ bộ nhớ. Khi bộ nhớ ngoài được sử dụng, *port 0* làm nhiệm vụ của port vào/ra. *Port* này trở thành bus địa chỉ (A0-A7) và bus dữ liệu (D0-D7) đơn kênh. Lỗi ra ALE chốt byte thấp của địa chỉ ở thời điểm bắt đầu mỗi một chu kỳ bộ nhớ ngoài. *Port 2* thường (nhưng không phải luôn luôn) được dùng làm byte cao của bus địa chỉ.



Hình II.35a - Kết nối bộ nhớ chương trình và dữ liệu ngoài

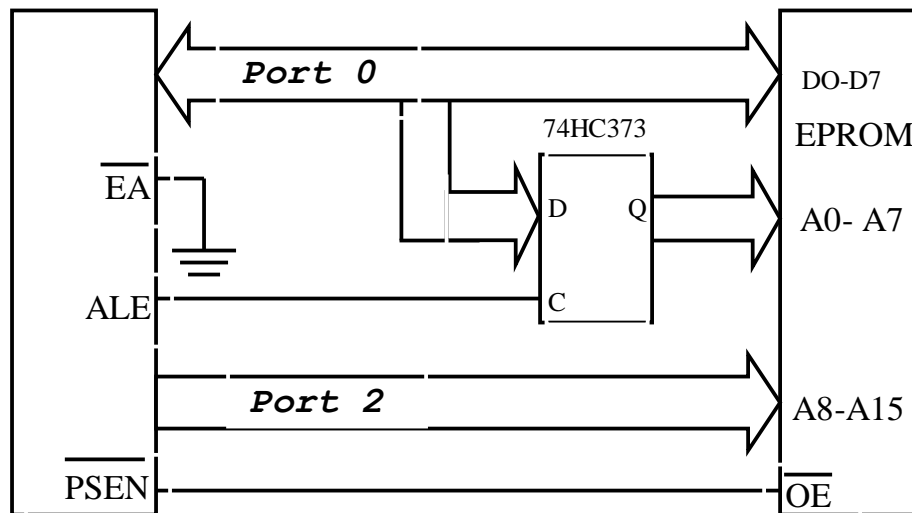
+ Truy xuất bộ nhớ chương trình ngoài

Bộ nhớ chương trình ngoài là bộ nhớ chỉ đọc, được cho phép truy xuất bởi tín hiệu PSEN. Khi có một EPROM ngoài được sử dụng, cả hai *port 0* và *port 2* không được sử dụng cho mục đích vào/ra. Kết nối 8051 với bộ nhớ ngoài EPROM được trình bày ở hình II.33a, II.13b

Một chu kỳ máy của 8051 có 12 xung nhịp. Nếu bộ giao động trên chip có tần số 12MHz, một chu kỳ máy dài 1 μ sec. Trong một chu kỳ máy điển hình, ALE có 2 xung và 2 byte của lệnh được đọc từ bộ nhớ chương trình (nếu lệnh chỉ có một byte, byte thứ hai được loại bỏ).

Bộ nhớ dữ liệu ngoài là bộ nhớ đọc/ghi được cho phép bởi các tín hiệu \overline{RD} và \overline{WR} ở các chân của P3. Lệnh dùng để truy xuất bộ nhớ dữ

liệu ngoài là MOVX, sử dụng hoặc con trỏ dữ liệu 16-bit DPTR hoặc R0, R1 làm thanh ghi chứa địa chỉ.



Hình II.35b - Kết nối 8051 với bộ nhớ chương trình ngoài

RAM có thể giao tiếp với 8051 theo cách như EPROM ngoại trừ đường \overline{RD} nối với đường cho phép xuất (\overline{OE}) của RAM và \overline{WR} nối với đường ghi (\overline{WR}) của RAM. Các kết nối với bus dữ liệu và bus địa chỉ giống như EPROM. Bằng cách sử dụng các *port 0* và *port 2* và các chip nhớ, ta có dung lượng RAM và ROM ngoài lên đến 64K (Hình II.13a).

II.4.3 Lập trình cho $\mu\text{C}8051$

Để lập trình cho 8051, người lập trình cần nắm thật vững cách tổ chức rất hữu hiệu nhưng tương đối phức tạp của bộ nhớ RAM tích hợp trong chip. Không đơn thuần đóng vai trò bộ nhớ dữ liệu trong MCS51, nó còn sử dụng một phần bộ nhớ RAM để làm thanh ghi đa năng và thanh ghi với các chức năng đặc biệt.

Tồn tại chương trình Assembler riêng cho họ MCS51, lập trình hợp ngữ tương đương như lập trình hợp ngữ cho họ 80x86. Điểm mạnh tương ứng là tồn tại một phiên bản ngôn ngữ C cho MCS51, tạo điều kiện cho những ai đã quen với lập trình C có thể tạo các phần mềm ứng dụng để cài đặt vào trong bộ nhớ ROM của MCS 51 cho những ứng dụng thực tế.

Bạn đọc có thể tham khảo tài liệu [1] về lập trình cho $\mu\text{C}8051$ được nêu ở cuối cuốn giáo trình này.

II.4.4 Các khả năng ứng dụng của $\mu\text{C}8051$

Thông thường, các trung tâm Vi xử lý được dùng để xây dựng nên các máy tính. Riêng các trung tâm của Single Chip Microcomputer, do những cấu trúc đặc trưng và tính năng kỹ thuật, được ứng dụng nhiều

trong các thiết kế nhỏ, đặc biệt là trong các hệ thống nhúng, với số thành phần phụ trợ thêm vào tối thiểu nhất. Nhờ cấu trúc và khả năng cài đặt các chương trình ứng dụng ngay trong bộ nhớ ROM hoặc bộ nhớ Flash tích hợp sẵn, các hướng và các ứng dụng cụ thể của họ Vi xử lý này chủ yếu tập trung vào các mục đích gia dụng và dân dụng. Thống kê một số lĩnh vực ứng dụng của các trung tâm Vi xử lý họ này được liệt kê trong bảng sau.

Trong gia đình	Đồ điện gia dụng Thiết bị đàm thoại Điện thoại Hệ thống an toàn Mở đóng cửa Trả lời tự động Máy FAX	Máy tính gia đình TV Truyền hình cáp VCR Camera Điều khiển từ xa Trò chơi điện tử	Nhạc cụ điện tử Máy khâu Điều khiển ánh sáng Máy nhắn tin v.v...
Thiết bị văn phòng	Điện thoại Máy tính Hệ thống an toàn	Máy FAX Lò vi sóng Photocopy	Máy in Laze Máy in màu Máy nhắn tin
Tự động hoá	Máy tính hành trình Điều khiển động cơ	Đo lường Truyền tin...	Điều hoà không khí v.v...

II.5. Vài nét về quy trình chế tạo chip CPU

Chỉ bằng đầu ngón tay nhưng chính những chip bán dẫn nhỏ bé này, như bộ vi xử lý Intel Core 2 Duo, đã thay đổi cách thức con người sống, làm việc và giải trí.

Ngày nay, chip silicon hiện diện ở khắp nơi. Chúng gia tăng sức mạnh của mạng Internet, cho phép tạo nên cuộc cách mạng về điện toán di động, tự động hóa các nhà máy, nâng cao năng lực của điện thoại và làm giàu lĩnh vực giải trí gia đình.

Việc sản xuất những chip như thế là một kỳ công. Công nghệ chế tạo của Intel đã tạo ra những mạch điện với kích thước chỉ bằng một phần nghìn sợi tóc trên miếng silicon. Loại chip phức tạp nhất - bộ vi xử lý - có thể chứa hàng trăm triệu, thậm chí là hàng tỷ bóng bán dẫn (transistor) được kết nối với nhau bằng những dây dẫn mỏng chế tạo từ đồng. Mỗi transistor có vai trò như một công tắc đóng/mở, điều khiển dòng điện chạy qua để gửi, nhận, và xử lý thông tin chỉ trong vòng một phần triệu giây.

II.5.1 Thiết kế

Việc sản xuất chip silicon bắt đầu bằng một thiết kế, hay bản vẽ. Ở giai đoạn này, Intel xem xét rất nhiều tham số như: Chọn kiểu chip nào và tại sao? Số lượng bóng bán dẫn có thể được tích hợp trên chip đó? Kích

cỡ tối ưu của chip là gì? Công nghệ nào sẽ được sử dụng để sản xuất chip? Khi nào thì chip phải được đưa ra thị trường? Chip đó sẽ được sản xuất và kiểm tra tại đâu?

Để trả lời những câu hỏi trên, Intel làm việc với các khách hàng, công ty phần mềm cũng như nhân viên tiếp thị, chế tạo và kiểm tra của Intel. Nhóm thiết kế lấy tham số đầu vào và bắt đầu xác định các tính năng của chip và thiết kế nó.

Khi bản chỉ tiêu kỹ thuật của chip đã được xác lập, Intel tạo ra *sơ đồ logic* - cách biểu diễn bằng ký hiệu của hàng triệu bóng bán dẫn và các kết nối điều khiển dòng điện đi qua toàn bộ chip. Khi giai đoạn này kết thúc, nhà thiết kế tạo các biểu diễn vật lý của mỗi lớp trong chip cùng các bóng bán dẫn của nó. Các mẫu giống như khuôn tô này (còn gọi là *Mặt nạ*) được sử dụng để che một số khu vực của chip khỏi tia cực tím trong quá trình sản xuất *in quang khắc*. Để hoàn tất thiết kế, kiểm tra và mô phỏng, Intel sử dụng máy workstation chạy chương trình thiết kế được máy tính hỗ trợ (CAD). Công đoạn này mất khoảng 2 năm với vài trăm người làm việc liên tục.

II.5.2 Chế tạo

Quá trình tạo chip bán dẫn được gọi là *chế tạo* và các nhà máy nơi chip được sản xuất gọi là *nhà máy chế tạo*, hay gọi tắt là *Fab*. "Phụ gia" xây dựng chip thay đổi tùy thuộc mục đích sử dụng dự kiến của nó. Intel sử dụng nhiều thành phần và thực hiện tới 300 bước, trong đó có hóa chất, khí ga hoặc ánh sáng để hoàn tất quá trình chế tạo.

Bắt đầu từ cát

Tất cả được bắt đầu với *silic*. Intel chế tạo chip theo từng mẻ trên các tấm *wafer* được làm từ silic tinh khiết - thành phần chính trong cát ở bờ biển. Hãng này dùng silic bởi vì nó là chất bán dẫn tự nhiên dễ bị oxy hóa. Không giống chất cách điện như thủy tinh (luôn ngăn dòng điện đi qua) hay chất dẫn điện như đồng (luôn cho phép dòng điện đi qua), bạn có thể biến silic thành chất dẫn điện hoặc cách điện.



Silic, nguyên tố hóa học phổ biến thứ hai trên trái đất sau oxy, được sử dụng vì nó là chất bán dẫn tự nhiên.

Để sản xuất tấm wafer, silic được xử lý hóa học nhằm đạt được độ tinh khiết đến 99,9999%. Silic đã tinh lọc được nung chảy và trở thành *thỏi hình trụ*. Những thỏi silic đó được cắt lát mỏng thành những tấm wafer và được đánh bóng cho đến khi chúng có bề mặt hoàn hảo, nhẵn bóng như gương. Khi mới bắt đầu sản xuất chip, Intel dùng các tấm wafer 2 inch còn hiện nay họ sử dụng wafer 300 mm (12 inch) và 200 mm (8 inch) để đạt sản lượng cao hơn và giảm được chi phí.

Cấu trúc nhiều lớp

Intel dùng quy trình "in" quang khắc để hình thành nhiều lớp bóng bán dẫn và dẫn điện trên tấm wafer của chip.

Phân loại các tấm wafer

Bước cuối cùng trong quy trình chế tạo là *phân loại wafer* - một phép kiểm tra về điện để phát hiện và loại bỏ các chip không hoạt động. Hệ thống máy tính hoàn tất một loạt các phép đo kiểm để đảm bảo rằng các mạch điện của chip đáp ứng yêu cầu kỹ thuật và hoạt động theo thiết kế.

II.5.3 Các nhà máy sản xuất chip (Fab)

Trong các cơ sở sản xuất tinh vi này, Intel chế tạo chip trong những khu vực đặc biệt được gọi là *Phòng sạch*. Do các phần tử không nhìn thấy hoặc bụi bẩn có thể phá hỏng mạch điện phức tạp trên chip, không khí trong *phòng sạch* cũng phải "siêu" sạch (gấp hàng nghìn lần độ sạch của phòng mổ trong bệnh viện). Nhân viên kỹ thuật mặc những bộ quần áo đặc biệt, thường được gọi là "trang phục của thờ" trước khi vào phòng sạch. Trang phục đó ngăn chất bẩn như xơ vải hoặc tóc rơi vào tấm wafer.

Tự động hóa cũng đóng vai trò quan trọng trong một nhà máy sản xuất. Các mẻ wafer được giữ sạch cũng như được xử lý nhanh và hiệu quả khi chúng đi qua các *bình đựng có cửa mở về phía trước (FOUP)*. Mỗi FOUP được gắn nhãn có mã vạch xác định các thành phần được sử dụng để sản xuất chip. Nhãn này đảm bảo việc xử lý chính xác ở mỗi bước trong quy trình sản xuất. Tại nhà máy 300 mm, một FOUP chứa tới 25 tấm wafer và nặng hơn 11 kg (25 pound). Tự động hóa cho phép đạt đến trọng lượng này vì nó quá nặng để có thể mang vác hiệu quả bằng nhân công.

II.5.4 Quy trình sản xuất

Tạo ra bóng bán dẫn trên các chip máy tính là một quy trình rất chính xác và phức tạp. Dưới đây là tổng hợp về các bước chính trong quy trình:

Cách điện và bọc: Một lớp cách điện được "cấy" trên bề mặt của tấm wafer đã được đánh bóng trong lò ở nhiệt độ cao với sự hiện diện của

ôxy. Tấm wafer sau đó được phủ chất liệu nhạy sáng gọi là *quang trở* và cũng giống như các tấm phim chụp ảnh, nó thay đổi về mặt hóa học khi bị phơi sáng dưới tia tử ngoại.

Che mặt nạ: Mặt nạ (tạo trong giai đoạn thiết kế) quy định hình dạng của mạch điện trên mỗi lớp của chip. Một chiếc máy tinh vi có tên máy xếp bậc sẽ cân chỉnh mặt nạ vào tấm wafer. Nó chiếu tia tử ngoại thông qua những chỗ bị phơi sáng trên mặt nạ. Quang trở ở phần bị chiếu sáng trở thành một lớp trong và dính.

Khắc axit: Những phần bị phơi sáng của quang trở bị loại bỏ làm lộ các phần của lớp oxyd silic phía dưới. Phần hở của oxyd silic bị loại bỏ bằng quá trình được gọi là khắc axit. Sau đó, phần quang trở còn lại cũng bị loại, để lại mẫu của oxyd silic trên tấm wafer.

Ghép thêm các lớp: Những vật liệu khác như là *Silic đa tinh thể (polysilicon)*, có tính dẫn điện được phủ lên tấm wafer thông qua các bước che mặt nạ và khắc mở rộng. Mỗi lớp của vật liệu mang một mẫu mạch điện duy nhất. Nhiều lớp được trải trên tấm wafer và sau đó được loại bỏ theo từng chỗ nhỏ để tạo thành bóng bán dẫn và kết nối. Chúng sẽ trở thành mạch điện bán dẫn của chip theo một cấu trúc trong không gian 3 chiều.

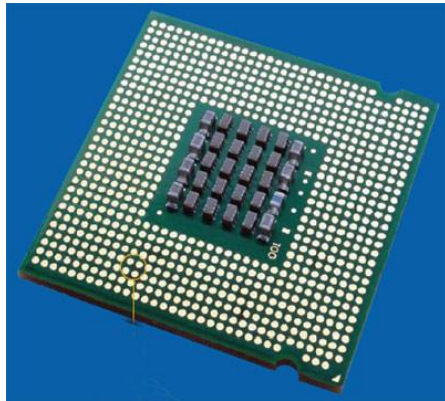
Hóa khắc (Doping): Những khu vực hở của wafer bị bắn phá bởi ion của các hợp chất hóa học khác nhau, làm thay đổi cách thức dẫn điện của silic trong những khu vực này. Hóa khắc là quá trình chuyển silic thành những bóng bán dẫn silicon, giúp bóng bán dẫn thực hiện việc bật và tắt - hai trạng thái được dùng để biểu diễn các số nhị phân 1 và 0. Các biểu diễn nhị phân này sau đó được dịch thành các ký tự, con số, màu sắc và hình ảnh đồng thời tạo ra nền tảng để lưu trữ dữ liệu trong máy tính.

Tạo công tắc: Để tạo kết nối với các lớp khác trên tấm wafer, các công tắc điện tử được tạo thành qua nhiều bước che mặt nạ và khắc lặp đi lặp lại.

Thêm kim loại: Nhiều lớp kim loại được sử dụng để hình thành kết nối điện giữa các lớp trong chip. Intel sử dụng nguyên liệu đồng trong khâu này.

Hoàn thiện tấm wafer: Một tấm wafer hoàn thiện chứa hàng triệu bóng bán dẫn. Mỗi bóng bán dẫn đóng vai trò như một công tắc đóng hoặc mở cho dòng điện đi qua. Một điện áp dương đặt trên *cực cổng* của bóng bán dẫn sẽ hút điện tử (cổng này tạo ra một kênh giữa cực nguồn và cực máng của bóng bán dẫn để điện tử có thể đi qua). Trong khi đó, một điện áp âm trên cực cổng sẽ chặn dòng điện không cho đi qua bóng bán dẫn.

II.5.5 Đóng gói



Sau khi được phân loại, wafer được đưa đến nhà máy lắp ráp của Intel, nơi một cái cưa chính xác phân tách mỗi tấm wafer thành từng chip riêng lẻ gọi là *khuôn (die)*. Mỗi chip được lắp ráp vào một đế (ngoài chức năng bảo vệ chip, chiếc đế đó còn có nhiệm vụ cung cấp nguồn và các kết nối điện với bo mạch chủ trên máy tính). Đây chính là những IC thành phẩm mà người ta sẽ gắn trên bo mạch chủ của máy tính hoặc điện thoại di động, PDA...

II.5.6 Tái kiểm tra

Intel thực hiện các phép kiểm tra về điện và độ tin cậy đối với từng thành phẩm. Họ thẩm định việc sản phẩm phải hoạt động ở tốc độ thiết kế trên toàn dải nhiệt độ. Bởi vì các chip có thể được sử dụng trong nhiều sản phẩm khác nhau, từ động cơ ô tô cho đến máy bay và laptop nên chúng phải có khả năng chống chịu những tác động của nhiều điều kiện môi trường. Sau khi được kiểm duyệt, các chip được đánh mã, kiểm tra bằng mắt, đóng gói và được chuyển giao cho khách hàng của Intel.

II.5.7 Một số thuật ngữ trong sản xuất chip

Nhi phân (Binary): Có hai phần. Hệ đếm nhị phân mà máy tính sử dụng bao gồm hai con số 0 và 1.

Lưỡng cực (Bipolar): Một công nghệ có tốc độ cao nhưng tương đối đắt đỏ. Được sử dụng trong sản phẩm đầu tiên của Intel - bộ nhớ 3101.

Chip: Một hình chữ nhật hoặc hình vuông nhỏ, mỏng có chứa các vi mạch điện tử tích hợp. Các chip còn được gọi là *khuôn*. Chip phức tạp nhất chính là bộ vi xử lý. (Xem thêm thuật ngữ "Bộ vi xử lý").

Mạch điện (Circuit): Đường đi cho dòng điện tử di chuyển.

Phòng sạch (Cleanroom): Phòng vô trùng dùng để chế tạo các chip. Không khí trong căn phòng này sạch hơn hàng nghìn lần so với không khí của các phòng phẫu thuật.

CMOS (complementary metal oxide semiconductor): Bao gồm cả các bóng bán dẫn kênh dương và kênh âm trên cùng một thiết kế mạch điện. Tạo ra các mạch điện có mức tiêu thụ nguồn thấp. Hiện nay phần lớn các chip của Intel được sản xuất bằng công nghệ này.

Thiết kế được máy tính hỗ trợ (Computer-aided design - CAD): Hệ thống máy tính và phần mềm tinh vi được sử dụng để thiết kế các chip vi mạch tích hợp.

Khuôn (Die): Xem "Chip" và "Bộ vi xử lý".

Hóa khắc (Doping): Một quy trình sản xuất tấm wafer trong đó những khu vực hồ của tấm silicon được bắn phá bằng các hợp chất hóa học để thay đổi bản chất dẫn điện của silicon tại những khu vực đó.

Kênh dẫn (Drain): Một khu vực có mức độ bắn phá bằng hợp chất hóa học cao, gần với một kênh mang dòng điện hiện tại của bóng bán dẫn. Truyền tải dòng điện tử đi ra từ bóng bán dẫn đến phần tử mạng hoặc dây dẫn mới.

Khắc axit (Etching): Việc loại bỏ những phần nhất định của vật liệu để xác định các lớp theo mẫu trên chip.

Chế tạo (Fabrication): Quá trình sản xuất chip.

Fab: Thuật ngữ rút gọn của "Nhà máy sản xuất chip", nơi chế tạo các chip silicon.

Đóng gói chip kiểu "lật" (Flip-chip packaging): Một kiểu đóng gói chip bằng cách lật chip về mặt trước và gắn nó với đế, khác với kiểu đóng gói chip như là kiểu nối dây trong đó gắn phần lưng của chip với đế.

Hộp chứa wafer có cửa mở ở mặt trước (Front-opening unified pod - FOUP): Một hộp chứa là một phần của hệ thống tự động trong một nhà máy sản xuất chip dùng để đựng và vận chuyển các tấm wafer. Màu của FOUP biểu thị khi nào thì các tấm wafer chứa đồng hoặc nhôm. Bình màu cam chứa đồng; Bình màu xanh chứa nhôm.

Cổng (Gate): Khu vực tín hiệu điều khiển của một bóng bán dẫn nơi có thể áp đặt một điện áp âm hoặc dương.

Thanh silic hình trụ (Ingot): Thanh silicon gần tinh khiết hình trụ giống như một chiếc xi-lanh dùng để cắt thành các tấm wafer.

Mặt nạ (Mask): Một tấm thạch anh với những mẫu vẽ giống như khuôn tô được khắc bằng crôm. Được sử dụng trong chế tạo tấm wafer khi "in" các mẫu mạch điện theo lớp trên một con chip.

Bộ vi xử lý (Microprocessor): "Bộ não" của mỗi chiếc máy tính. Nhiều bộ vi xử lý phối hợp làm việc với nhau tạo thành "trái tim" của máy chủ, thiết bị truyền thông và nhiều thiết bị số khác.

Nanometer: Một phần tỷ mét.

Công nghệ NMOS (negative-channel metal oxide semiconductor): Công nghệ ưa thích trong sản xuất những chip hàng đầu ở thập kỷ 70 và

80 thế kỷ trước. Đây là một công nghệ rẻ và có mật độ bóng bán dẫn cao hơn so với công nghệ lưỡng cực và có tốc độ cao hơn công nghệ PMOS.

Công nghệ quang khắc (Photolithography): Quá trình chuyển các mẫu thiết kế mạch điện của chip lên bề mặt của tấm wafer bằng cách sử dụng các mặt nạ để chuyển hình ảnh bằng cơ cấu quang học.

Quang trở (Photoresist): Một chất liệu bị tan ra khi chiếu tia tử ngoại. Được sử dụng để hình thành mẫu mạch điện trong quá trình chế tạo chip.

Mảng chân cắm lưới (Pin grid array - PGA): Một công nghệ đóng gói chỉ được sử dụng cho các bộ vi xử lý cao cấp.

Công nghệ PMOS (positive-channel metal oxide semiconductor): Một công nghệ có tốc độ thấp và hiện nay đã lỗi thời từng được sử dụng trong những sản phẩm MOS thế hệ đầu tiên của Intel.

Silic đa tinh thể (Polysilicon): Viết tắt của polycrystalline silicon, hay silic được tạo thành từ nhiều tinh thể. Vật liệu dẫn điện này được sử dụng như là một lớp kết nối trên một chip.

Sơ đồ mạch điện (Schematic): Sơ đồ biểu thị thuộc tính logic của mạch điện.

Chất bán dẫn điện (Semiconductor): Một loại vật liệu (như là silic) có thể bị biến đổi để cho phép dòng điện tử đi qua hoặc chặn dòng điện đó lại.

Silic (Silicon): Vật liệu được sử dụng để chế tạo các tấm wafer và từ đó sản xuất ra các chip. Nó là vật liệu bán dẫn điện tự nhiên và là nguyên tố phổ biến thứ hai trên trái đất chỉ đứng sau oxy.

Điôxit Silic (Silicon dioxide): Vật liệu được phủ lên trên tấm wafer trong quá trình chế tạo chip với vai trò của một lớp cách điện. Thủy tinh là một dạng thường thấy của Điôxit Silic.

Cực nguồn (Source): Khu vực của một bóng bán dẫn nơi các điện tử đi vào kênh dẫn.

Công nghệ hàn bề mặt (Surface-mount technology - SMT): Một dạng công nghệ hàn cho phép dán chip lên trên bề mặt thay vì gắn với các lỗ cắm trên một bảng mạch in.

Công nghệ đóng gói chip kiểu xếp chồng (Stacked-chip packaging): Là một công nghệ hàn cho phép xếp chồng nhiều chip trên một bảng mạch.

Bóng bán dẫn (Transistor): Một dạng công tắc điều khiển luồng chảy của dòng điện tử. Một con chip có thể chứa hàng triệu, thậm chí hàng tỷ bóng bán dẫn.

Tấm Wafer: Một miếng silicon mỏng được cắt ra từ thanh silicon hình trụ. Được sử dụng như là vật liệu nền để sản xuất vi mạch tích hợp.

Phân loại tấm Wafer (Wafer sort): Một thủ tục kiểm tra bằng điện tử để phát hiện những chip không hoạt động trên một tấm wafer.

Nối dây (Wire bonding): Quá trình kết nối những sợi dây cực mảnh từ các mạch điện trên chip với dây dẫn trên đế chip.

CÂU HỎI ÔN TẬP CHƯƠNG II.

1. Các tính năng cơ bản của CPU 8085
2. Các nhóm tín hiệu của CPU 8085
3. BUS là gì, chức năng và bản chất vật lý của BUS
4. Mô tả và giải thích chức năng của BUS địa chỉ, BUS dữ liệu và BUS điều khiển.
5. Tính năng và công dụng của mạch 3 trạng thái trong các hệ Vi xử lý
6. Giải thích biểu đồ định thời thực hiện thao tác trao đổi dữ liệu của CPU với bộ nhớ
7. Chu kỳ BUS, định nghĩa và phân loại các loại chu kỳ BUS.
8. Mục đích và ý nghĩa của Ngắt, của truy nhập trực tiếp bộ nhớ (DMA). Phân tích các lợi thế của các phương pháp sử dụng ngắt và sử dụng truy nhập trực tiếp bộ nhớ.
9. Chức năng của khối tạo xung nhịp (Clock Generator).
10. Trình bày và giải thích khái niệm vi chương trình.
11. Phân loại và giải thích chức năng các nhóm lệnh của CPU 8085.
12. Giải thích chức năng thanh ghi PC. Nội dung ô nhớ do nội dung thanh ghi PC chỉ đến là dữ liệu gì?
13. Giải thích hoạt động của thanh ghi SP, tác dụng của vùng nhớ ngăn xếp trong hoạt động thực thi chương trình.
14. Mô tả sơ lược chế độ MIN, MAX của CPU8088/86
15. Mô tả chức năng các chân của CPU8086
16. Các khối chức năng trong trung tâm CPU8086, so sánh sự khác nhau giữa CPU 8086 với CPU 8085
17. Tổ chức thanh ghi của CPU 8086

18. Phương thức quản lý bộ nhớ của CPU 8086, các phương pháp định vị ô nhớ.
19. Nội dung thanh ghi DS là 4200H, nội dung thanh ghi BX là 2000H, nội dung thanh ghi SI là 2600H. Hãy viết lệnh chuyển dữ liệu ở ô nhớ có địa chỉ vật lý là 46622H vào thanh ghi AL.
20. Phân biệt kiến trúc RISC và cấu trúc CISC.
21. Bộ nhớ cache và hoạt động ghi/đọc bộ nhớ cache.
22. Tổng quan về cấu trúc và chức năng của họ MCS51
23. Phân tích khả năng ứng dụng MCS51 trong thực tế.
24. Phân tích sự khác nhau giữa cấu trúc MCS51 và vi xử lý họ 8x86.

CHƯƠNG III. BỘ NHỚ TRONG CỦA HỆ VI XỬ LÝ

III.1 Bộ nhớ trong hệ Vi xử lý

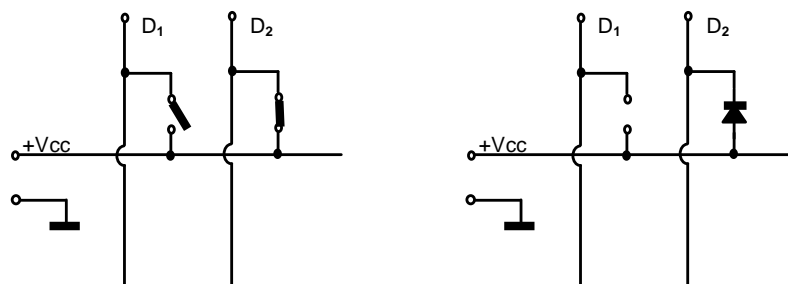
Bộ nhớ được sử dụng để lưu giữ mã lệnh của chương trình và dữ liệu cần xử lý. Bộ nhớ được ghép nối trực tiếp với CPU qua BUS hệ thống và là nơi đầu tiên CPU truy xuất tới để lấy thông tin khi khởi động hệ thống. Yêu cầu đặt ra cho bộ nhớ là phải cho phép truy xuất với tốc độ cao để đáp ứng kịp thời các đòi hỏi của CPU. Chỉ có bộ nhớ bán dẫn mới đáp ứng được yêu cầu cao về tốc độ truy xuất cao (hàng trăm đến hàng chục nsec).

Bộ nhớ bán dẫn được chia ra hai loại: Bộ nhớ chỉ đọc ROM (Read Only Memory) và bộ nhớ truy xuất ngẫu nhiên RAM (Random Access Memory).

III.1.1 Phân tử nhớ, vi mạch nhớ, từ nhớ và dung lượng bộ nhớ

a) Phân tử nhớ

Phân tử nhớ thông thường là một mạch điện có thể ghi lại và lưu giữ một trong hai giá trị của một biến nhị phân, hoặc “0” hoặc “1”, tương ứng với *không có điện áp* hoặc *có điện áp*, được gọi là **bit**. Trên mạch điện dưới đây (Hình III.1), trên dây D_1 sẽ không có điện áp (do công tắc mở), trong khi dây D_2 có điện áp (vì công tắc đóng, hay thông qua diode mắc theo chiều thuận), gần bằng giá trị nguồn nuôi V_{cc} , tương ứng với bit $D_1 = “0”$ và bit $D_2 = “1”$.



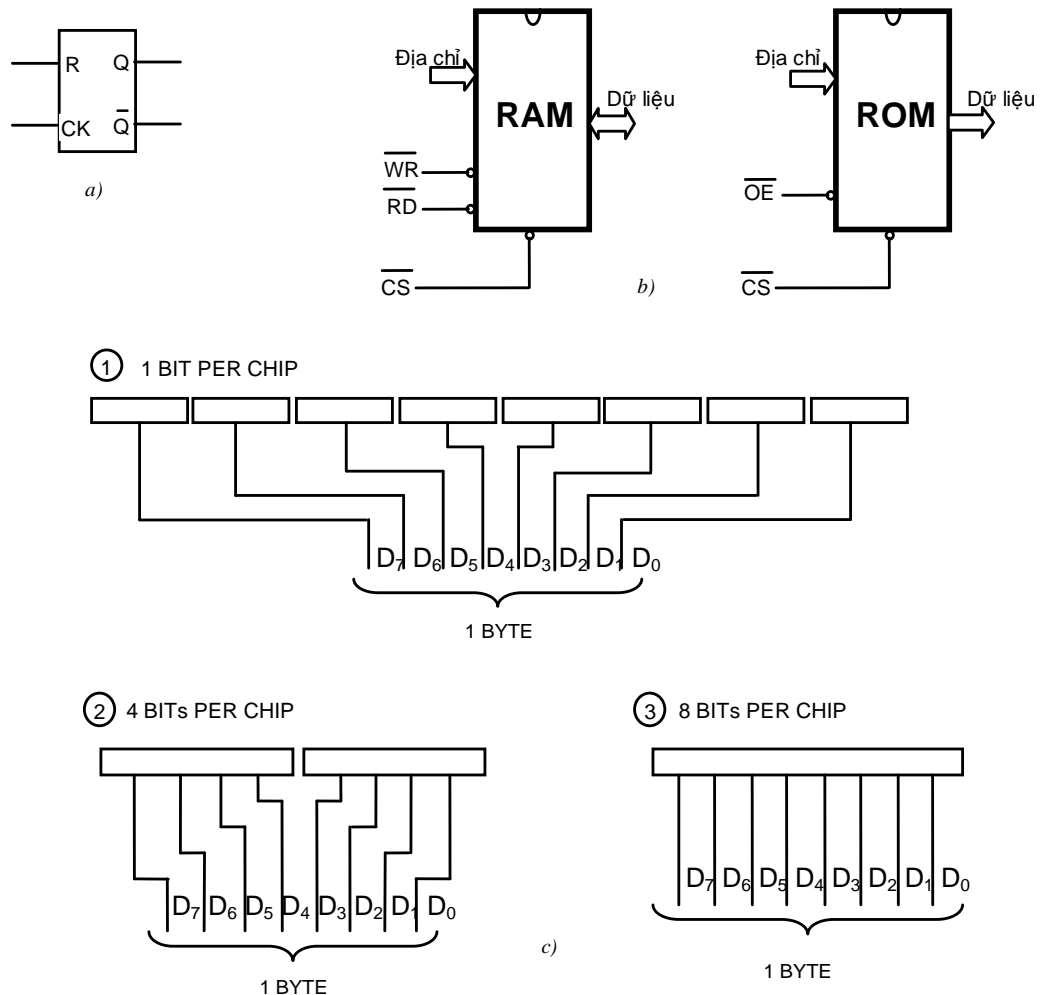
Phương pháp tạo phân tử nhớ $D_1 = 0$ và $D_2 = 1$ bằng mạch điện đơn giản

Hình III.1 Mô phỏng phân tử nhớ

Mạch Flip-Flop RS (còn gọi là trigger RS) đồng bộ là một mạch có khả năng lưu giữ các giá trị “0” hoặc “1” ở lối ra. Có thể dùng RS Flip-Flop làm một mạch lưu giữ tín hiệu vào R bằng cách chốt dữ liệu đó lại tại đầu ra Q (hình III.2a). Các hãng chế tạo thực hiện mạch này bằng công nghệ cao, nên kích thước vô cùng nhỏ, có thể có hàng nhiều triệu phân tử nhớ trên một diện tích 1mm^2 . Các vi mạch nhớ thông thường được chế tạo với độ dài từ nhớ và số lượng từ nhớ cố định. *Số bit nhớ được liên kết tại một vị trí nhớ (có cùng địa chỉ) trong một chip nhớ* được gọi là *từ nhớ*

của chip nhớ, thường được chọn là 1, 4, hoặc 8bit. Để tạo được một từ nhớ của bộ nhớ, tức là từ nhớ có độ dài (số bit trong một từ) chuẩn (theo chuẩn IBM là 8 bits), trong một số trường hợp nhất định cần phải tiến hành ghép các chip nhớ lại với nhau.

Hình III.2 a), b) và c) cho ta khái niệm về khả năng tạo một từ nhớ cơ bản (byte) khi từ nhớ của chip nhớ là 1bit, 2bits và 4 bits. Trong trường hợp độ dài từ nhớ của chip nhớ là 8 bits, việc liên kết là không cần thiết.



Hình III.2 a) Mạch Flip-flop RS như một phần tử nhớ giá trị nhị phân
 b) Chip nhớ RAM và chip nhớ ROM
 c) Ghép các chip nhớ có độ dài từ nhớ khác nhau để tạo được từ nhớ có độ dài 8 bits

III.1.2 Bộ nhớ trong của hệ Vi xử lý và máy tính PC

Do ưu điểm tương thích tuyệt đối về kích thước, tiêu thụ năng lượng thấp và mức logic, đặc biệt là tốc độ truy nhập, nên bộ nhớ bán dẫn được sử dụng làm bộ nhớ chính (Main Memory) trong các hệ Vi xử lý cũng như trong các máy tính PC, nhiều khi được ghép nối ngay trong bo mạch chính, hoặc được thiết kế như những vi nhỏ cắm vào khe cắm riêng trên bo mạch chính.

Nhờ những tiến bộ vượt bậc của công nghệ vi mạch, đặc biệt là công nghệ cao (High Technology) các chip nhớ được chế tạo ngày càng nhỏ và có dung lượng tương đối lớn, tốc độ truy nhập rất cao và giá thành thấp. Hiện đã có các chip nhớ có dung lượng hàng trăm triệu từ nhớ, được cấu thành từ hàng chục tỷ transistor trên một cấu trúc cỡ 1mm^2 .

Bộ nhớ trong của một hệ Vi xử lý gồm hai loại chính:

- Bộ nhớ ROM – là bộ nhớ chỉ đọc (Read Only Memory), thông thường chứa các chương trình giám sát (monitoring) các hoạt động chức năng của hệ Vi xử lý: chương trình thiết lập hệ thống, chương trình vào/ra dữ liệu, quản lý và phân phát bộ nhớ, quản lý các thiết bị vào/ra v.v... Đối với máy tính PC, đó là chương trình hệ thống vào/ra cơ sở (BIOS – Basic Input Output System). Đặc điểm cơ bản nhất của bộ nhớ này là sự bảo toàn dữ liệu khi không có nguồn nuôi.
- Bộ nhớ RAM – là bộ nhớ ghi/đọc tùy tiện (Random Access Memory). Vì có khả năng ghi/đọc tùy theo người dùng, nên bộ nhớ này được sử dụng để lưu giữ dữ liệu, các chương trình ứng dụng nhất thời của người dùng v.v... Trong máy tính PC, bộ nhớ này là nơi chương trình hệ điều hành được nạp khi khởi động máy, hay nơi chứa các chương trình ứng dụng lúc nó được thực thi. Bộ nhớ này bị mất dữ liệu khi bị mất nguồn nuôi.

Trong các hệ Vi xử lý đơn giản, hai bộ nhớ này thường được thiết kế và lắp ráp từ các chip nhớ riêng biệt thành một vi nhớ. Địa chỉ được *giải mã cho từng chip nhớ* nhờ khối giải mã, thông thường là một vi mạch giải mã hay được xây dựng từ các mạch tổ hợp logic. Các tín hiệu điều khiển việc ghi/đọc bộ nhớ do CPU cung cấp. Mạch trigger RS đồng bộ là một mạch có khả năng lưu giữ các giá trị “0” hoặc “1” ở lối ra. Có thể dùng RS Flip-Flop làm một mạch lưu giữ tín hiệu vào R bằng cách chốt dữ liệu đó lại tại đầu ra Q (hình III.2a)

Bộ nhớ được xây dựng từ các chip nhớ. Các chip nhớ RAM (SRAM hoặc DRAM) thường có các từ nhớ có độ dài 1 bit, 4 bits hoặc 8 bits. Từ các chip nhớ loại này có thể xây dựng được bộ nhớ với mỗi ô nhớ chứa được byte dữ liệu (8 bits).

III.1.3 Phân loại các chip nhớ ROM, RAM

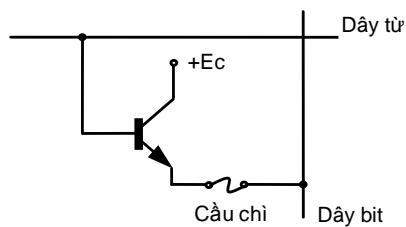
Bộ nhớ chỉ đọc ROM

Bộ nhớ chỉ đọc (*Read-only memory - ROM*) là một loại thiết bị lưu trữ dữ liệu dùng trong máy tính và các thiết bị khác. Nó có tên như vậy vì thực tế hầu như không dễ để ghi thông tin lên nó.

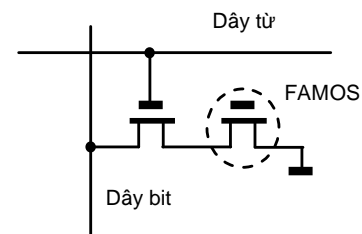
ROM, theo đúng nghĩa, chỉ cho phép đọc dữ liệu từ chúng tuy nhiên tất cả các loại ROM đều cho phép ghi dữ liệu ít nhất một lần, hoặc khi sản xuất lần đầu hoặc trong bước thực hiện ghi chương trình. Một số loại ROM cho phép xóa và ghi lại chương trình nhiều lần.

Các loại ROM

- **PROM** (*Programmable Read-Only Memory*): Được chế tạo bằng các mối nối (cầu chì - có thể làm đứt bằng mạch điện). Nó thuộc dạng **WORM** (*Write-Once-Read-Many*). Chương trình nằm trong PROM có thể lập trình được bằng những thiết bị đặc biệt. Loại ROM này chỉ có thể lập trình được một lần.

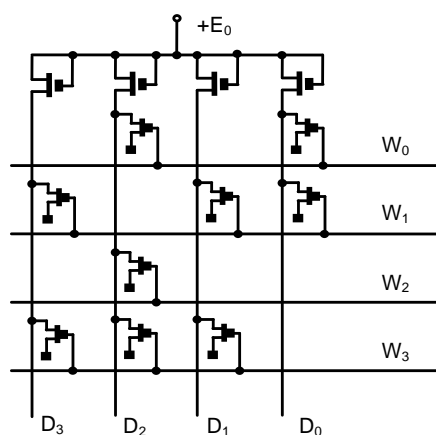


Phần tử nhớ PROM (cầu chì là hợp kim Ni-Cr)



Phần tử nhớ EPROM (cầu chì là Transistor FAMOS)

EPROM được chế tạo bằng nguyên tắc phân cực tĩnh điện. Cửa sổ nhỏ dùng để xóa bằng tia cực tím.



Mạch nhớ ROM 4 bit

- **EPROM** (*Erasable Programmable Read-Only Memory*): Được chế tạo bằng nguyên tắc phân cực tĩnh điện. Loại ROM này có thể bị xóa bằng tia cực tím và ghi lại thông qua thiết bị ghi EPROM.

- **EAROM** (*Electrically Alterable Read-Only Memory*): Loại ROM này có thể thay đổi từng bit một lần. Tuy nhiên quá trình viết khá chậm và sử dụng điện thế không chuẩn. Việc viết lại EAROM không được thực hiện thường xuyên.
- **EEPROM** (*Electrically Erasable Programmable Read-Only Memory*): Được chế tạo bằng công nghệ bán dẫn. Nội dung của ROM này có thể viết vào và xóa (bằng điện).

RAM

RAM (viết tắt từ **R**andom **A**ccess **M**emory trong tiếng Anh) là một loại bộ nhớ chính của máy tính. RAM được gọi là **bộ nhớ truy cập ngẫu nhiên** vì nó có đặc tính: thời gian thực hiện thao tác đọc hoặc ghi đối với mỗi ô nhớ là như nhau, cho dù đang ở bất kỳ vị trí nào trong bộ nhớ. Mỗi ô nhớ của RAM đều có một địa chỉ. Thông thường, mỗi ô nhớ là một byte (8 bit); tuy nhiên hệ thống lại có thể đọc ra hay ghi vào nhiều byte (2, 4, 8 byte).

RAM khác biệt với các thiết bị bộ nhớ tuần tự (*sequential memory device*) chẳng hạn như các băng từ, đĩa; mà các loại thiết bị này bắt buộc máy tính phải di chuyển cơ học một cách tuần tự để truy cập dữ liệu.

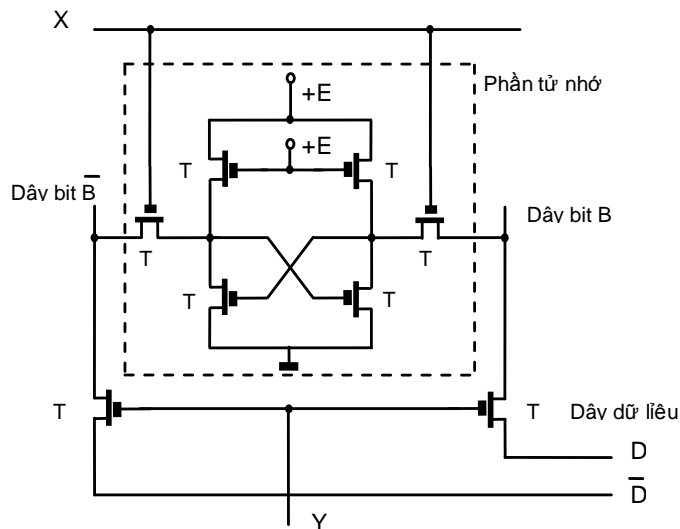
Bởi vì các chip RAM có thể đọc hay ghi dữ liệu nên thuật ngữ RAM cũng được hiểu như là một bộ nhớ đọc-ghi, trái ngược với bộ nhớ chỉ đọc ROM (*read-only memory*).

RAM thông thường được sử dụng cho bộ nhớ chính (*main memory*) trong máy tính để lưu trữ các thông tin thay đổi, và các thông tin được sử dụng hiện hành. Cũng có những thiết bị sử dụng một vài loại RAM như là một thiết bị lưu trữ thứ cấp (*secondary storage*). Thông tin lưu trên RAM chỉ là tạm thời, chúng sẽ mất đi khi mất nguồn điện cung cấp.

Phân loại RAM

Tùy theo công nghệ chế tạo, người ta phân biệt thành 2 loại:

- **SRAM** (Static RAM): RAM tĩnh
- **DRAM** (Dynamic RAM): RAM động
- **RAM tĩnh**
 - Trên hình bên là sơ đồ nguyên lý một ô nhớ của RAM tĩnh, mạch nhớ sử dụng 6 transistor. RAM tĩnh được chế tạo theo công nghệ ECL (dùng trong CMOS và BiCMOS). Mỗi bit nhớ gồm có các cổng logic với 6 transistor MOS.



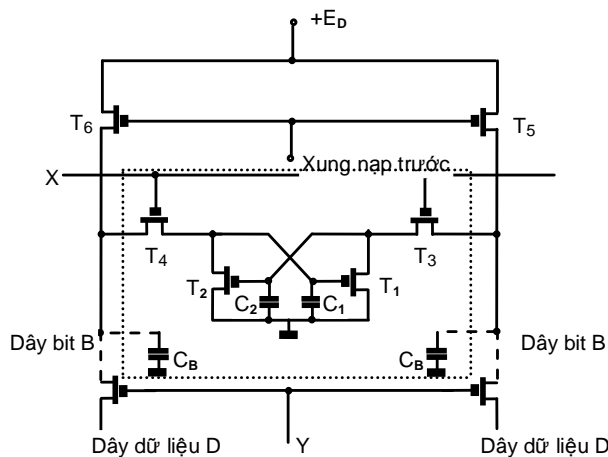
Hình III.7b – Sơ đồ cấu trúc các phần tử nhớ RAM tĩnh

SRAM là bộ nhớ nhanh, việc đọc không làm hủy nội dung của ô nhớ và thời gian thâm nhập bằng chu kỳ của bộ nhớ.

RAM động

Hình bên là sơ đồ một ô nhớ RAM động được tạo từ 1 transistor và 1 tụ điện. RAM động dùng kỹ thuật MOS. Mỗi bit nhớ gồm một transistor và một tụ điện. Việc ghi nhớ dữ liệu dựa vào việc duy

trì điện tích nạp vào tụ điện và như vậy việc đọc một bit nhớ làm nội dung bit này bị hủy. Do vậy sau mỗi lần đọc một ô nhớ, bộ phận điều khiển bộ nhớ phải viết lại nội dung ô nhớ đó. Chu kỳ bộ nhớ cũng theo đó mà ít nhất là gấp đôi thời gian thâm nhập ô nhớ.



Phần tử RAM động (Dynamic RAM) MOS

Việc lưu giữ thông tin trong bit nhớ chỉ là tạm thời vì tụ điện sẽ phóng hết điện tích đã nạp và như vậy phải làm tươi bộ nhớ sau khoảng thời gian $2\mu s$. Việc làm tươi được thực hiện với tất cả các ô nhớ trong bộ nhớ. Công việc này

được thực hiện tự động bởi một vi mạch bộ nhớ.

Bộ nhớ DRAM chậm nhưng rẻ tiền hơn SRAM.

Các loại DRAM sử dụng trong máy vi tính PC

1. **SDRAM** (Viết tắt từ **S**ynchronous **D**ynamic **R**AM) được gọi là DRAM đồng bộ. SDRAM gồm 3 phân loại: SDR, DDR, và DDR2.

- **SDR SDRAM** (**S**ingle **D**ata **R**ate SDRAM), thường được giới chuyên môn gọi tắt là "**SDR**". Có 168 chân. Được dùng trong các máy vi tính cũ, bus speed chạy cùng vận tốc với clock speed của memory chip, nay đã lỗi thời.
- **DDR SDRAM** (**D**ouble **D**ata **R**ate SDRAM), thường được giới chuyên môn gọi tắt là "**DDR**". Có 184 chân. DDR SDRAM là cải tiến của bộ nhớ SDR với tốc độ truyền tải

gấp đôi SDR nhờ vào việc truyền tải hai lần trong một chu kỳ bộ nhớ. Đã được thay thế bởi **DDR2**.

- o **DDR2 SDRAM (Double Data Rate 2 SDRAM)**, Thường được giới chuyên môn gọi tắt là "**DDR2**". Là thế hệ thứ hai của DDR với 240 chân, lợi thế lớn nhất của nó so với DDR là có bus speed cao gấp đôi clock speed.

2. **RDRAM (Viết tắt từ Rambus Dynamic RAM)**, thường được giới chuyên môn gọi tắt là "**Rambus**". Đây là một loại DRAM được thiết kế kỹ thuật hoàn toàn mới so với kỹ thuật SDRAM. RDRAM hoạt động đồng bộ theo một hệ thống lặp và truyền dữ liệu theo một hướng. Một kênh bộ nhớ RDRAM có thể hỗ trợ đến 32 chip DRAM. Mỗi chip được ghép nối tuần tự trên một module gọi là RIMM (Rambus Inline Memory Module) nhưng việc truyền dữ liệu được thực hiện giữa các mạch điều khiển và từng chip riêng biệt chứ không truyền giữa các chip với nhau. Bus bộ nhớ RDRAM là đường dẫn liên tục đi qua các chip và module trên bus, mỗi module có các chân vào và ra trên các đầu đối diện. Do đó, nếu các khe cắm không chứa RIMM sẽ phải gắn một module liên tục để đảm bảo đường truyền được nối liền. Tốc độ Rambus đạt từ 400-800MHz. Rambus tuy không nhanh hơn SDRAM là bao nhưng lại đắt hơn rất nhiều nên có rất ít người dùng. RDRAM phải cắm thành cặp và ở những khe trống phải cắm những thanh RAM giả (còn gọi là C-RIMM) cho đủ.

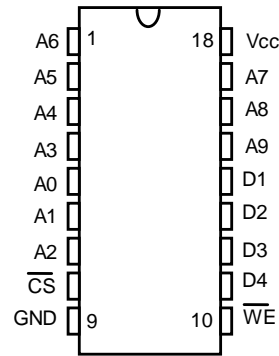
III.2 Tổ chức bộ nhớ cho hệ Vi xử lý

III.2.1 Tổ chức bộ nhớ vật lý

Tổ chức bộ nhớ cho một hệ Vi xử lý (máy vi tính) phụ thuộc không chỉ vào một hệ Vi xử lý cụ thể, mà còn phụ thuộc vào cách bố trí thuận lợi bên trong hệ thống. Trước hết, hãy làm quen với các khái niệm chip nhớ và từ nhớ để phân tích vấn đề tổ chức vật lý một bộ nhớ, sau đó mở rộng khái niệm tổ chức theo quan điểm của người lập trình (tổ chức logic).

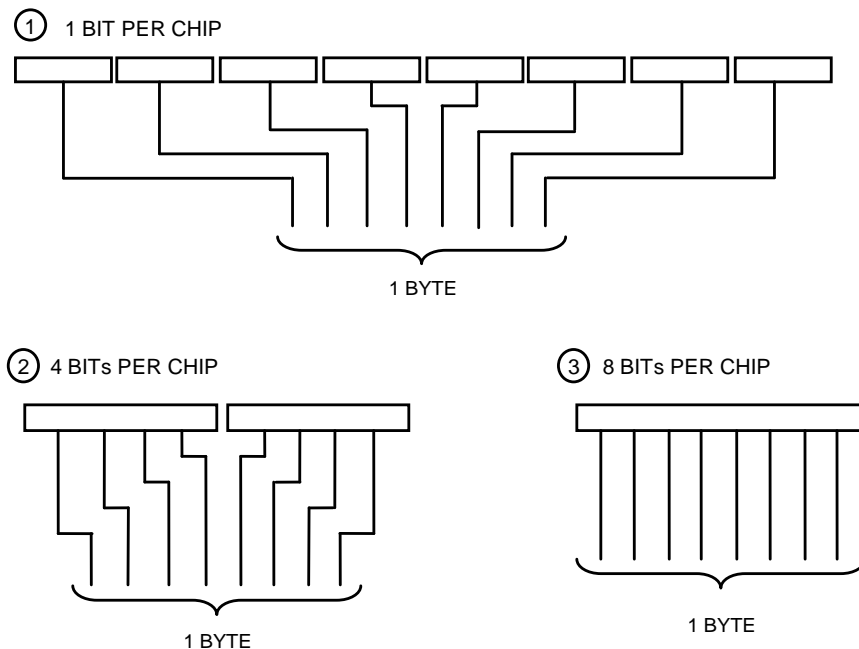
Các chip nhớ được sản xuất dưới nhiều kích cỡ và dung lượng nhớ, tổ chức từ nhớ khác nhau, phụ thuộc vào thế loại và công nghệ chế tạo. Chip nhớ là một vi mạch cụ thể, được bố trí các chân cơ bản như Hình III.8. Các chân của một chip nhớ thông thường gồm các lối vào của BUS địa chỉ, lối dữ liệu, các chân điều khiển chọn chip, ghi/đọc và các chân nguồn.

A0 ÷ A9 Các chân địa chỉ
 D1 ÷ D4 Các chân dữ liệu
 CS Chân chọn chip
 WE Điều khiển Ghi/Đọc
 Vcc Chân nguồn nuôi +5V
 GND Chân nối đất



Hình III.8 Sơ đồ nối chân một vi mạch nhớ RAM 1Kx4

Tùy theo từng chip, số lượng chân địa chỉ và số lượng chân dữ liệu có thể khác nhau phụ thuộc vào độ dài từ nhớ và dung lượng của chip nhớ. Độ dài từ nhớ của chip nhớ có thể là 1bit, 4 bits hoặc 8 bits, trong khi số chân địa chỉ có thể từ 10 trở lên tùy thuộc vào dung lượng của chip nhớ. Trong trường hợp độ dài từ nhớ của chip là 1 bit, ta cần phải ghép song song 8 chip để tạo thành 1 byte, ghép song song 16 chip để tạo một từ word – 2 bytes).

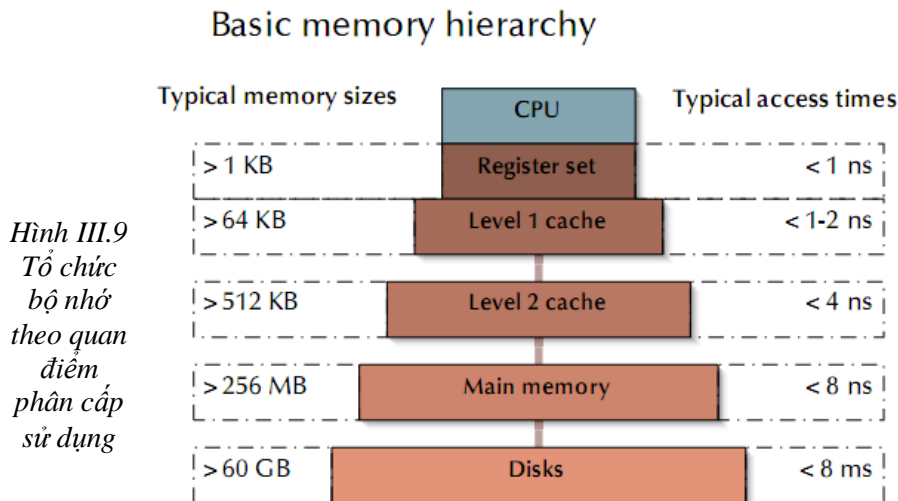


Hình III.8 Tạo từ nhớ 8 bit từ các chip nhớ có độ dài từ nhớ nhỏ hơn

Vấn đề tổ chức logic của bộ nhớ là một chương mục phức tạp vượt ngoài tầm của giáo trình, nghiêng về các phương diện kiến trúc máy tính PC, nên ở đây chỉ nêu lên một vài khái niệm đơn giản như sau:

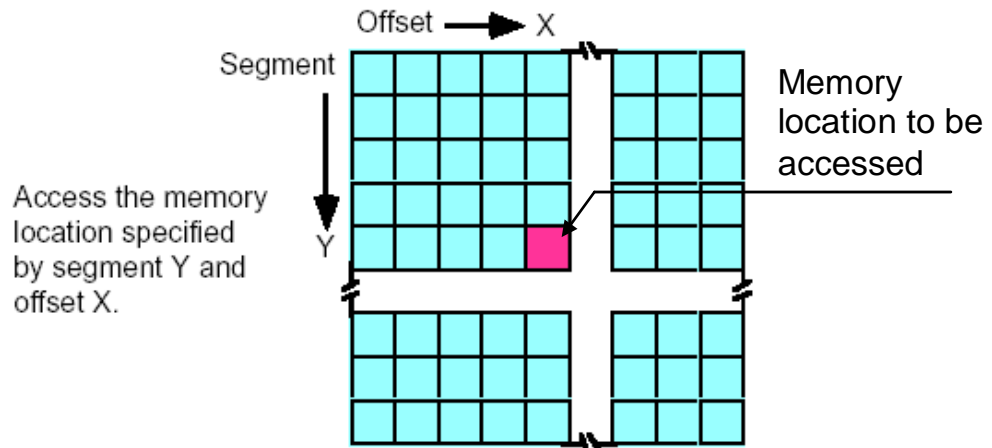
a) Thể loại bộ nhớ:

Memory sizes and access times: (typical workstation)

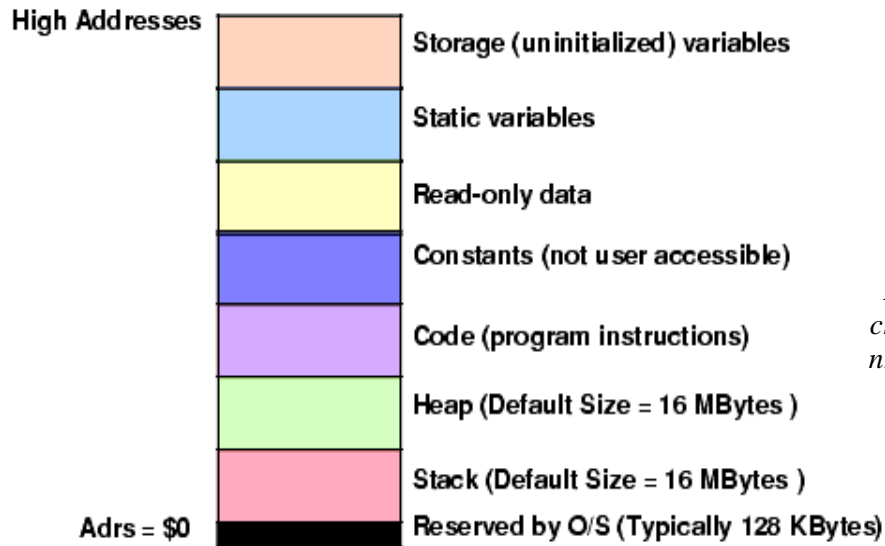


Một số khái niệm cần lưu ý:

- Thứ nhất, là bộ nhớ có các mức khác nhau, tùy theo mục đích sử dụng và tốc độ truy xuất
 - Tốc độ truy xuất nhanh nhất thuộc về tập các thanh ghi, thực chất là các vị trí nhớ tích hợp bên trong CPU
 - Bộ nhớ cache là bộ nhớ SRAM với tốc độ truy xuất cao, có giá thành cao nên được tổ chức với dung lượng giới hạn. Bộ nhớ cache lưu giữ các thông tin dữ liệu mà CPU truy xuất trực tiếp, không qua hệ thống BUS, tốc độ truy xuất cao hơn bộ nhớ chính của máy PC. Với các thế hệ CPU từ 80486 trở về sau này, cache được chia ra hai mức: mức một (Level 1) được tích hợp trực tiếp trên CPU còn mức 2 (Level 2) được đặt bên ngoài CPU.
 - Bộ nhớ chính (RAM) thông thường có tốc độ truy xuất thấp hơn, được sử dụng với dung lượng lớn hơn nhiều.
- b) Tổ chức bộ nhớ theo phân đoạn là đặc tính đặc trưng của CPU họ 80x86.



- o Các đặc trưng của phương pháp tổ chức này đã được trình bày trong chương II
- o Trong chế độ thực, cách thức tính địa chỉ vật lý từ một địa chỉ logic tuân thủ công thức đã nêu ở mục II.2.5 của chương II.
- c) Tổ chức bộ nhớ trong chế độ bảo vệ (protected mode)
 - o Khác với tổ chức bộ nhớ trong chế độ thực, trong chế độ bảo vệ, địa chỉ vật lý của một vị trí nhớ được xác định theo *phương thức tra bảng*. Các thanh ghi đoạn chỉ tới một cấu trúc dữ liệu chứa các thông tin cần thiết để truy xuất vị trí ô nhớ. Mỗi chương trình được thực hiện trong chế độ bảo vệ thường bao gồm một bảng các bộ mô tả (descriptor) có dung lượng 8 byte xác định vị trí đầu và vị trí cuối của đoạn. Tùy thuộc vào chức năng của đoạn mà bộ mô tả còn có thêm các thông tin về quyền truy nhập. Các thông tin trong bộ mô tả bao gồm:
 1. *Khởi đầu đoạn: địa chỉ tuyệt đối 32 bit*
 2. *Giới hạn đoạn: địa chỉ cao nhất đoạn có thể tham chiếu tới*
 3. *Trạng thái đoạn: mức đặc quyền, trạng thái tồn tại đoạn, thể loại đoạn v.v...*
- d) Các phương thức định vị vị trí nhớ đã được trình bày trong chương II
- e) Hệ điều hành tổ chức và quản lý bộ nhớ chính theo các phương thức sau:



Hình II.10 Tổ chức quản lý bộ nhớ chính trong máy tính PC

- Vùng địa chỉ thấp nhất được sử dụng như vùng nhớ giành riêng cho Hệ điều hành và dự phòng, thông thường người dùng không thể truy xuất được
- Tiếp theo là 6 vùng nhớ dành cho các thể loại dữ liệu khác nhau như vùng nhớ ngăn xếp, vùng nhớ sử dụng đa mục đích, vùng mã lệnh, vùng các dữ liệu chỉ đọc chứa các hằng số, vùng chứa các biến tĩnh, và vùng sử dụng cho các biến, truy xuất ngẫu nhiên không có giá trị khởi đầu.

III.2.2 Thiết kế vi nhớ cho hệ Vi xử lý

Thiết kế vi nhớ là một việc rất quan trọng và rất cần thiết trong việc xây dựng một hệ Vi xử lý. Các vi nhớ được thiết kế thông thường là EPROM, các loại vi nhớ RAM, từ các chip nhớ có sẵn. Thông thường, các chip nhớ được chọn là những chip thông dụng trên thị trường, có các thông số kỹ thuật chủ yếu sau:

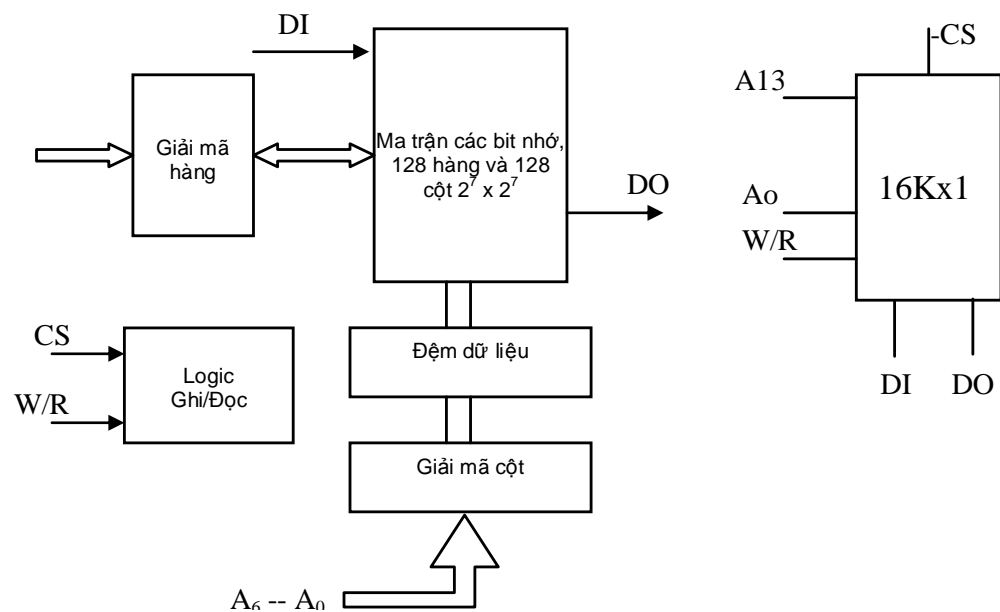
- Dung lượng nhớ của chip nhớ tính theo đơn vị Kbyte
- Độ dài từ nhớ của chip nhớ tính theo số bits
- Một số thông số kỹ thuật khác như thời gian truy xuất, công suất tiêu tán của chip v.v... Những thông số này không có ảnh hưởng lớn đến quá trình thiết kế và xây dựng vi nhớ.

Các thông số được cho trước trong việc thiết kế một vi nhớ bao gồm:

- Loại chip nhớ: ví dụ dùng EPROM 2764 (8Kx8) hay RAM TMS 2064 (8Kx8) v.v...
- Dung lượng của vi nhớ: là dung lượng vi nhớ phải có, ví dụ 64KB, 128KB v.v...
- Địa chỉ đầu của vùng nhớ: ví dụ vi nhớ có địa chỉ đầu là A0000H chẳng hạn.
- Xây dựng bộ nhớ với các chip SRAM

Giả sử cần xây dựng một bộ nhớ kích thước 16Kbyte trên cơ sở các chip SRAM loại 16Kx1bit.

Bảng nhớ SRAM 16Kbyte được xây dựng trên cơ sở 8 chip SRAM loại 16K x 1bit, để có được ô nhớ có độ dài 8 bits (từ nhớ cơ bản). Để làm được điều này người ta sắp đặt 8 chip SRAM loại 16K x 1bit sao cho mỗi



Hình III.3 Chip nhớ SRAM 64K bit (64K x 1)

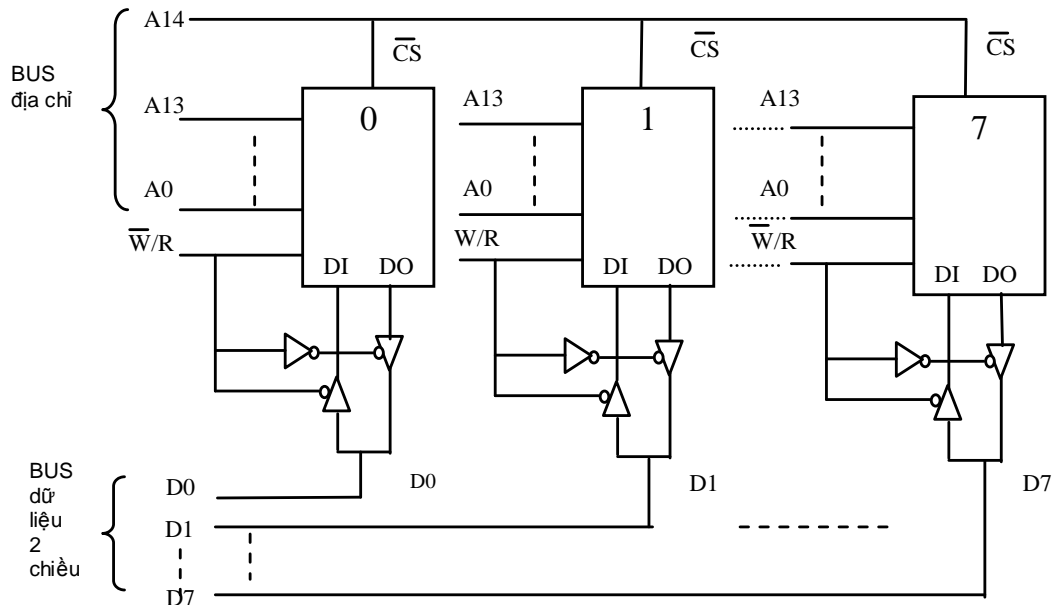
chip tại một vị trí xác định sẽ đảm nhiệm lưu trữ bit dữ liệu có trọng số tương ứng trong byte dữ liệu.

Các đường tín hiệu :

A13 - A0 BUS địa chỉ

\overline{CS} : Tín hiệu chọn chip. Nếu $\overline{CS} = 0$ thì truy nhập được chip

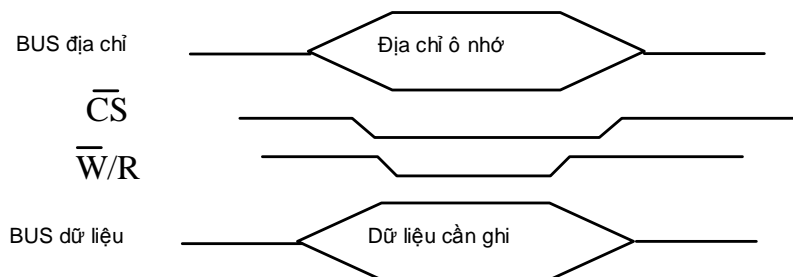
$\overline{W/R}$: Tín hiệu điều khiển ghi/đọc. $\overline{W}=0$ điều khiển ghi



Hình III.4 Sơ đồ vi nhớ 16KB xây dựng từ các chip 16Kx1

D0 - D7: Các đường dây truyền các bit dữ liệu từ D0 đến D7.

Chu kỳ ghi bộ nhớ SRAM :



Hình III.5 - Biểu đồ thời gian ghi đọc bộ nhớ

e. Tổ chức bộ nhớ với DRAM

DRAM dùng phương pháp dồn kênh để nạp lần lượt (2 lần) địa chỉ hàng và địa chỉ cột vào đệm địa chỉ.

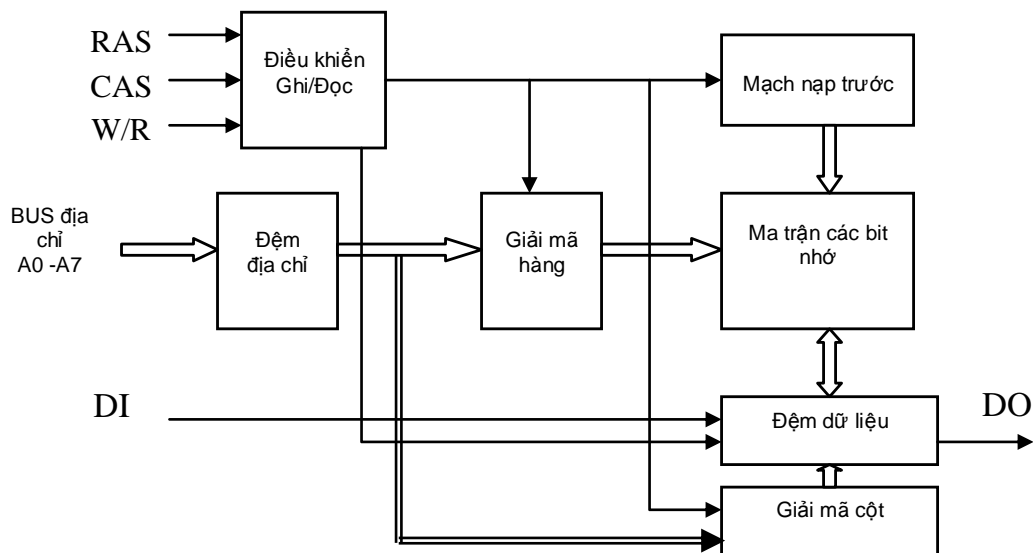
Tín hiệu điều khiển :

+ RAS: khi RAS (Row Access Strobe) tích cực thì địa chỉ hàng được nạp (chốt lại).

+ CAS: khi CAS (Column Access Strobe) tích cực thì địa chỉ cột được nạp (chốt lại).

+ WE: WE ≡ “0” điều khiển ghi chip, WE ≡ “1” điều khiển đọc chip.

Việc xây dựng bộ nhớ từ các chip DRAM được thực hiện gần tương tự như với SRAM.



Hình III.6 - Cấu trúc bên trong chip DRAM

Ví dụ minh họa: Dùng EPROM 2764 (8Kx8bit) xây dựng vi nhớ có dung lượng 32KB, địa chỉ đầu là 22000H.

Giải: Dựa trên yêu cầu của đề ra, phải thực hiện các bước sau:

1. Xác định số chip nhớ cần thiết để tạo từ nhớ cơ bản (độ dài 8 bits), có thể tính theo công thức:

$$n = \frac{8}{k} \text{ trong đó } n \text{ là số chip cần để tạo được từ nhớ cơ}$$

bản, k là độ dài từ nhớ của chip nhớ

Tín hiệu chọn vô \overline{CS} của các chip này được nối chung với nhau, các chip này được coi như một chip liên thông, các bit dữ liệu sẽ được định vị theo thứ tự từ $D_7 \div D_0$ tương ứng với các bit từ $D_7 \div D_0$ của BUS dữ liệu.

2. Xác định số chip nhớ, hoặc số chip liên thông để tạo được dung lượng nhớ theo yêu cầu. Trong trường hợp cụ thể của đề ra, cần 4 chip để tạo được dung lượng nhớ 32KB. Tính theo công thức:

$$M = \frac{Q}{D} \text{ trong đó } Q \text{ là dung lượng của vi nhớ}$$

D là dung lượng của chip nhớ hoặc dung lượng của chip liên thông

M là số chip nhớ hoặc số chip liên thông cần thiết

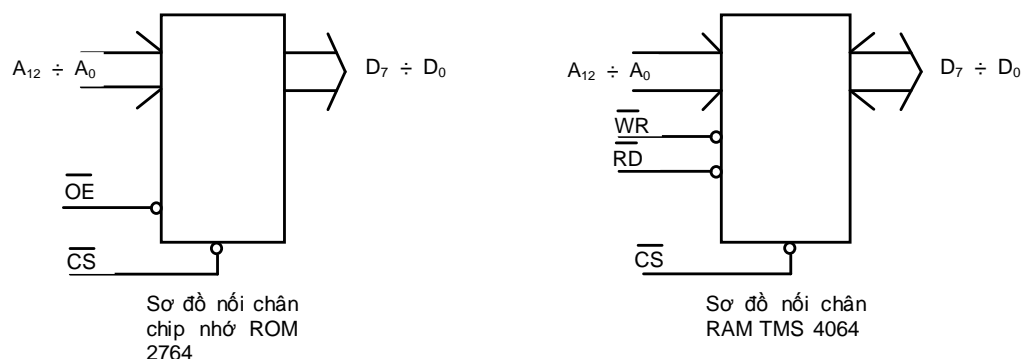
3. Xác định số dây địa chỉ cơ sở (tức là số dây địa chỉ thấp được nối trực tiếp vào chip nhớ hoặc chip liên thông): Số dây địa chỉ m phụ thuộc vào dung lượng nhớ của chip nhớ hoặc chip liên thông theo biểu thức sau:

$$2^m = D \text{ trong đó } D \text{ là dung lượng của chip nhớ, } m \text{ là số dây địa chỉ cơ sở}$$

4. Từ số chip hoặc số chip liên thông, xác định số dây địa chỉ cần thiết để tạo các dây chọn chip riêng biệt. Tính theo công thức:

$2^i = M$ trong đó i là số dây địa chỉ cần để giải mã xác định các tín hiệu chọn chip cho các chip nhớ hoặc chip liên thông. M là số lượng chip hoặc số lượng chip liên thông, i là số dây địa chỉ cần để giải mã tạo tín hiệu chọn chip \overline{CS}_i .

5. Các dây địa chỉ còn lại được sử dụng để tạo tín hiệu xác định vùng nhớ của vi nhớ trong không gian nhớ (được gán cho vi nhớ theo địa chỉ đầu của vi nhớ theo yêu cầu).

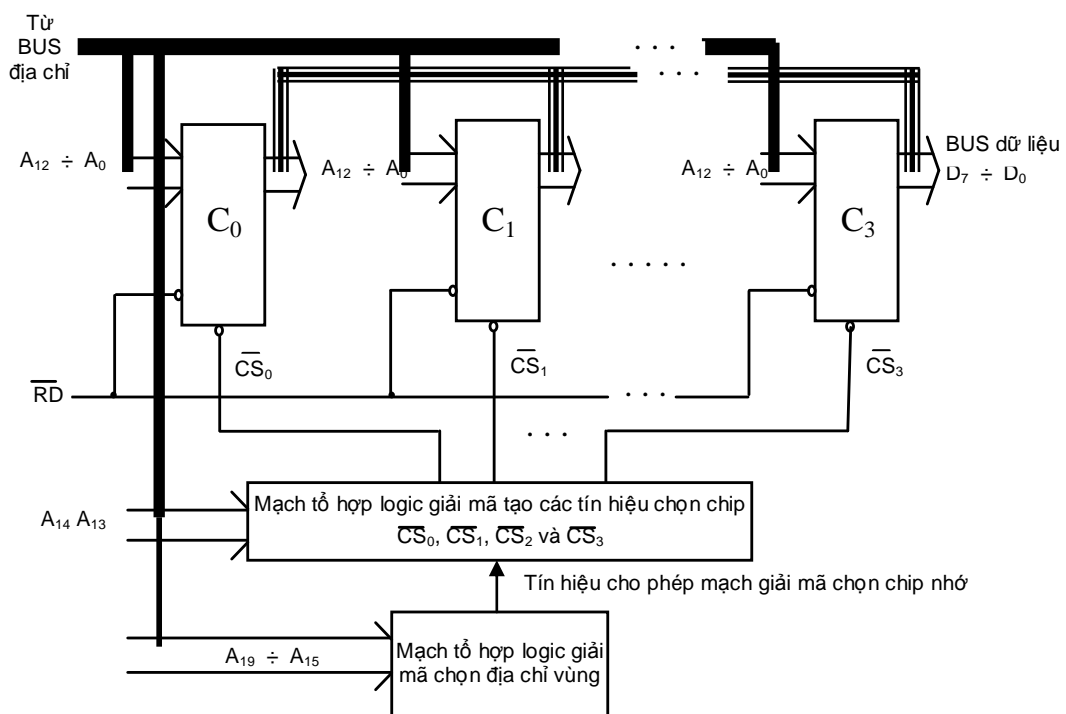


Hình III.9 Sơ đồ nối chân chip nhớ ROM và chip nhớ RAM

Sơ đồ khối ví như sau, các mạch tổ hợp logic xây dựng theo kiến thức học được ở môn học *Kỹ thuật điện tử số*.

Mạch tổ hợp logic giải mã chọn địa chỉ vùng được thiết kế dựa vào thông số địa chỉ đầu của vùng nhớ, đó là các bit nhớ cao nhất, có giá trị cụ thể và không thay đổi trong toàn bộ quá trình truy xuất đến các vị trí nhớ trong vi nhớ.

Mạch tổ hợp logic giải mã tín hiệu chọn chip nhớ có đầu vào là các bit địa chỉ tiếp theo kể từ các bit địa chỉ độc lập (tức là các bit địa chỉ được nối trực tiếp vào các chân địa chỉ của chip nhớ). Số lượng bit địa chỉ là lối vào của mạch này phụ thuộc vào số lượng chip nhớ hoặc số lượng chip liên thông tạo nên từ nhớ cơ bản, tuân thủ công thức $2^i = M$ trong điểm 4 của ví dụ minh họa nêu trong phần này.



Hình III. 10 Sơ đồ khối vi nhớ 32KB từ các chip ROM 2764

Ví dụ 1:

Thiết kế mạch giải mã chọn 1 từ 4 bằng các phần tử logic cơ bản (OR, NOT). Tín hiệu ra xác định việc chọn có giá trị là “0” logic.

Lời giải:

Thấy rằng, để có 4 trị, ta dùng tổ hợp 2 bit nhị phân với các giá trị tương ứng là 00, 01, 10, và 11. Như vậy, mạch giải mã chọn 1 từ 4 có 2 đầu vào và 4 đầu ra, tuân thủ bảng giá trị chân thực sau:

Để dàng nhận thấy rằng hàm logic thực hiện việc giải mã chọn 1 từ 4 có thể viết đơn giản như sau:

$$O_0 = I_0 + I_1$$

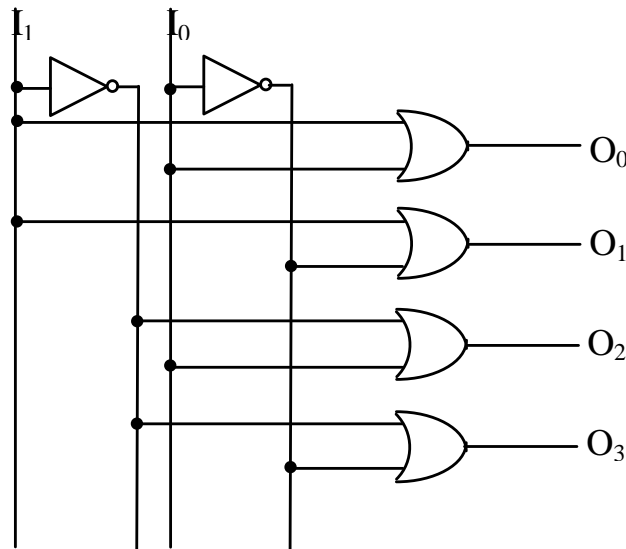
$$O_1 = \bar{I}_0 + \bar{I}_1$$

$$O_2 = \bar{I}_0 + I_1$$

$$O_3 = \bar{I}_1 + I_0$$

I_1	I_0	O_0	O_1	O_2	O_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

Mạch logic thực hiện các hàm trên xây dựng từ các phần tử OR và NOT như sau:



Ví dụ 2:

Cho vi mạch nhớ RAM tĩnh với các thông số kỹ thuật:

- Tổ chức từ nhớ 4bits
- Dung lượng 64 K từ nhớ (64Kx4bit)
- Địa chỉ đầu đoạn nhớ là 80000H

Hãy thiết kế một vi nhớ 512Kbytes cho phối ghép với CPU 8086

Lời giải:

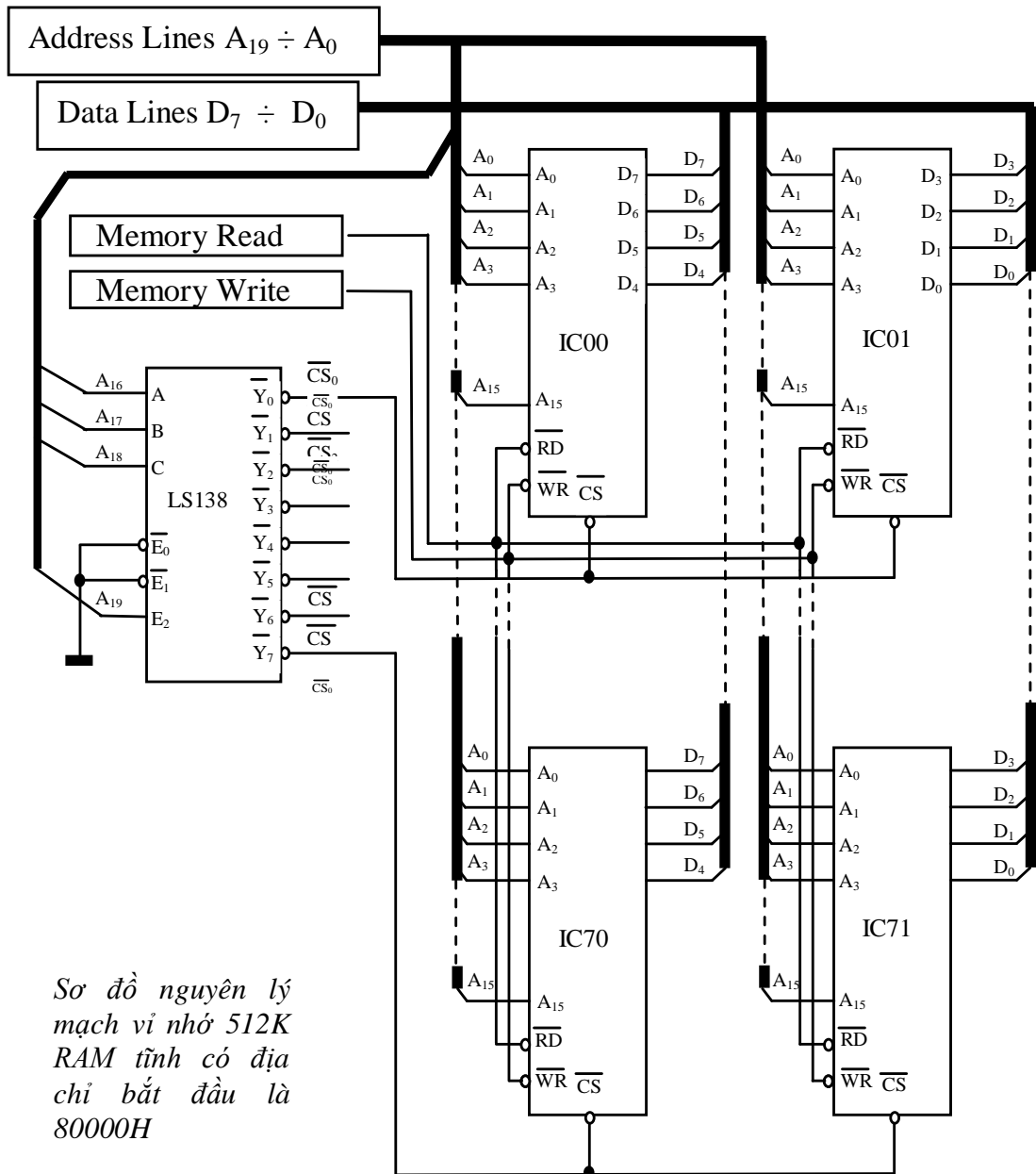
Từ nhớ của chip nhớ có độ dài 4 bit, do vậy cần $n = \frac{8}{4} = 2$ chip để tạo từ nhớ cơ bản (Byte = 8 bit). Tín hiệu chọn vở \overline{CS} của các chip này được nối chung với nhau, các chip này được coi như một chip liên thông, các bit dữ liệu sẽ được định vị theo thứ tự từ $D_7 \div D_0$ tương ứng với các bit từ $D_7 \div D_0$ của BUS dữ liệu.

Số chip liên thông cần để tạo dung lượng 512K là $M = \frac{Q}{D}$; Với $Q = 512K$ và $D = 64K$, ta có $M = 8$; do vậy phải tạo 8 tín hiệu chọn chip liên tục, ký hiệu từ \overline{CS}_0 đến \overline{CS}_7 cho các chip liên thông.

Số dây địa chỉ cơ sở: Với dung lượng chip nhớ là 64K, ta có $2^{16} = 65536$, (tức là 64K) do vậy, $m = 16$, tức là các dây địa chỉ từ $A_{15} \div A_0$ của BUS địa chỉ sẽ được nối trực tiếp với các chân địa chỉ tương ứng của tất cả các chip liên thông.

A_{19}	A_{18}	A_{17}	A_{16}	\overline{CS}_0	\overline{CS}_1	\overline{CS}_2	\overline{CS}_3	\overline{CS}_4	\overline{CS}_5	\overline{CS}_6	\overline{CS}_7
1	0	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	0	1	1	1	1	1	1
1	0	1	0	1	1	0	1	1	1	1	1
1	0	1	1	1	1	1	0	1	1	1	1
1	1	0	0	1	1	1	1	0	1	1	1
1	1	0	1	1	1	1	1	1	0	1	1
1	1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	0

Từ các phân tích trên, suy ra cần $i = 3$ dây địa chỉ để tạo được 8 tín hiệu chọn chip, sử dụng mạch giải mã 1/8 có sẵn là 74LS138, ta có được các tín hiệu chọn chip từ \overline{CS}_7 đến \overline{CS}_0 cho các chip liên thông. Đó là các dây địa chỉ A_{16} , A_{17} và A_{18} . Sơ đồ nguyên lý ví như được thể hiện trên hình vẽ sau



CÂU HỎI ÔN TẬP CHƯƠNG III.

1. Định nghĩa BIT nhớ, vai trò của bộ nhớ trong trong các hệ Vi xử lý.
2. Các khái niệm phần tử nhớ, vi mạch nhớ, từ nhớ và dung lượng bộ nhớ.
3. Phương pháp mở rộng từ nhớ và dung lượng nhớ của vi nhớ khi biết từ nhớ và dung lượng của chip nhớ.
4. Phân loại ROM, phân loại RAM và sơ đồ chân chip nhớ ROM, chip nhớ RAM tĩnh, RAM động.
5. Phân biệt sự khác nhau về tính năng giữa RAM tĩnh và RAM động.
6. Trình bày khái niệm và phương pháp làm tươi (refreshing) bộ nhớ RAM động.
7. Tổ chức bộ nhớ vật lý trong hệ Vi xử lý.
8. Khái niệm giải mã địa chỉ ô nhớ.
9. Phân tích hoạt động của mạch giải mã 74LS138
10. Thiết kế mạch giải mã chọn 1 từ 4 bằng các phần tử logic cơ bản (AND, OR, NOT). Tín hiệu ra xác định việc chọn có giá trị là “0” logic.
11. Các bước thiết kế một vi nhớ cho hệ Vi xử lý.
12. Cho vi mạch nhớ RAM tĩnh với các thông số kỹ thuật:
 - Tổ chức từ nhớ 4bits
 - Dung lượng 64 K từ nhớ (64Kx4bit)
 - Địa chỉ đầu đoạn nhớ là 80000HHãy thiết kế một vi nhớ 512Kbytes cho phối ghép với CPU 8086

CHƯƠNG IV. CÁC CHIP KHẢ LẬP TRÌNH (Programmable)

IV.1 Tổng quan

Chip khả lập trình (Programmable) là một loại mạch điện tử chuyên dụng có khả năng thực hiện chức năng thông qua việc cung cấp các từ điều khiển (Control Word - CW) được CPU gửi tới (do người lập trình soạn). Nội dung các bit định chức năng trong từ điều khiển sẽ điều khiển mạch làm việc theo những chế độ định trước. Tồn tại một số mạch chức năng chuyên dụng tiêu biểu cho các hệ Vi xử lý μ P8085 và họ các trung tâm vi xử lý 80x86 như mạch phối ghép ngoại vi song song khả lập trình PPI8255 điều khiển việc phối ghép vào/ra dữ liệu song song giữa CPU với các thiết bị ngoại vi, mạch đếm định thời và tạo khoảng thời gian PIT8253/54, mạch phối ghép vào ra dữ liệu nối tiếp USART 8251, mạch điều khiển ngắt PIC8259 v.v... Phần tiếp theo sẽ tìm hiểu một số mạch tiêu biểu.

IV.2 Một số mạch chức năng tiêu biểu

IV.2.1 Mạch vào/ra dữ liệu song song PPI-8255 (Programmable Peripheral Interface).

a) Giới thiệu chung

PPI8255 là mạch giao diện thiết bị ngoại vi khả lập trình, được thiết kế để làm việc trong hệ thống vi tính của hãng Intel. PPI8255 thực hiện chức năng giao diện song song giữa các thiết bị ngoại vi và máy vi tính.

Cấu hình hoạt động của PPI8255 có thể lập trình được bằng phần mềm. PPI8255 thường được dùng để thiết kế lắp ráp các mạch vào/ra dữ liệu số dạng song song.

Sơ đồ khối các thành phần chức năng của mạch PPI8255 được thể hiện trên Hình IV. 1, gồm một đệm BUS dữ liệu, khối điều khiển ghi/đọc, hai khối điều khiển hai nhóm cổng A và B, và các cổng 8bits PA, PB và PC.

Đệm BUS dữ liệu: là bộ đệm 8 bits hai chiều 3 trạng thái. Dữ liệu được phát hoặc nhận qua bộ đệm này. Từ điều khiển và trạng thái cũng được truyền từ CPU đến PPI8255 qua bộ đệm này.

Logic điều khiển và ghi/đọc: logic điều khiển và ghi/đọc quản lý toàn bộ các quá trình truyền dữ liệu và điều khiển các cổng PA, PB, PC.

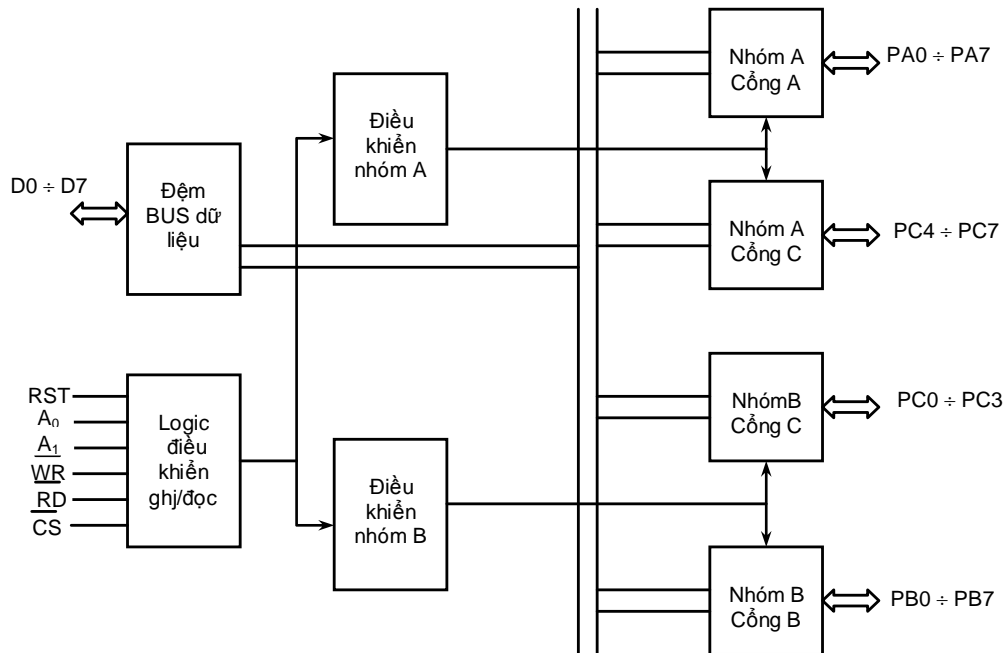
Các tín hiệu điều khiển từ CPU: Tín hiệu đọc RD, tín hiệu ghi WR và tín hiệu tái thiết lập theo mặc định RST.

RST – Reset: tín hiệu RST đặt tất cả 3 cổng A, B, C ở chế độ đầu vào.

Các tín hiệu \overline{RD} , \overline{WR} , A1, A0:

Địa chỉ A0, A1 phối hợp với tín hiệu \overline{RD} , \overline{WR} điều khiển việc ghi/đọc đối với 3 cổng A, B, C:

Các cổng PA, PB và PC : các cổng PA, PB và PC là các cổng vào/ra dữ liệu loại 8 bit. Chức năng của từng cổng được xác định bằng phần mềm (bằng từ điều khiển).



Hình IV.1 Cấu trúc theo khối chức năng PPI8255

A1	A0	\overline{RD}	\overline{WR}	Thao tác
0	0	0	1	BUS dữ liệu \leftarrow Cổng A
0	1	0	1	BUS dữ liệu \leftarrow Cổng B
1	0	0	1	BUS dữ liệu \leftarrow Cổng C
0	0	1	0	BUS dữ liệu \Rightarrow Cổng A
0	1	1	0	BUS dữ liệu \Rightarrow Cổng B
1	0	1	0	BUS dữ liệu \Rightarrow Cổng C
1	1	1	0	Thanh ghi điều khiển

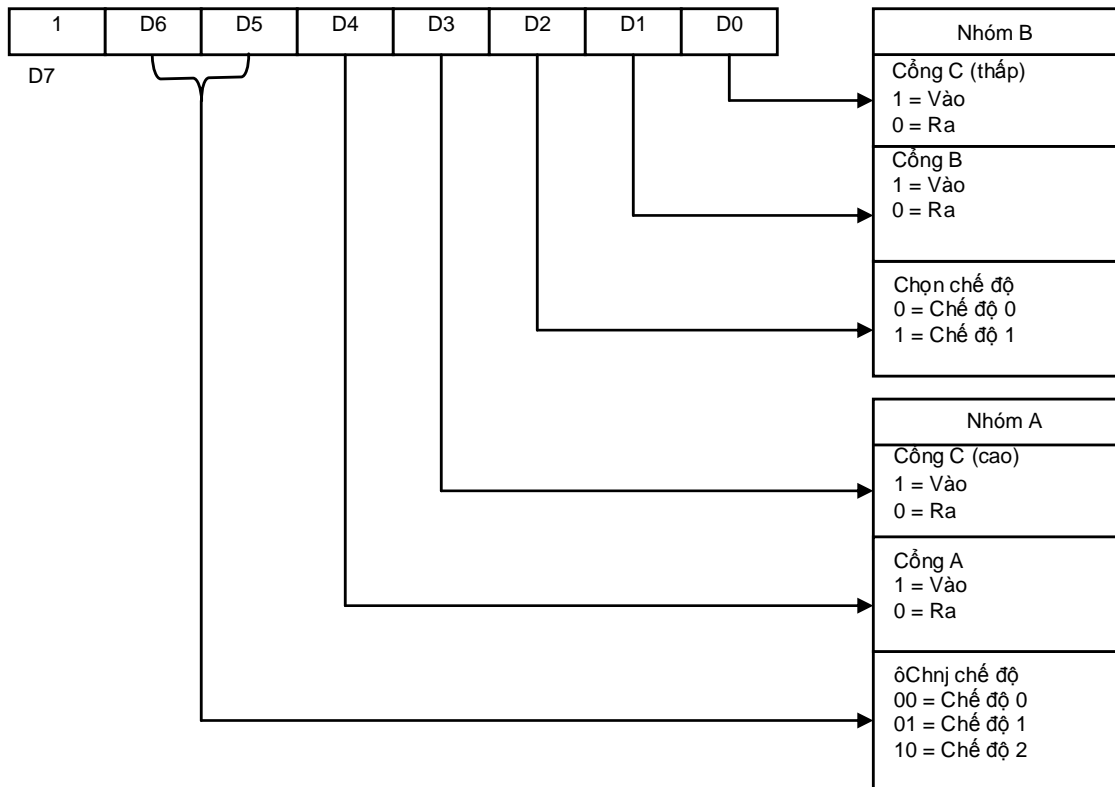
- Cổng A: là cổng ra 8 bit có chốt dữ liệu hoặc là cổng vào 8 bit.
- Cổng B: là cổng ra 8 bit có chốt dữ liệu hoặc là cổng vào 8 bit.
- Cổng C được chia thành 2 phần :
 4 bit cao (PC7÷PC4) cùng với cổng A làm thành nhóm A.
 4 bit thấp (PC3÷PC0) cùng với cổng B làm thành nhóm B.

Tùy theo chế độ hoạt động (được xác lập thông qua từ điều khiển) mà hai phần này của cổng C có thể thực hiện chức năng vào/ra dữ liệu 4 bit hoặc nhận/phát tín hiệu bắt tay cho từng nhóm tương ứng

Mạch PPI8255 có 3 chế độ làm việc

- Chế độ 0: vào/ra cơ bản
- Chế độ 1: vào/ra có xung chốt dữ liệu
- Chế độ 2: vào/ra hai chiều (chỉ cho nhóm A)

b) Chế độ làm việc và từ điều khiển



Hình IV. 2 Cấu trúc từ lệnh của PPI 8255

Có thể chọn và đặt lại chế độ làm việc của PPI8255 qua các từ điều khiển.

Khuôn dạng từ điều khiển chế độ làm việc được mô tả trên hình IV.2.

+ **Chế độ 0:** vào/ra cơ bản , ra có chốt, vào không chốt dữ liệu.

Từ điều khiển:

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1/0	x	x	x	x

Tính chất cơ bản của chế độ 0:

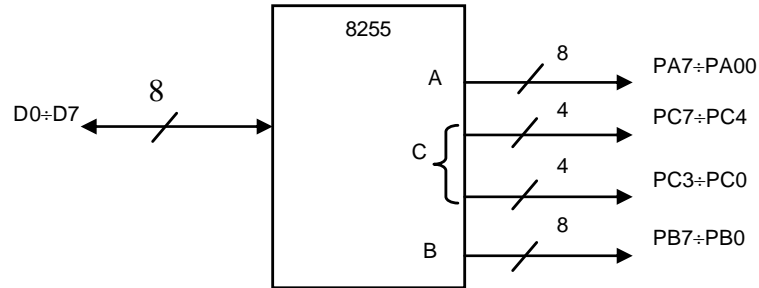
- Hai cổng 8 bit
- Hai cổng 4 bit
- Ra có chốt

- Vào không chốt
- Cho phép chọn và dùng 1 trong 16 cấu hình cổng vào/ra

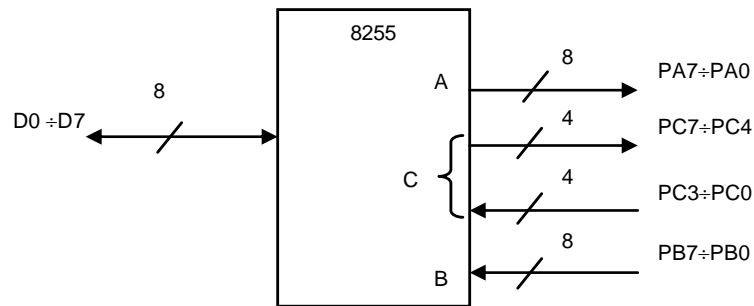
Ví dụ:

Khi đặt từ điều khiển #0, là 10000000, cấu hình các cổng của 8255 được đặt như sau:

Tất cả các cổng đều ở chế độ *Output* như ở hình vẽ



Khi đặt từ điều khiển #3, là 10000011, cấu hình các cổng của 8255 như sau



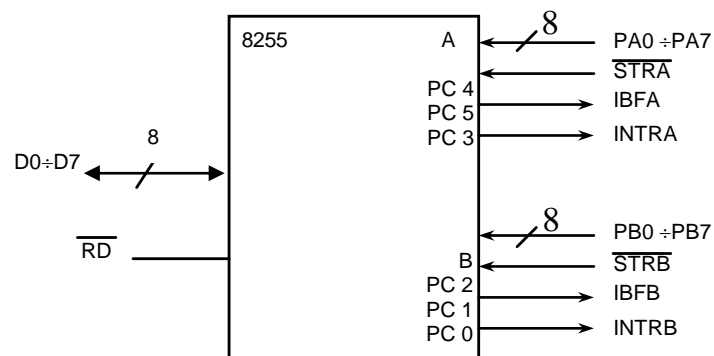
+ **Chế độ 1** : vào/ ra dữ liệu có xung chốt dữ liệu.

Đặc tính của chế độ 1: có hai nhóm A và B, mỗi một nhóm có một cổng vào/ra 8 bit và một cổng điều khiển 4 bit.

Cấu hình cổng vào dữ liệu (chế độ 1):

Từ điều khiển:

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



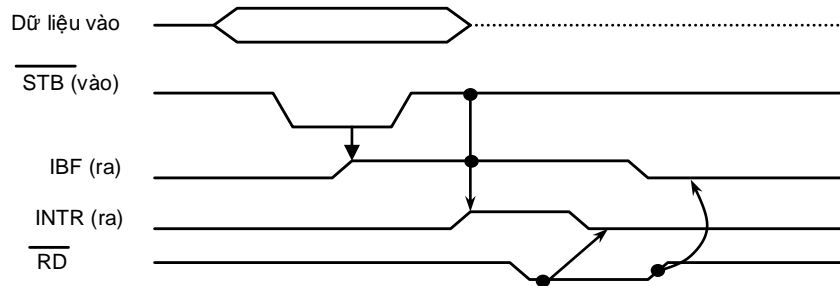
Các tín hiệu điều khiển:

$\overline{\text{STRA}}/\overline{\text{STRB}}$: mức tích cực thấp chốt dữ liệu vào 8255

IBFA/IBFB (*Input Buffer Full*): mức tích cực cao báo dữ liệu đã được chốt trong 8255

INTRA/INTRB (*Interrupt Request*): yêu cầu ngắt

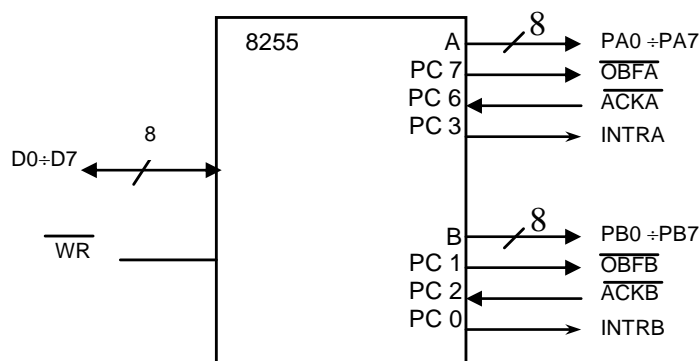
INTEA và INTEB (*Interrupt Enable*) : được đặt/xoá (1/0) bởi bit PC4 và PC2



Cấu hình cổng ra dữ liệu (chế độ 1)

Từ điều khiển:

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

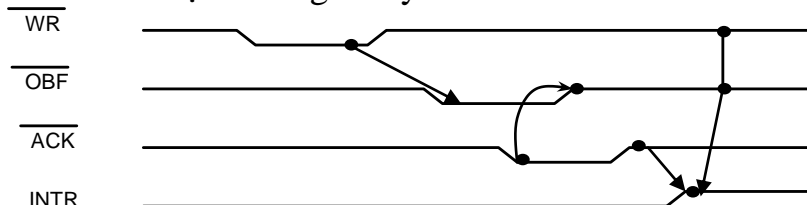


Các tín hiệu điều khiển:

$\overline{\text{OBFA}}/\overline{\text{OBFB}}$ (*Output Buffer Full*): tín hiệu ra, mức tích cực thấp khi có dữ liệu ra ở các cổng A/B.

$\overline{\text{ACKA}}/\overline{\text{ACKB}}$ (*Acknowledge*) : tín hiệu vào, mức tích cực thấp, báo 8255 là dữ liệu ra ở cổng A/B đã được nhận.

INTRA/INTB: yêu cầu ngắt, yêu cầu đưa dữ liệu (tiếp theo) ra 8255 theo tín hiệu báo ngắt này.



- + **Chế độ 2** : Vào/ ra hai chiều có xung chốt dữ liệu (riêng cho nhóm A)

Đặc tính chế độ 2: chỉ được dùng cho nhóm A. Cổng A là cổng vào/ra 8bit hai chiều. Cổng C có 5 bit được dùng làm các tín hiệu điều khiển bắt tay. Vào /ra dữ liệu đều được chốt.

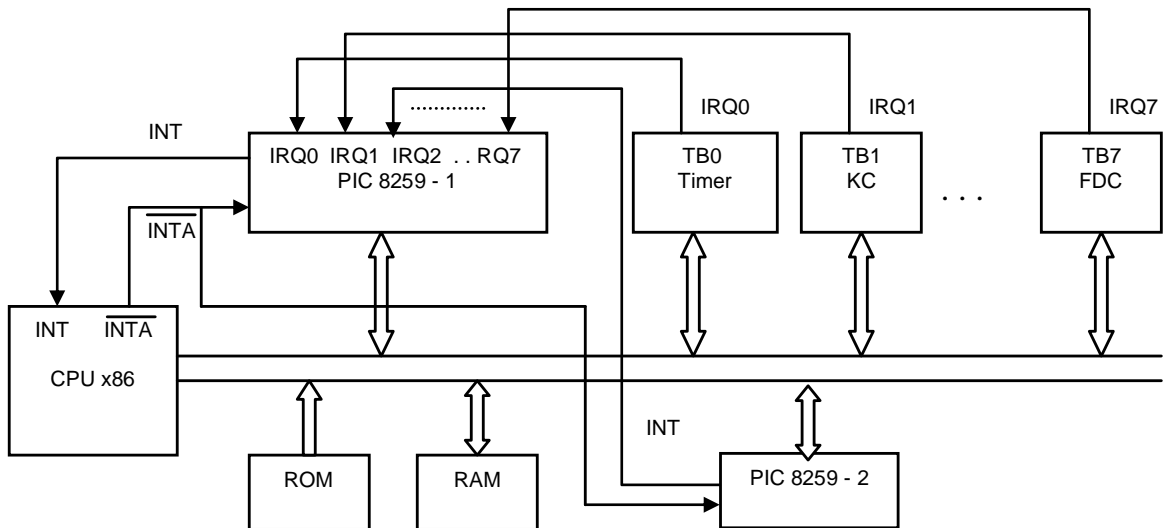
Khả năng ứng dụng: chế độ 2 cung cấp công cụ truyền tin với thiết bị ngoại vi theo cách phát và nhận dữ liệu 8 bit song song trên cùng một đường BUS. Quá trình truyền tin thuộc kiểu không đồng bộ. Các tín hiệu “bắt tay” STB, IBF, OBF, ACK được dùng để phối hợp việc truyền dữ liệu giữa máy tính và thiết bị ngoại vi.

Các bit của cổng C có thể được thiết lập lên “1” (set) hay về “0” (reset) bằng cách ghi từ điều khiển với $D_7 = “0”$, việc chọn bit cần SET hay RESET qua chọn bit $D_0 = “1”$ hoặc “0” và vị trí của bit cổng C thông qua các bit D_1, D_2 và D_3 như sau: 000 = bit PC_0 , 001 = bit PC_1 ... và 111 = bit PC_7

D7	D6	D5	D4	D3	D2	D1	D0
0	x	x	x	B2	B1	B0	S/R

IV.2.2 Mạch điều khiển ngắt PIC-8259

CPU được thiết kế để đáp ứng được với các quá trình ngắt cứng. CPU có một đầu vào nhận tín hiệu ngắt INT, khi nhận được tín hiệu này CPU sẽ phản ứng theo cơ chế ngắt cứng.



Hình IV.3 – Sơ đồ ghép nối PIC8259 trong hệ Vi xử lý

Trong thực tế có nhiều thiết bị ngoại vi yêu cầu được phục vụ theo phương pháp ngắt cứng (bàn phím, đồng hồ hệ thống, máy in, v.v.) và sinh ra nhiều yêu cầu ngắt, do vậy cần có một bộ điều khiển giúp CPU

quản lý và phục vụ các yêu cầu ngắt, đó là bộ điều khiển ngắt PIC8259 (Programmable Interrupt Controller).

Cấu trúc hệ thống ngắt cứng :

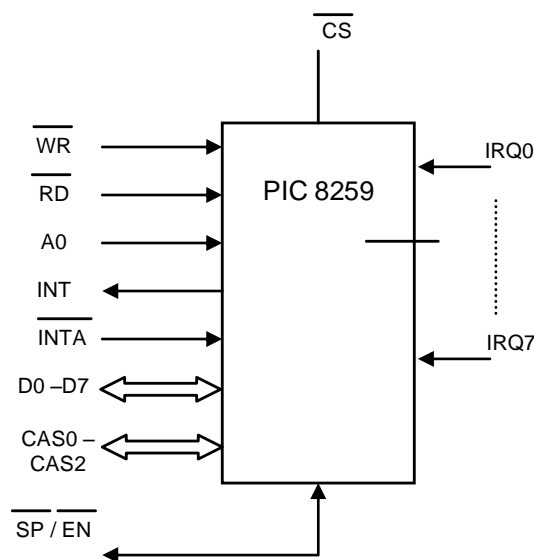
Hệ thống ngắt cứng của máy vi tính PC được xây dựng trên cơ sở 2 bộ điều khiển ngắt PIC 8259, mỗi PIC 8259 có thể nhận 8 tín hiệu yêu cầu ngắt IRQ từ thiết bị vào/ra. Hai PIC này được kết nối với nhau theo kiểu ghép tầng, kết hợp hoạt động để có thể phục vụ được 16 yêu cầu ngắt IRQ.

Chức năng cơ bản của PIC 8259 : PIC 8259 là một vi mạch điện tử khả trình được thiết kế để giúp CPU thực hiện quá trình ngắt cứng. PIC 8259 thực hiện các chức năng sau:

- Ghi nhận được 8 yêu cầu ngắt IRQ_i , $i=0,1,\dots,7$.
- Cho phép chọn và phục vụ các yêu cầu ngắt theo mức ưu tiên.
- Cung cấp cho CPU số ngắt tương ứng với yêu cầu ngắt IRQ_i . Số ngắt này đại diện cho địa chỉ của chương trình con phục vụ thiết bị yêu cầu ngắt IRQ_i .
- Cho phép hoặc không cho phép các yêu cầu ngắt IRQ_i kích hoạt hệ thống ngắt.

a) Thiết bị điều khiển ngắt PIC 8259 và cơ chế hoạt động của hệ thống ngắt cứng

Cấu trúc bên ngoài của PIC 8259 :



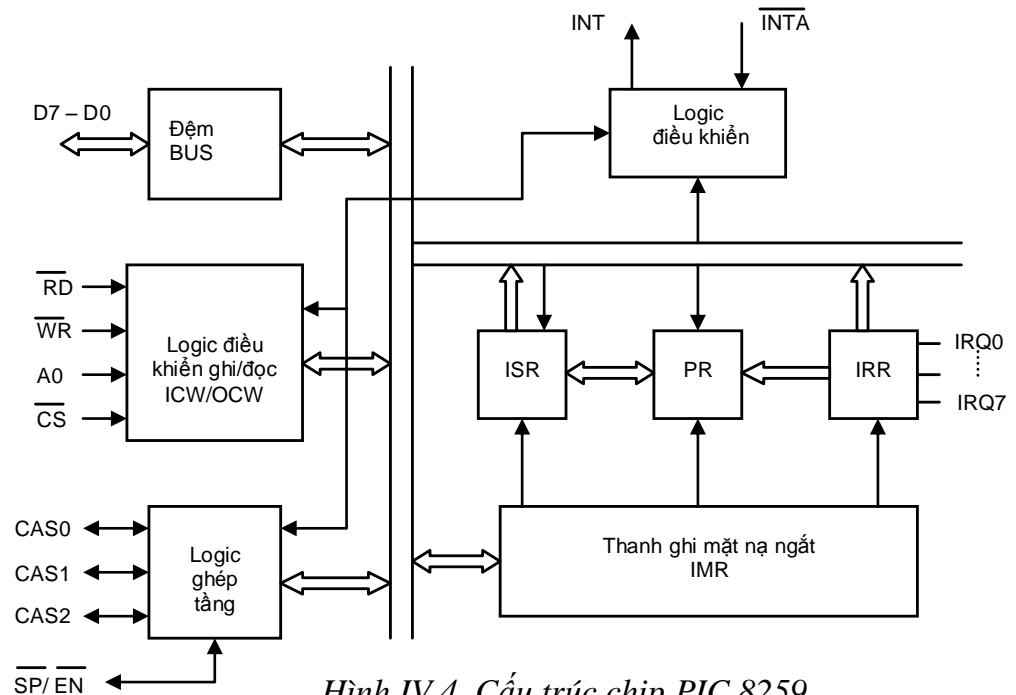
Cấu trúc bên trong của PIC 8259 :

Các khối chức năng:

- IRR (Interrupt Request Register - Thanh ghi yêu cầu ngắt) : là thanh ghi 8 bit. IRR chứa (ghi nhận) tất cả các yêu cầu ngắt

IRQi đòi phục vụ. Nếu tín hiệu $IRQi = "1"$ thì bit $IRRi$ tương ứng được đặt bằng "1".

- PR (Priority Resolver- Bộ giải quyết ưu tiên): là thanh ghi 8 bit. PR cho phép xác lập mức ưu tiên của các yêu cầu ngắt. Ngắt có ưu tiên cao nhất được chọn và đặt vào bit tương ứng trong ISR trong chu kỳ INTA.



Hình IV.4 Cấu trúc chip PIC 8259

- ISR (In Service Register - Thanh ghi ngắt đang được phục vụ) : là thanh ghi 8 bit. ISR ghi nhận các ngắt đang được phục vụ. Yêu cầu ngắt $IRQi$ nào đang được phục vụ thì bit $ISRi$ tương ứng được đặt bằng "1".
- Khối logic điều khiển : khối logic điều khiển đưa ra tín hiệu INT, được nối thẳng với chân INT của CPU. Khi INT có mức cao là đòi CPU phục vụ ngắt. Khối logic điều khiển nhận tín hiệu INTA từ CPU. Khi nhận được tín hiệu INTA, PIC 8259 sẽ cung cấp số ngắt ra BUS dữ liệu cho CPU .
- Khối đếm BUS: là loại 8 bit, 2 hướng, 3 trạng thái. Các từ điều khiển ICW, OCW được đưa vào PIC 8259 qua khối này để xác lập chế độ hoạt động của 8259. Số ngắt và trạng thái hoạt động của PIC cũng được đưa ra BUS dữ liệu qua khối này.
- Khối ghép tầng
- PIC 8259 có cơ cấu cho phép nối ghép tầng các PIC 8259 với nhau và phối hợp hoạt động của các PIC này. Tầng thứ nhất có đầu ra INT nối trực tiếp với CPU, gọi là PIC 8259-chủ. Đầu vào $IRQi$ của PIC chủ được nối với đầu ra INT của PIC 8259 thứ

hai. PIC này được gọi là PIC 8259-thợ . Cơ chế ghép tầng cho phép xây dựng một hệ thống ngắt cứng quản lý được đến 64 yêu cầu ngắt IRQ.

- Khối logic ghi/đọc và giải mã: thực hiện giải mã các từ điều khiển ICW (Initialization Command Word - Từ điều khiển khởi động) và OCW (Operation Command Word - Từ điều khiển hoạt động). Qua hai loại từ điều khiển này người sử dụng có thể lập trình xác lập chế độ hoạt động cho PIC.
- Thanh ghi IMR : là thanh ghi 8 bit, chứa mặt nạ ngắt.

Bảng các tín hiệu \overline{CS} , A0, \overline{RD} , \overline{WR} , và cách ghi/đọc PIC 8259.

CS	A0	RD	WR	D4	D3	Hướng thông tin
0	0	0	1	X	X	IRR, ISR => BUS
0	1	0	1	X	X	(IMR) = 0CW1 => BUS
0	0	1	0	0	0	BUS => 0CW2
0	0	1	0	0	1	BUS => 0CW3
0	0	1	0	1	X	BUS => ICW1
0	1	1	0	X	X	BUS => ICW2, ICW3, ICW4, 0CW1

b) Cơ chế hoạt động của hệ thống ngắt cứng :

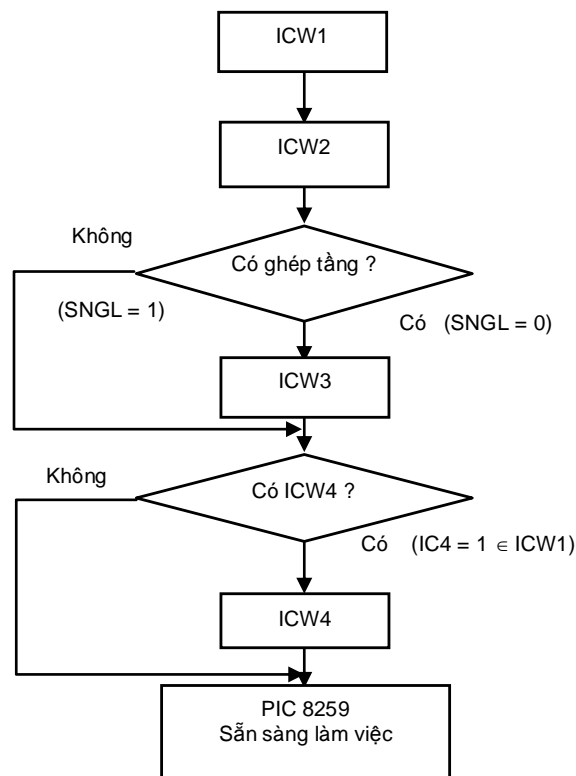
Điều kiện ban đầu : PIC 8259 cần được lập trình khởi động qua các từ điều khiển ICW. Sau khi các từ điều khiển ICW được nạp thì PIC 8259 sẵn sàng hoạt động.

- Một hoặc nhiều thiết bị vào-ra có yêu cầu được phục vụ phát tín hiệu $IRQ_i = "1"$ (mức tích cực) cho PIC. PIC ghi nhận các yêu cầu ngắt IRQ_i này bằng cách đặt các bit IRR_i tương ứng lên "1".
- PIC 8259 chọn IRQ_i có mức ưu tiên cao nhất để phục vụ. PIC gửi tín hiệu INT cho CPU, đòi CPU phục vụ.
- CPU thực hiện các thao tác sau :
 - Thực hiện nốt lệnh của quá trình hiện hành .
 - Lưu địa chỉ trở về (nội dung của các thanh ghi CS, IP) và thanh ghi cờ FLAGS vào ngăn xếp.
 - Gửi hai tín hiệu trả lời ngắt \overline{INTA} cho PIC .
- Khi PIC 8259 nhận được tín hiệu \overline{INTA} thứ 1 : bit ISR_i ứng với IRQ_i có mức ưu tiên cao nhất được thiết lập ($ISR_i=1$) và bit IRR_i tương ứng bị xóa ($IRR_i=0$). Trong chu kỳ \overline{INTA} thứ nhất này PIC 8259 không gửi gì cho CPU qua BUS dữ liệu.
- Khi PIC 8259 nhận được tín hiệu \overline{INTA} thứ 2: PIC 8259 gửi số ngắt tương ứng với IRQ_i đang được phục vụ qua BUS dữ liệu cho CPU.

- CPU nhận số ngắt và trên cơ sở số ngắt này vào vị trí tương ứng trong Bảng véc tơ ngắt để xác định địa chỉ của chương trình phục vụ ngắt. CPU nạp địa chỉ chương trình phục vụ ngắt vào các thanh ghi CS và IP và bắt đầu thực hiện chương trình phục vụ ngắt này.
- Khi thực hiện xong chương trình phục vụ ngắt thì quá trình phục vụ ngắt của CPU cũng kết thúc. Hệ thống ngắt cũng có thể kết thúc phục vụ ngắt hiện thời theo hai chế độ:
 - Kết thúc ngắt bình thường EOI (End OF Interrupt): khi PIC được đặt chế độ kết thúc ngắt bình thường EOI thì CPU phải phát lệnh báo kết thúc ngắt EOI (qua OCW2) cho PIC trước khi rời khỏi chương trình con phục vụ ngắt. Khi đó bit ISR_i của ngắt đang được phục vụ sẽ được đặt xuống 0.
 - Kết thúc ngắt tự động AEOI (Automatic EOI) : khi PIC được đặt chế độ kết thúc ngắt tự động AEOI thì tại chu kỳ INTA thứ 2 bit ISR_i của ngắt đang được phục vụ sẽ được đặt xuống 0.

Bằng các cách nói trên hệ thống ngắt cũng có thể tiếp tục phục vụ yêu cầu ngắt này ở những lần tiếp theo.

c) *Lập trình khởi động PIC 8259 và các từ điều khiển khởi động ICW*



Cần xác lập chế độ làm việc của PIC 8259 trước khi sử dụng. Quá trình này được gọi là lập trình khởi động thiết bị. Việc lập trình khởi động PIC 8259 được thực hiện qua các từ điều khiển ICW và theo lưu đồ sau :

Các bit D5 - D7 không dùng cho CPU x86.

- IC4 (bit D0) : Cho biết có cần ICW4 ?

IC4 = 0 : không cần ICW4.

IC4 = 1 : có ICW4.

a- ICW1

D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	1	LTIM	ADI	SNGL	IC4

- + SNGL (bit D1) : cho biết hệ thống ngắt chỉ có một PIC hay có nhiều PIC ghép tầng.

SNGL = 0 có ghép tầng

SNGL = 1 chỉ có một PIC 8259

- + ADI (bit D2): không dùng cho hệ CPU x86

- + LTIM : xác định dạng tín hiệu IRQ

LTIM = 1 IRQ phải là tín hiệu mức TTL

LTIM = 0 IRQ phải là tín hiệu dạng sườn xung.

- + D4 = 1

- + D5 = D6 = D7 = 0

a- ICW2:

- ICW2 định nghĩa số ngắt nền cho 7 số ngắt còn lại .

D7	D6	D5	D4	D3	D2	D1	D0
T7	T6	T5	T4	T3	x	x	x

Các bit T7 - T3 là 5 bit cao của số ngắt, 3 bit còn lại liên quan đến các đầu vào IRQ_i

Năm bit cao T7 - T3 (do người sử dụng tùy chọn) cùng với 3 bit thấp nhất bằng 0 xác định số ngắt nền . Dựa trên số ngắt nền ứng với IRQ₀ này, PIC 8259 tự tạo ra các số ngắt tiếp theo tương ứng với các IRQ₁ đến IRQ₇.

Ví dụ : ở hệ thống ngắt cứng của máy vi tính PC, các số ngắt do PIC 8259-chủ cung cấp như sau :

0	0	0	0	1	0	0	0	ứng với IRQ0
0	0	0	0	1	0	0	1	ứng với IRQ1
.....							
0	0	0	0	1	1	1	1	ứng với IRQ7

– ICW3: liên quan đến ghép tầng.

Mạch phân cứng có chân SP/EN xác định chủ/thợ ở chế độ ghép tầng : nếu SP = 1 thì PIC là chủ, nếu SP = 0 thì PIC là thợ.

Có hai loại ICW3

– ICW3 cho PIC chủ : xác định đầu vào IRQ_i nhận tín hiệu INT từ PIC thợ thứ i.

7	6	5	4	3	2	1	0
IRQ7	IRQ6	IRQ5	IRQ4	IRQ3	IRQ2	IRQ1	IRQ0

Nếu $S_i = 1$ báo có PIC thợ nối vào chân IRQ_i của chủ

– ICW3 cho PIC thợ : xác định địa chỉ (chỉ thị nhận dạng) của PIC thợ.

					ID2	ID1	ID0
--	--	--	--	--	-----	-----	-----

Các bit ID2, ID1, ID0 xác định địa chỉ riêng của các PIC 8259-thợ. Khi nhận được tín hiệu INTA2, PIC 8259-thợ so sánh các tín hiệu CAS0 - CAS2 (phát ra từ PIC 8259-chủ) với ID2 - ID0, nếu chúng giống nhau thì PIC 8259 - thợ gửi số ngắt lên BUS dữ liệu cho CPU, ngược lại thì không gửi.

– ICW4:

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	SFNM	BUFF	M/S	AEOI	μP

+ Bit μp : báo cho PIC 8259 biết phải làm việc và họ vi xử lý nào.

$\mu p = 1$: làm việc với họ x86

$\mu p = 0$: làm việc với họ 8085

+ Bit AEOI: xác lập chế độ kết thúc ngắt.

AEOI = 0 : kết thúc bình thường EOI

AEOI = 1 : kết thúc tự động AEOI

+ Bit BUFF: báo chế độ có bộ đệm BUS

BUFF = 1: PIC làm việc ở chế độ đệm BUS, lúc này tín hiệu SP/EN ở chế độ ra và việc định nghĩa chủ/thợ được xác định bằng bit M/S.

+ Bit M/S: xác định chủ/thợ

M/S = 1 : PIC là chủ

M/S = 0 : PIC là thợ.

Nếu BUFF = 0 thì M/S không có ý nghĩa.

- + Bit SFNM: bit này được đặt bằng 0 ngay khi khởi động hệ thống. Kiểu ưu tiên cố định là mặc định, trong đó IRQ₀ có mức ưu tiên cao nhất, IRQ₇ có mức ưu tiên thấp nhất. Có thể thay đổi kiểu ưu tiên bằng từ điều khiển OCW₂. Trong kiểu ưu tiên cố định, khi SFNM = 0, khi bit ISR_i = 1 thì tất cả các IRQ_i có mức ưu tiên thấp hơn đều bị cấm. Chỉ có các IRQ_i có mức ưu tiên cao hơn được phép gây ngắt chương trình phục vụ ngắt hiện thời.

d) Các từ điều khiển hoạt động OCW

Các từ điều khiển OCW được dùng để xác lập các chế độ làm việc cụ thể trong quá trình hoạt động của PIC 8259. Có thể gửi các từ OCW này cho PIC8259 vào bất kỳ lúc nào sau khi khởi động hệ thống ngắt.

- + OCW₁: cho phép hoặc cấm nhận một yêu cầu ngắt IRQ_i nào đó bằng mặt nạ ngắt .
- + Với PIC chủ : địa chỉ thanh ghi chứa OCW₁ là 21H

Với PIC thợ : địa chỉ thanh ghi chứa OCW₁ là A1H

D7	D6	D5	D4	D3	D2	D1	D0
M7	M6	M5	M4	M3	M2	M1	M0

Mỗi bit M_i tương ứng với IRQ_i

Khi M_i = 1 mặt nạ ngắt được đặt , cấm PIC nhận IRQ_i(cấm IRQ_i gây ngắt)

Khi M_i = 0 mặt nạ ngắt được xoá, cho phép PIC nhận IRQ_i (cho phép IRQ_i gây ngắt)

Hệ điều hành đặt mặt nạ che chắn các IRQ mà hệ thống chưa dùng đến.

- + OCW₂ : dùng để đổi kiểu ưu tiên và báo kết thúc ngắt EOI.

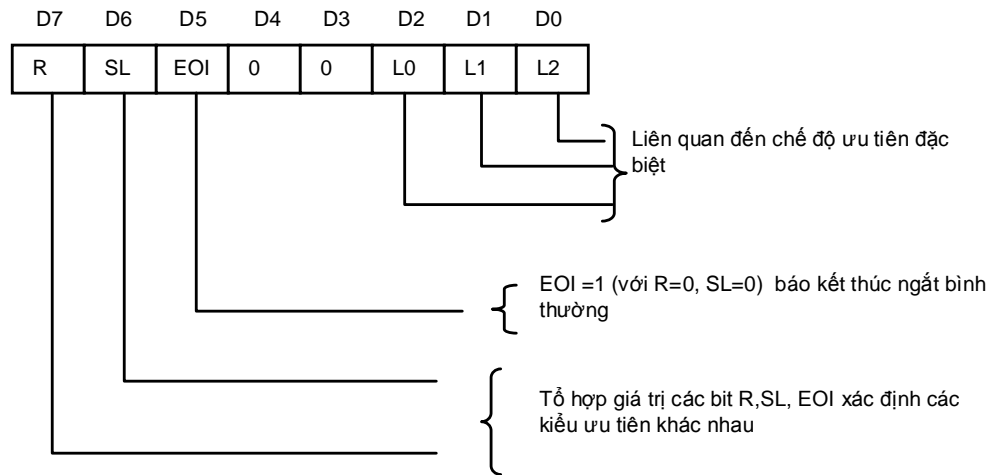
PIC cho phép chọn một trong ba chế độ ưu tiên:

Ưu tiên cố định: IRQ₀ có mức ưu tiên cao nhất, IRQ₇ có mức ưu tiên thấp nhất. Trong chế độ này IRQ mức cao có quyền ngắt chương trình phục vụ ngắt có mức ưu tiên thấp hơn.

Ưu tiên quay vòng: IRQ_i nào vừa được phục vụ thì bit ISR_i sẽ bị xoá xuống 0 và tự động có mức ưu tiên thấp nhất. Điều này thực tế đã tạo ra các mức ưu tiên bằng nhau.

Ưu tiên đặc biệt : người lập trình có thể thay đổi mức ưu tiên bằng chương trình. Nếu các bit trong OCW2 R=1, SL=1 thì các bit L2-L0 sẽ đặt IRQ_n xuống mức thấp nhất và IRQ_{n+1} lên mức cao nhất.

Địa chỉ thanh ghi chứa OCW2 : 20H (PIC chủ), A0H (PIC thợ)



+ OCW3 cho phép đặt/đọc ISR và các thanh ghi khác của PIC 8259. ESMM =1 và SMM cho phép đặt/xoá chế độ mặt nạ đặc biệt. Chế độ mặt nạ đặc biệt này chỉ cấm một IRQ và cho phép tất cả các IRQ còn lại được yêu cầu ngắt.

- D4 = 0 , D3 = 1
- bit P : cho phép PIC 8259 làm việc với CPU ở chế độ hỏi đáp, không cần qua các tín hiệu INTR, INTA. Nếu P=1 thì PIC coi tín hiệu điều khiển đọc RD như là tín hiệu INTA .
- Các bit RR và RIS :

RR = 1 & RIS = 0: báo sẽ đọc IRR ở lệnh đọc tiếp sau

RR = 1 & RIS = 1: báo sẽ đọc ISR ở lệnh đọc tiếp sau.

e) Phân bố chức năng các yêu cầu ngắt và số ngắt trong máy PC

•PIC 8259-chủ :

PIC 8259-chủ chiếm hai địa chỉ cổng : 20H, 21H

•PIC 8259-thợ :

PIC 8259-thợ chiếm hai địa chỉ cổng : A0H, A1H

IRQ _i	Số ngắt	Thiết bị yêu cầu ngắt
IRQ ₀	08H	Bộ tạo xung nhịp đồng hồ hệ thống
IRQ ₁	09H	Thiết bị giao diện bàn phím
IRQ ₂	0AH	PIC 8259-thợ
IRQ ₃	0BH	Thiết bị giao diện vào/ra nối tiếp 2 (COM 2)
IRQ ₄	0CH	Thiết bị giao diện vào/ra nối tiếp 1 (COM 1)
IRQ ₅	0DH	Dự phòng
IRQ ₆	0EH	Thiết bị giao diện ổ đĩa mềm FDC
IRQ ₇	0FH	Thiết bị giao diện vào/ra song song (LPT1)

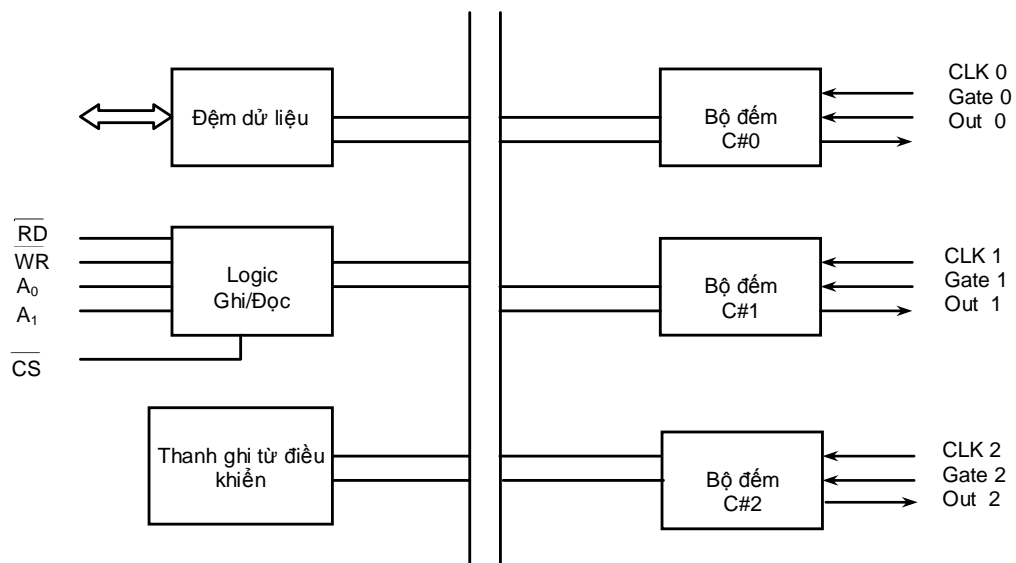
Dây IRQ	Số ngắt	Thiết bị yêu cầu ngắt
IRQ ₈	70H	Đồng hồ thời gian thực
IRQ ₉	71H	Dự phòng
IRQ ₁₀	72H	Card âm thanh
IRQ ₁₁	73H	Thiết bị giao diện vào/ra USB
IRQ ₁₂	74H	Thiết bị giao diện chuột PS/2
IRQ ₁₃	75H	Bộ đồng xử lý x87
IRQ ₁₄	76H	Bộ điều khiển BUS IDE 1(primary)
IRQ ₁₅	77H	Bộ điều khiển BUS IDE 2 (secondary)

IV.3.3 Mạch đếm định thời đa năng PIT-8253 (Programmable Interval Timer)

Vi mạch PIT-8253 là mạch đếm định thời, tạo xung có độ rộng thay đổi đa năng và thu thập tín hiệu dạng xung., được thiết kế để sử dụng với hệ vi xử lý dòng Intel. Mạch 8253 thực hiện được nhiều chức năng. Các chức năng được xác định bằng phần mềm thông qua từ điều khiển.

Các chức năng chính của PIT-8253:

- Tạo khoảng thời gian chính xác
- Phát xung với tần số f lập trình được
- Đếm sự kiện
- Chia tần số
- Đồng hồ thời gian thực
- Phát xung đơn



Hình IV.5 – Sơ đồ cấu trúc bên trong PIT-8255

Đệm dữ liệu: là bộ đếm 8 bit, hai chiều, 3 trạng thái, được sử dụng để giao diện với BUS của máy tính.

Logic đọc/ghi: logic Ghi / Đọc nhận các tín hiệu từ hệ thống BUS , từ đó điều khiển việc truy nhập các thanh ghi của PIT-8253.

Thao tác chọn bộ đếm và ghi/đọc bộ đếm:

A1	A0	RD	WR	Công việc được thực hiện
0	0	1	0	Nạp bộ đếm C#0
0	1	1	0	Nạp bộ đếm C#1
1	0	1	0	Nạp bộ đếm C#2
1	1	1	0	Ghi từ điều khiển
0	0	0	1	Đọc bộ đếm C#0
0	1	0	1	Đọc bộ đếm C#1
1	0	0	1	Đọc bộ đếm C#2

Thanh ghi điều khiển: thanh ghi điều khiển nhận từ điều khiển xác định chế độ hoạt động cho PIT-8253.

Bộ đếm C#0, C#1, C#2:

Mạch PIT-8253 có 3 bộ đếm, mỗi một bộ đếm là loại 16bit, đếm lùi, tự khởi động lại. Mỗi một bộ đếm có thể được lập trình và hoạt động độc lập. Có thể đọc nội dung của từng bộ đếm ngay trong khi đang hoạt động. Bằng từ điều khiển có thể chọn chế độ làm việc cho các bộ đếm (6 chế độ).

1. Từ điều khiển.

Từng bộ đếm của PIT-8253 có thể được lập trình hoạt động độc lập bằng cách ghi từ điều khiển vào thanh ghi từ điều khiển (A0=1, A1=1).

Khuôn dạng từ điều khiển.

DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0
SCSC1	SCSC0	RLRL1	RLRL0	MM2	MM1	MM0	BCDCD

SC (Select Counter): chọn bộ đếm.

SC1	SC0	
0	0	Bộ đếm 0
0	1	Bộ đếm 1
1	0	Bộ đếm 2
1	1	Không hợp lệ

RL (Read/Load): Xác định Đọc/Nạp bộ đếm.

RLRL1	RLRL0	
0	0	Thao tác chốt bộ đếm Cho phép đọc nội dung bộ đếm trong quá trình đếm
1	0	Đọc hoặc nạp Byte cao
0	1	Đọc hoặc nạp Byte thấp
1	1	Đọc hoặc nạp Byte thấp trước, Byte cao sau

M (Mode): chế độ làm việc

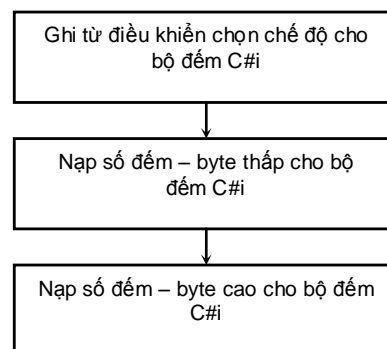
MM2	MM1	MM0	
0	0	0	Chế độ 0
0	0	1	Chế độ 1
X	1	0	Chế độ 2
X	1	1	Chế độ 3
1	0	0	Chế độ 4
1	0	1	Chế độ 5

BCD : kiểu mã đếm

BCD	Kiểu mã đếm
0	Mã nhị phân 16 bit
1	Mã BCD (4 chữ số BCD)

Nạp nội dung bộ đếm: thanh ghi đếm chỉ được nạp khi cả hai byte giá trị đếm được ghi.

2. Thủ tục xác lập chế độ làm việc cho bộ đếm.



3. Các chế độ làm việc

Chế độ làm việc xác định cách đáp ứng của đầu ra Output đối với đầu vào là dãy xung CLK và tín hiệu Gate.

Bộ đếm thực hiện đếm lượng chu kỳ xung tính từ nửa thấp của chu kỳ đầu tiên

Có 6 chế độ làm việc.

a. Chế độ 0:

Tạo khoảng thời gian trễ xác định và đặt đầu ra Output = “1” khi kết thúc đếm.

Đầu Output = “0” ngay sau khi chọn chế độ.

Ngay sau khi số đếm được nạp thì bắt đầu đếm. Điều kiện làm việc là Gate = “1”

Khi kết thúc đếm thì Output = “1” và giữ nguyên cho đến khi được nạp lại.

Việc nạp lại số đếm gây ra hai sự kiện:

- + Ghi byte đầu tiên làm dừng đếm.
- + Ghi byte thứ hai làm khởi đầu lần đếm mới.

b. Chế độ 1

Tạo xung đơn có độ rộng xác định.

Đầu ra Output = “0” khi GATE = “1” và bắt đầu đếm.

Output = “1” khi kết thúc đếm.

Việc nạp lại số đếm trong khi Output = “0” không làm ảnh hưởng tới độ rộng xung đầu ra.

Việc đếm được khởi đầu lại (xung Output bị kéo dài) nếu GATE = “0” và sau đó GATE = “1”.

c. Chế độ 2

Bộ chia tần – phát xung.

Bộ đếm được dùng như một bộ chia tần. Nội bộ đếm được nạp xác định hệ số chia. Chu kỳ dãy xung đầu ra, tính từ một xung đầu ra Output = “0” đến một xung Output = “0” tiếp theo đúng bằng số lượng xung vào CLK.

Điều kiện làm việc là $GATE \equiv “1”$.

Độ rộng mức “0” của xung ra đúng bằng chu kỳ T của xung CLK.

Có thể dùng tín hiệu GATE để đồng bộ quá trình đếm – phát xung.

d. Chế độ 3

Bộ chia tần – phát xung vuông.

Làm việc giống chế độ 2, chỉ khác ở chỗ là độ rộng mức “0” bằng độ rộng mức “1”.

Điều kiện làm việc là GATE = “1”.

e. Chế độ 4

Tạo xung chốt bằng phần mềm.

Sau khi đặt chế độ làm việc thì Output = “1”.

Sau khi số đếm được nạp thì bắt đầu đếm. Điều kiện làm việc là GATE = “1”.

Đầu ra Output = “0” khi kết thúc đếm, độ rộng xung đầu ra bằng độ rộng chu kỳ xung CLK.

Việc nạp lại số đếm trong khi đếm làm khởi động lại việc đếm.

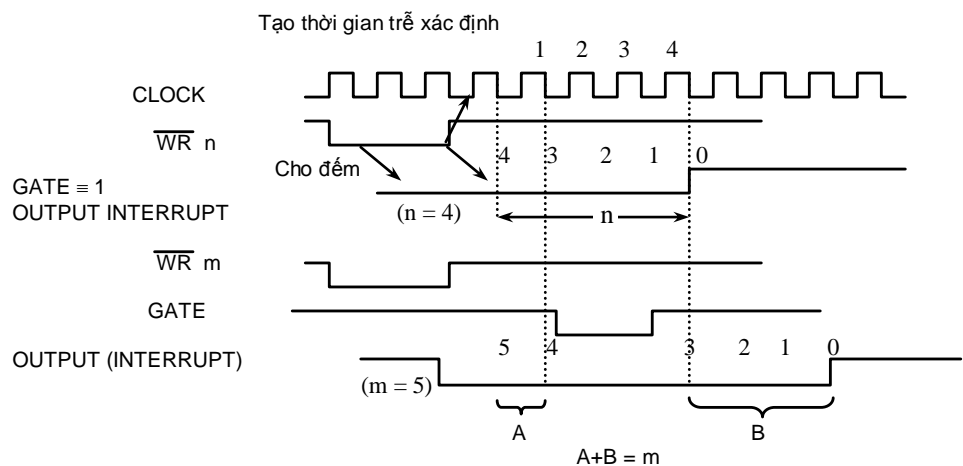
f. Chế độ 5

Tạo xung chốt bằng phần cứng.

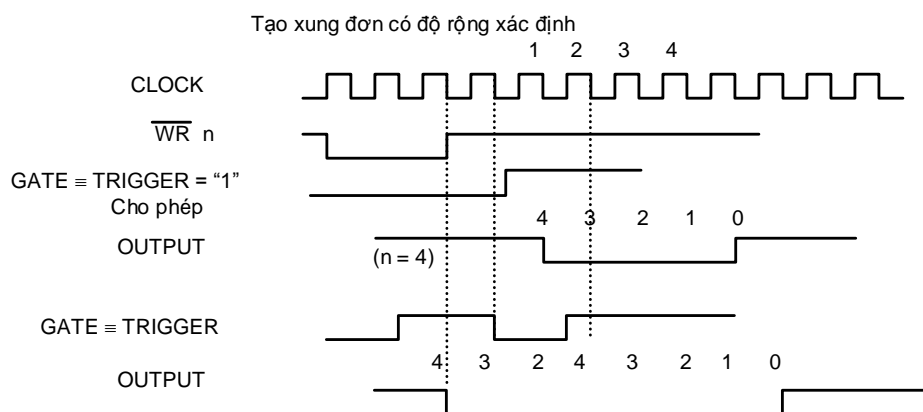
Sau khi đặt chế độ thì Output = “1”.

Bắt đầu đếm khi GATE = “1”. Đầu ra Output = “0” khi kết thúc đếm, độ rộng xung đầu ra bằng độ rộng chu kỳ xung CLK.

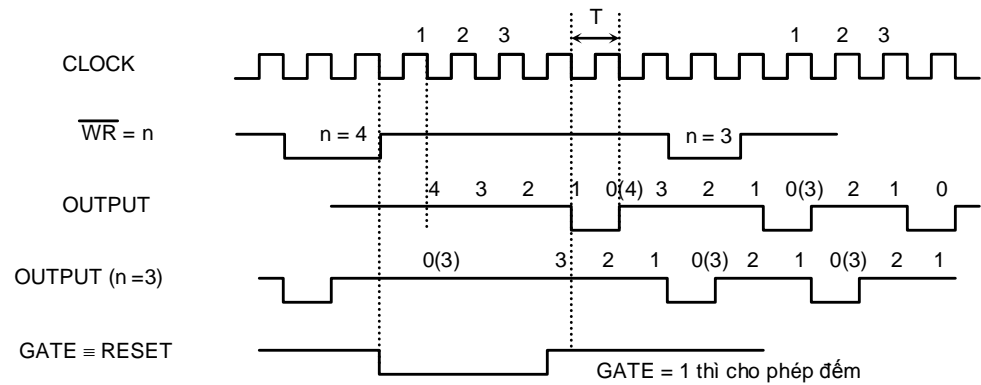
MODE 0: Interrupt on Terminal Count with GATE = 1



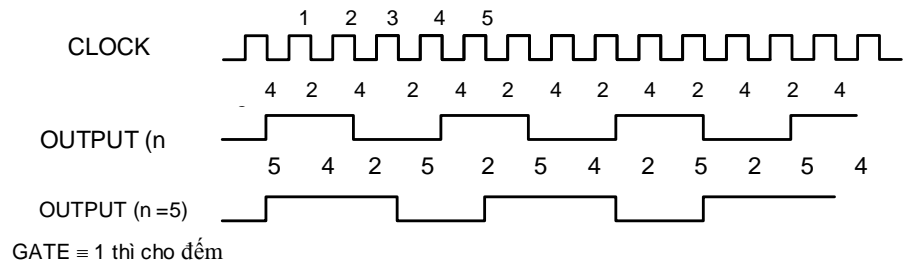
MODE 1: Programmable One-Shot.



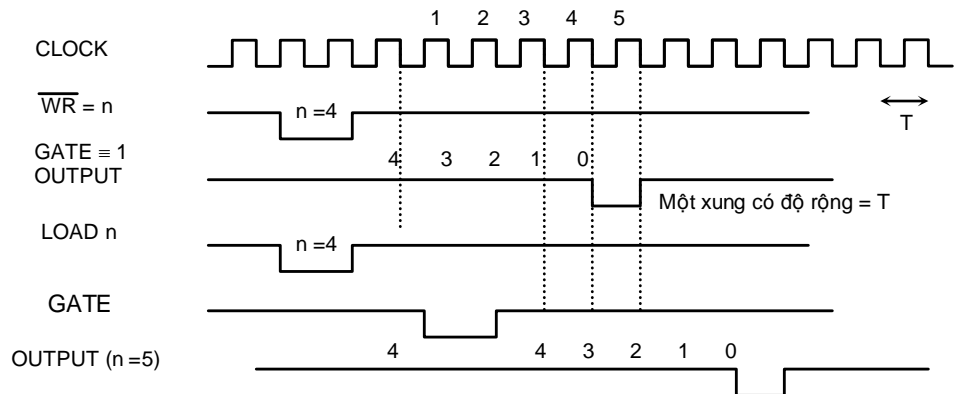
MODE 2: Rate Generator. (with GATE=1)



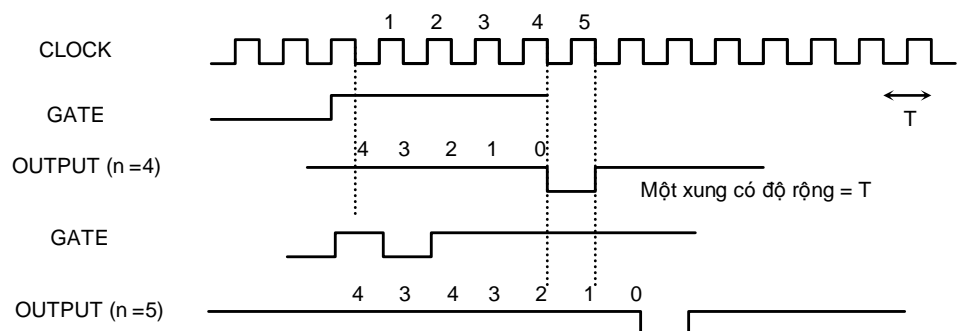
MODE 3: Square Wave Generator (with GATE=1)



MODE 4: Software Triggered Strobe (with GATE=1)



MODE 5: Hardware Triggered Strobe



4. Khả năng đọc nội dung bộ đếm trong khi đếm (Đọc trong khi đếm). Để thực hiện được thao tác đọc trong khi đếm thì cần nạp từ điều khiển đặc biệt vào thanh ghi có địa chỉ A1, A0 = 11 .

D7	D6	D5	D4	D3	D2	D1	D0
SC1	SC0	0	0	X	X	X	X

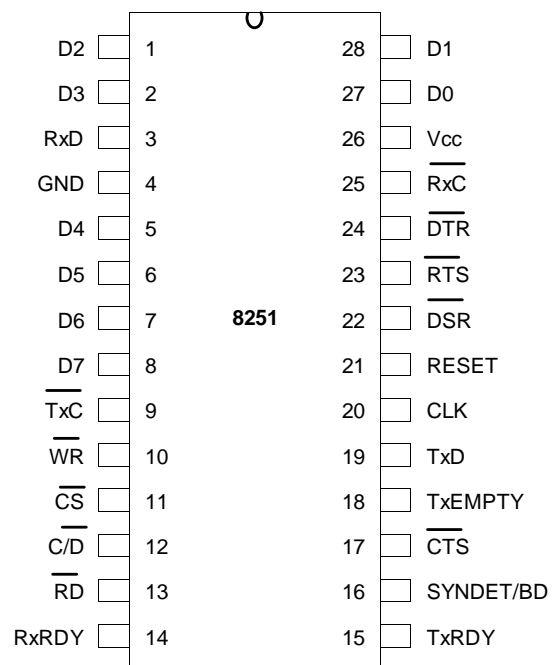
SC1, SC0: Bộ đếm được chọn đặt chế độ đọc trong khi đếm.
 D5, D4: (00, Mã xác định chế độ đọc trong khi ghi)
 X Không tác động

IV.4.4 Mạch điều khiển vào/ra nối tiếp đồng bộ/dị bộ USART-8251 (Universal Synchronous/Asynchronous Receiver Transmitter)

USART-8251 là một mạch giao diện vào/ra khả lập trình của hãng Intel. Các tính năng chủ yếu của mạch bao gồm:

- Hoạt động ở một trong hai chế độ đồng bộ hoặc không đồng bộ
- Hoạt động đồng bộ với mã 5 – 8 bits, ký tự đồng bộ nội bộ hoặc từ bên ngoài, có chế độ đồng bộ tự động

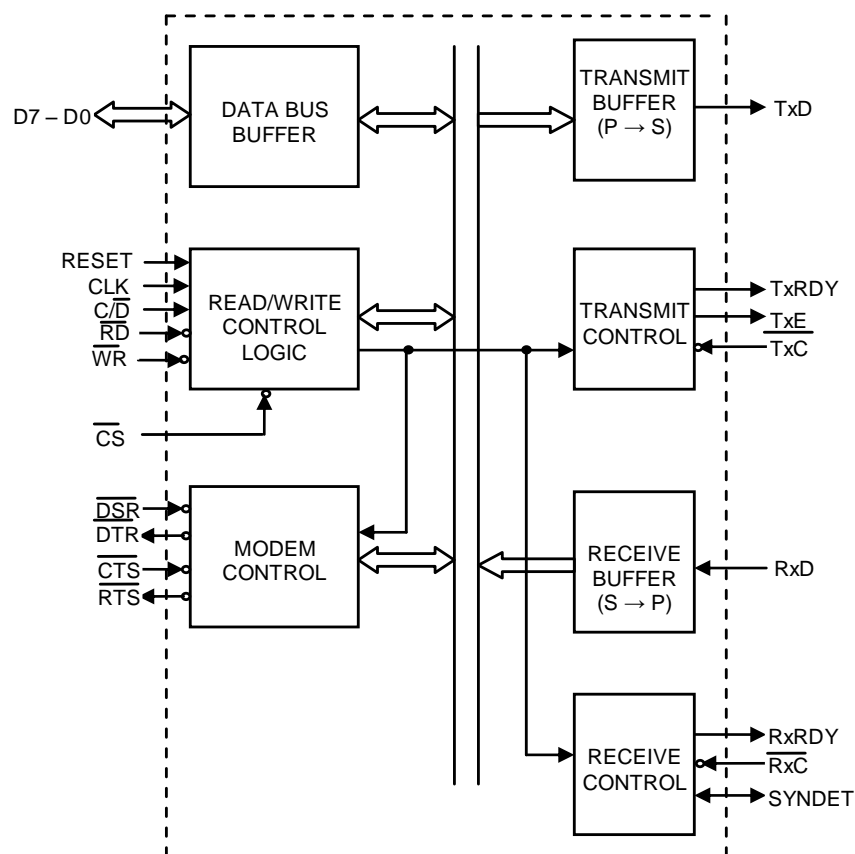
Hình IV.6a - Sơ đồ nối chân của USART 8251



- Hoạt động không đồng bộ với mã 5 – 8 bits, hệ số xung nhịp 1, 16 hoặc 64 lần tốc độ Baud, tạo ký tự tạm dừng; với 1, $1\frac{1}{2}$ và 2 bits Stop, phát hiện lỗi bit Start và ký tự Break (tạm dừng)
- Khả năng tự phát hiện lỗi thu phát

- Tương thích hoàn toàn với các chip họ 80x86

Mạch USART 8251 được thiết kế cho mục đích trao đổi dữ liệu nối tiếp giữa CPU và các thiết bị ngoại vi. Người lập trình có thể chọn các phương thức thu/phát dữ liệu đồng bộ hoặc không đồng bộ, chọn tốc độ thu/phát phù hợp thông qua các từ điều khiển (Control Word - CW). Bản thân USART thực hiện công việc chuyển đổi dữ liệu từ CPU thành dữ liệu nối tiếp để gửi ra thiết bị ngoại vi, đồng thời, mạch cũng tự chuyển dữ liệu nối tiếp thu nhận được thành dữ liệu song song để chuyển cho CPU. Chức năng của USART 8251 là làm trung gian cho việc giao tiếp với thiết bị ngoại vi (interFacing) bằng thu/phát dữ liệu nối tiếp, còn bản thân dữ liệu trao đổi giữa CPU và USART 8251 vẫn là giao diện song song.



Hình IV.6b - Sơ đồ cấu trúc bên trong của USART 8251

a) *Mô tả chức năng:*

Cũng như các mạch chức năng khác trong hệ thống 80x86, cấu hình chức năng của mạch USART-8251 rất uyển chuyển nhờ được thiết lập bằng phần mềm. Trong môi trường trao đổi dữ liệu, giao diện nối tiếp thực hiện việc biến đổi dữ liệu sóng song của hệ thống thành dạng dữ liệu nối tiếp để gửi đi và biến dạng dữ liệu nối tiếp thu nhận được thành dữ liệu sóng song để CPU đọc vào. Tất nhiên, khi thực hiện công việc biến đổi, USART-8251 sẽ tự động bỏ đi hoặc thêm vào các bit hoặc ký tự

đồng nhất chức năng trong kỹ thuật thu phát thông tin. Chính nhờ vậy, giao diện giữa CPU và USART-8251 là hoàn toàn minh bạch, chỉ đơn thuần là gửi đi hay nhận về một byte dữ liệu.

+ **Đệm BUS dữ liệu:**

Là bộ đệm 3 trạng thái hai chiều với độ rộng 8 bits dùng làm giao diện giữa CPU và mạch 8251. Dữ liệu được gửi đi hay nhận về qua bộ đệm khi thực hiện lệnh INPUT hay lệnh OUTPUT trong CPU. Các từ lệnh (Command Word), từ điều khiển (Control Word) hay thông tin trạng thái cũng được chuyển qua thanh đệm dữ liệu. Thanh ghi trạng thái lệnh (Command Status Register), thanh ghi dữ liệu ra (Data Out Register) và thanh ghi dữ liệu vào (Data In Register) là những thanh ghi độc lập và cùng được kết nối BUS dữ liệu của hệ thống thông qua đệm dữ liệu.

+ **RESET:**

Mức “1” logic ở đầu vào này đưa 8251 về chế độ nghỉ. Chế độ này tồn tại cho đến khi một chuỗi từ điều khiển, từ lệnh mới được gửi tới 8251 để xác định chế độ làm việc. Mức “1” này phải tồn tại trong khoảng thời gian ngắn nhất của 6 chu kỳ xung nhịp hệ thống.

+ **CLK (Clock):**

Là đầu vào xung nhịp cho 8251 làm việc. Tần số xung nhịp này phải lớn hơn tối thiểu 30 lần so với tốc độ thu/phát của 8251.

+ **\overline{WR} (ghi):**

Mức “0” logic xuất hiện ở đầu vào này là xung điều khiển từ CPU trong việc ghi từ điều khiển hoặc gửi dữ liệu cho 8251.

+ **\overline{RD} (đọc):**

Mức “0” logic xuất hiện ở đầu vào này là xung điều khiển từ CPU trong việc đọc trạng thái của 8251 hoặc đọc dữ liệu từ 8251 vào CPU.

+ **C/\overline{D} :**

Đầu vào điều khiển, kết hợp với các tín hiệu vào gồm \overline{CS} , \overline{WR} và \overline{RD} xác định cho 8251 dữ liệu tồn tại trên BUS là ký tự dữ liệu, từ điều khiển hay thông tin trạng thái. “1” ứng với CONTROL/STATUS, “0” ứng với DATA.

C/\overline{D}	\overline{RD}	\overline{WR}	\overline{CS}	
0	0	1	0	8251 DATA → DATA BUS
0	1	0	0	DATA BUS → 8251 DATA
1	0	1	0	STATUS → DATA BUS
1	1	0	0	DATA BUS → CONTROL
X	1	1	0	DATA BUS → TRI-STATE
X	X	X	1	DATA BUS → TRI-STATE

Lưu ý: Chân C/\overline{D} thường được nối với dây địa chỉ A0 của BUS địa chỉ, do vậy có thể dễ dàng phân biệt hai địa chỉ duy nhất của 8251 là: Địa chỉ nền là địa chỉ đọc hoặc ghi dữ liệu, địa chỉ nền + 1 là địa chỉ cho ghi từ điều khiển và đọc trạng thái.

+ **\overline{CS} (Chip Select):**

Tín hiệu chọn vô đối với 8251. Mức “0” là tích cực, chip 8251 được chọn. Khi $\overline{CS} = “1”$, các tín hiệu đọc (RD) và ghi (WR) không có tác động đối với 8251.

+ **Modem Control:**

Vi mạch 8251 có một tập tín hiệu vào/ra có thể sử dụng để đơn giản hoá việc phối ghép với các MODEM. Các tín hiệu do khối chức năng điều khiển Modem tạo ra nhằm mục đích hoàn toàn tương thích với các tín hiệu điều khiển trao đổi thông tin thông qua thiết bị Modem khi cần thiết. Đó là các tín hiệu DSR (Data Set Ready), DTR (Data Terminal Ready), RTS (Request To Send), và CTS (Clear To Send).

+ **Đệm phát (Transmitter BuFFer):**

Đệm phát tiếp nhận dữ liệu song song từ đệm dữ liệu, chuyển đổi thành chuỗi bits nối tiếp, chèn thêm các ký tự hoặc các bit thích hợp cần thiết trong kỹ thuật truyền tin và gửi chuỗi bits này ra đầu phát TxD theo sườn xuống của xung nhịp phát TxC. Khối phát bắt đầu công việc ngay khi tín hiệu $\overline{CTS} = “0”$ và dừng lập tức với trạng thái được giữ nguyên khi TxE là “0” hay $\overline{CTS} = “1”$.

+ **Điều khiển phát (Transmitter Control):**

Khối điều khiển phát giám sát toàn bộ mọi hoạt động liên quan đến truyền dữ liệu nối tiếp. Khối có nhiệm vụ chấp nhận và tạo ra tất cả các tín hiệu tương ứng để thực hiện việc truyền dữ liệu.

+ **TxRDY (Transmitter Ready):**

Tín hiệu ra của 8251 thông báo cho CPU biết nó sẵn sàng nhận dữ liệu để truyền đi. Tín hiệu này có thể sử dụng làm tín hiệu yêu cầu ngắt đối với hệ thống, và khác với tín hiệu TxE (Transmitter Empty). Trong chế độ phát có thăm dò, CPU có thể thông qua tín hiệu TxRDY để quyết định chuyển dữ liệu cho 8251. Tín hiệu này bị Reset bởi tín hiệu WR khi dữ liệu được gửi tới 8251 từ CPU.

Lưu ý rằng, trong chế độ phát theo thăm dò, tín hiệu TxRDY không bị ràng buộc bởi tín hiệu TxE, nó chỉ có tác dụng thông báo trạng thái đầy hay rỗng của thanh ghi đệm phát.

+ **TxE (Transmitter Empty):**

Khi 8251 chuyển xong một ký tự. “không còn gì để phát đi”, đầu ra TxE sẽ chuyển đổi lên mức “1” logic. Có thể thông qua tín hiệu này để biết được trạng thái kết thúc truyền của 8251, đặc biệt trong chế độ half-duplex.

Trong chế độ thu phát đồng bộ giá trị “1” ở đầu ra này chỉ ra rằng chưa có dữ liệu được truyền đi, ký tự SYNC hoặc là ký tự dữ liệu sắp sửa

được truyền. TxE không thay đổi mức khi ký tự SYNC bắt đầu được gửi đi.

+ **$\overline{\text{TxC}}$ (Transmitter Clock):**

Xung nhịp phát điều khiển tốc độ truyền các ký tự. Ở chế độ thu phát đồng bộ, tốc độ Baud Rate (1x) bằng chính tần số $\overline{\text{TxC}}$. Trong chế độ thu phát không đồng bộ tốc độ này luôn theo một tỷ lệ tương ứng của Baud Rate. Các tỷ lệ thường được sử dụng là 1x, 16x hoặc 64x tốc độ Baud Rate. Sườn xuống của $\overline{\text{TxC}}$ dịch chuyển dữ liệu nối tiếp ra chân TxD của 8251.

+ **Đệm thu (Receiver Buffer):**

Đệm thu thu nhận dữ liệu nối tiếp và chuyển đổi thành dữ liệu song song sau khi đã loại bỏ những ký tự hoặc bit tương ứng sử dụng trong kỹ thuật thu phát thông tin. Tín hiệu thu được đưa qua chân RxD và được dịch chuyển vào thanh ghi đệm thu theo sườn lên của xung $\overline{\text{RxC}}$.

+ **Khởi điều khiển thu (Receiver Control):**

Khởi này giám sát và điều khiển mọi hoạt động liên quan đến việc nhận dữ liệu nối tiếp. Các tính năng chủ yếu của khởi này như sau:

- Trong điều kiện nghỉ, RxD ngăn mọi tín hiệu “low”. Trước khi bắt đầu nhận dữ liệu nối tiếp, trên chân này phải được khẳng định giá trị “1” logic, từ đó, khởi bắt đầu dò tìm giá trị “0” có nghĩa, tương ứng với Start Bit
- Mạch nhận biết bit Start bằng cách loại trừ mọi tín hiệu nhiễu thông qua dò tìm sườn xuống trên RxD và xác định bit Start cho việc thu nhận dữ liệu.
- Phát hiện lỗi chẵn lẻ thông qua bit trạng thái chẵn lẻ
- Phát hiện lỗi khung dữ liệu thông qua bit Stop ở cuối byte dữ liệu cuối cùng trong chế độ thu phát không đồng bộ.

+ **RxRDY (Receiver Ready):**

Tín hiệu ra RxRDY báo rằng 8251 đã nhận xong một ký tự và đang sẵn sàng chuyển cho CPU. RxRDY cũng có thể nối vào chân yêu cầu ngắt đối với CPU trong phương pháp vào/ra theo ngắt, hoặc làm tín hiệu báo trạng thái trong phương pháp vào/ra thăm dò (polled operation). Tín hiệu RxEnable khi là “oFF”, sẽ giữ cho RxRDY ở điều kiện tái khởi động. Thanh ghi đệm vào phải được phép dò tìm bit Start của dữ liệu mới và ký tự hoàn chỉnh đã được nhận phải được gửi vào thanh ghi dữ liệu ra. Khi xảy ra sự cố đọc ký tự đã nhận được từ thanh ghi dữ liệu ra, chip sẽ tạo lỗi Overrun, ký tự vừa nhận sẽ bị bỏ qua.

+ **$\overline{\text{RxC}}$ (Receiver Clock):**

Xung nhịp nhận tạo lập tốc độ thu dữ liệu. Dữ liệu được ghi nhận từng bit theo sườn lên của xung nhịp $\overline{\text{RxC}}$. Trong chế độ thu phát đồng bộ,

tần số \overline{RxC} bằng đúng tần số của xung Baud Rate. Còn trong chế độ thu phát không đồng bộ, tần số xung này được lấy theo tỷ lệ 1x, 16x hoặc 64x tần số tốc độ Baud Rate. Có thể lấy ví dụ:

Baud Rate là 2400 Baud, yêu cầu đối với xung nhịp \overline{RxC} là

$$\overline{RxC} = 2400\text{Hz ở chế độ 1x}$$

$$\overline{RxC} = 38,4\text{KHz ở chế độ 16x và}$$

$$\overline{RxC} = 153,6\text{KHz ở chế độ 64x.}$$

Lưu ý rằng, tốc độ Baud Rate là một tốc độ phải chọn theo quy chuẩn quốc tế, thông thường là 300, 600, 1200, 2400, 4800, 9600, 19200 Baud, v.v..., chứ không phải là một số bất kỳ, nên việc tạo xung tần số cho \overline{RxC} và \overline{TxC} thường được sử dụng những thạch anh có tần số là bội 16, bội 64 của chuỗi số trên với độ chính xác rất cao, chứ không sử dụng tùy tiện. Hơn nữa, trong phần lớn các hệ thống thu phát thông tin, tốc độ thu và tốc độ phát là như nhau, dẫn đến tần số \overline{RxC} và \overline{TxC} cũng là một và được lấy chung từ bộ tạo tốc độ Baud Rate Generator để đơn giản hoá phần giao diện.

+ **SYNDET (SYNC Detect/BRKDET Break Detect):**

Chân này được sử dụng trong chế độ thu phát đồng bộ để nhận biết ký tự đồng bộ, có thể sử dụng như đầu vào hoặc đầu ra, được định nghĩa qua từ điều khiển. Chân được chuyển đầu ra sau khi hệ thống có Reset. Trong chế độ đồng bộ nội (Internal Sync Mode), chân này lên mức “1” được sử dụng như đầu ra trạng thái báo đã định vị được ký tự đồng bộ trong chế độ thu. Khi được lập trình ở chế độ xung đồng bộ kép (Double Sync Character), hay còn gọi là bi-sync, SYNDET sẽ lên mức “1” ở giữa bit cuối của ký tự đồng bộ thứ hai. SYNDET được Reset trong khi thực hiện đọc trạng thái. Ở chế độ đồng bộ ngoại (External Sync Detect Mode), sườn xung lên tại chân SYNDET khởi động 8251 bắt đầu ghép dữ liệu ký tự từ sườn lên của xung nhịp \overline{RxC} . Chế độ này bị cấm khi lập trình cho 8251 hoạt động ở chế độ Internal Sync Mode.

+ **BREAK (chỉ có trong chế độ không đồng bộ (Asynchronous Mode):**

Đầu ra này sẽ lên “1” khi trên lối vào là LOW (=”0”) xuyên suốt hai lần gặp bit Stop trong chuỗi (tất nhiên kể cả bit Start, các bits dữ liệu và bit chặn lẻ). Bit BREAK cũng có thể đọc được như một bit trạng thái.

b) *Mô tả hoạt động*

Việc xác định chế độ làm việc cho USART-8251 được thực hiện thông qua chương trình mềm. Một chuỗi các từ điều khiển cần được CPU gửi tới 8251 để xác định các định dạng truyền tin. Các từ điều khiển sẽ xác lập: Baud Rate, độ dài mã ký tự, số bit Stop, đồng bộ hay dị bộ, kiểm tra chẵn lẻ v.v... Trong chế độ đồng bộ, còn cần xác định Internal Sync

hay External Sync Mode. Sau khi đã nhận được các từ điều khiển cần thiết, 8251 sẵn sàng làm việc. Tất nhiên, sau khi nhận các từ điều khiển, 8251 còn phải chờ cho đến khi bit TxEnable được thiết lập nhờ từ lệnh làm việc (Command Instruction Word) và tín hiệu \overline{CTS} (Clear To Send).

b) Lập trình cho 8251:

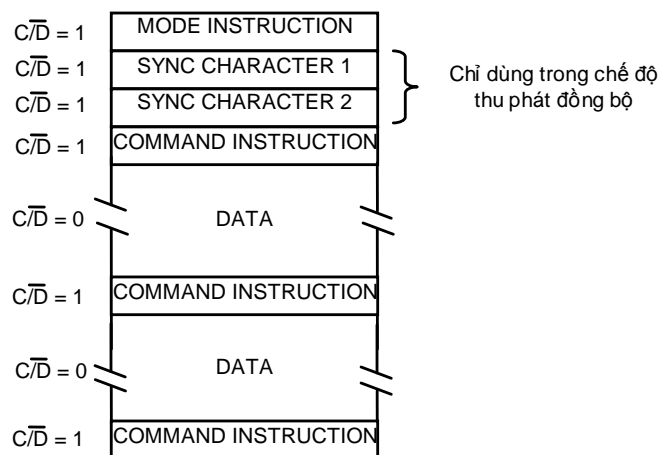
Trước khi phát hay nhận dữ liệu, 8251 phải được nhận một chuỗi từ điều khiển. Từ điều khiển 8251 có hai loại: Lệnh chế độ (Mode Instruction) và Lệnh làm việc (Command Instruction).

– Mode Instruction (MI):

Từ điều khiển chế độ làm việc cho 8251, được nạp vào sau khi mạch được khởi động hay tái khởi động cứng hoặc mềm (Reset). Khi đã được CPU ghi vào 8251, các ký tự SYNC hoặc từ lệnh làm việc (Command Instruction) có thể được chuyển tiếp cho 8251 để kích hoạt 8251.

– Mode Instruction (CI):

Từ lệnh làm việc cho 8251, được dùng để điều khiển công việc thực thụ của mạch. Từ điều khiển (MI) và từ lệnh (CI) phải được gửi cho 8251 theo một tuần tự khe khắt (Xem Hình IV.6). MI phải được ghi vào 8251 ngay sau khi có tín hiệu Reset trước khi sử dụng cho việc truyền dữ liệu.

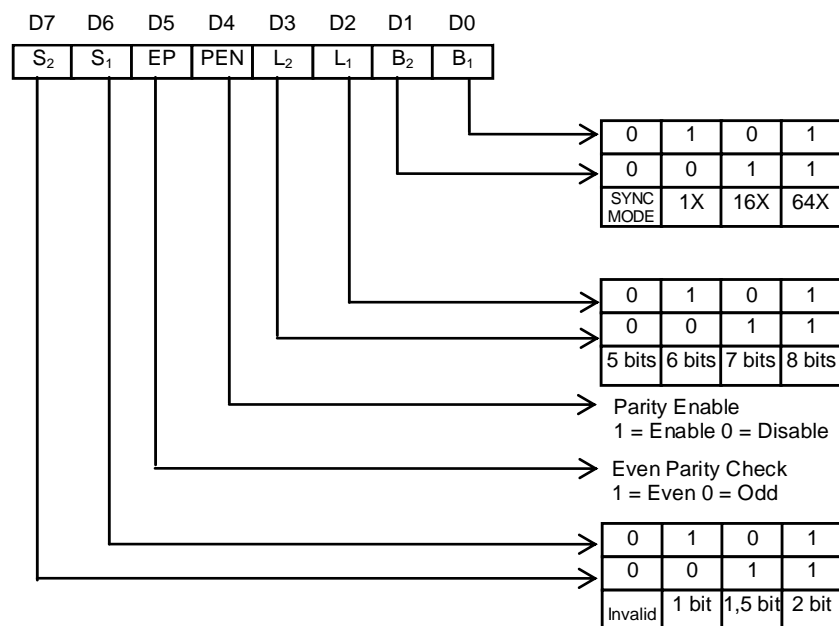


Hình IV.7 Tuần tự khởi động và làm việc với 8251.
Ký tự SYNC thứ hai có thể bỏ qua khi MI xác định cho 8251 làm việc ở chế độ tự đồng bộ (Internal SYNC Mode)

8251 có thể làm việc trong hai chế độ: Dị bộ (Asynchronous) và đồng bộ (Synchronous). Để hiểu được các chế độ làm việc này, có thể giả

thiết rằng 8251 là hai vi mạch độc lập: một mạch thu phát đồng bộ và một mạch thu phát dị bộ cùng được ghép vào trong một vỏ. Các chế độ chỉ có thể đặt cho 8251 sau tín hiệu Reset hệ thống. Lưu ý rằng khi cho phép sử dụng bit kiểm tra chẵn lẻ (parity bit), nó sẽ không được tính vào trong độ dài của ký tự thu phát. Giá trị thực của bit chẵn lẻ trên lối vào RxD không được đọc vào 8251. Nếu độ dài ký tự nhỏ hơn 8 bits, nó sẽ bị bỏ đi khi đọc vào 8251, còn nếu độ dài ký tự bé hơn, sẽ được coi là mức “0”.

Cấu trúc từ điều khiển chế độ không đồng bộ (Asynchronous Mode) được thể hiện trên Hình IV.7. Từ điều khiển được gửi cho 8251 ngay sau khi có tín hiệu Reset. Dạng từ điều khiển của USART-8251 được biểu diễn trên Hình IV. 7

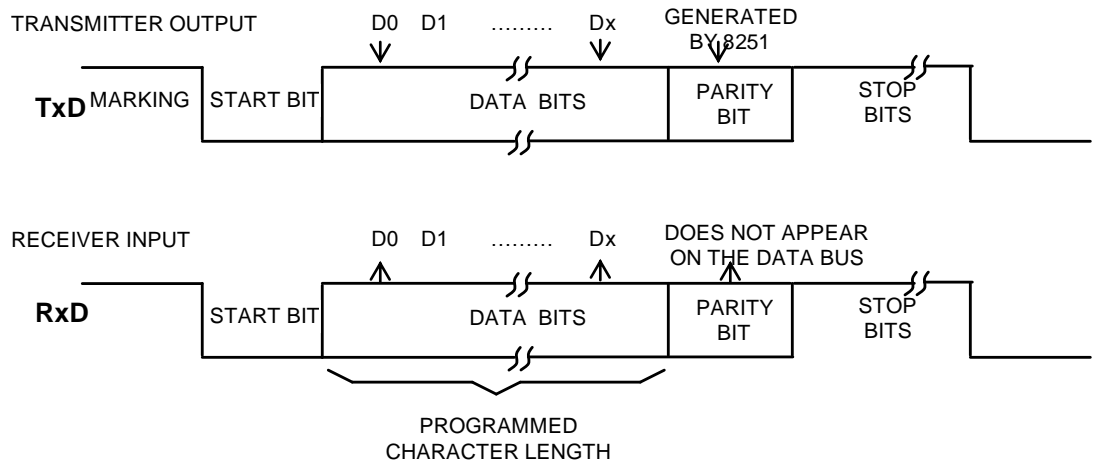


Hình IV.8 Khuôn dạng từ điều khiển cho 8251, Chế độ không đồng bộ (Asynchronous Mode)

(Only affects Tx, Rx never requires more than one stop bit)

+ **Phát không đồng bộ (Asynchronous Transmission):**

Khi dữ liệu được chuyển cho 8251 phát đi, 8251 tự động ghép thêm bit Start (mức “0”), sau đó là các bit dữ liệu, bắt đầu bằng LSBit và gán thêm bit kiểm tra chẵn lẻ, rồi đến bit Stop theo đúng *khung dữ liệu* được định nghĩa trong từ điều khiển. khung dữ liệu này được truyền đi như một chuỗi xung trên lối ra TxD. Các bit được dịch chuyển ra TxD bằng sườn xuống của xung nhịp $\overline{\text{TxC}}$ theo tốc độ 1, 16 hay 64 lần xung nhịp $\overline{\text{TxC}}$ theo từ điều khiển đã xác định sẵn. Khi không còn dữ liệu để truyền đi, TxD chuyển lên mức cao “1” (marking).

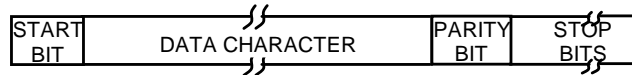


TRANSMISSION FORMAT

CPU BYTE (5 – 8 BITS/CHAR)



ASSEMBLED SERIAL DATA OUTPUT (TxD)



RECEIVE FORMAT

SERIAL DATA INPUT (RxD)



CPU BYTE (5 – 8 BITS/CHAR)



Lưu ý: Nếu độ dài ký tự được gán theo lệnh là 5, 6 hoặc 7 bits
Các bits không dùng tới sẽ được gán bằng "0"

+ *Thu không đồng bộ (Asynchronous Receive):*

Bình thường, RxD ở mức cao ("1"), sườn xuống xuất hiện được coi là sự bắt đầu của bit Start. Tính hợp lệ của bit Start được xác nhận bằng xung mẫu tại điểm giữa của xung này (đối với trường hợp tốc độ 16x hay 64x). Nếu vẫn có giá trị là "0", đó là bit Start hợp lệ, và bộ đếm bit bắt đầu hoạt động. Các bit của dữ liệu và bit chẵn lẻ được lấy mẫu tại điểm giữa trên lối vào RxD bằng sườn lên của xung nhịp \overline{RxC} . Nếu mức thấp được nhận biết ở đoạn tồn tại của bit Stop, cờ lỗi sẽ được thiết lập. Bit Stop báo hiệu kết thúc của một ký tự. Lưu ý rằng phần thu chỉ cần nhận được một bit Stop, bất kể số lượng bit Stop được gán là bao nhiêu. Ký tự nhận được sẽ được chuyển vào bộ đệm song song của 8251. Tín hiệu RxRDY sẽ chuyển lên mức cao để báo cho CPU biết có thể nhận ký tự.

Nếu ký tự trước chưa được CPU đọc về, ký tự mới vẫn sẽ được chuyển vào thanh ghi đệm này, và cờ báo lỗi tràn sẽ được thiết lập (tức là ký tự trước bị bỏ qua). Tất cả các cờ báo lỗi có thể Reset nhờ lệnh Error Reset. Hình IV.4 cho thấy các dạng thức *khung dữ liệu* (Data Fram) được phát đi và thu về trong chế độ làm việc không đồng bộ.

+ *Phát đồng bộ (Synchronous Transmission):*

Đầu ra TxD ở mức cao cho đến khi CPU chuyển từ đầu tiên đến 8251, thông thường đó là ký tự đồng bộ SYNC. Khi đầu \overline{CTS} chuyển sang mức thấp, ký tự đầu tiên được phát nối tiếp ra TxD. Tất cả các ký tự được dịch chuyển nối tiếp ra theo sườn xuống của \overline{TxC} . Tốc độ phát dữ liệu bằng đúng tốc độ \overline{TxC} . Khi hoạt động phát đã được khởi động, chuỗi dữ liệu trên TxD được đồng bộ theo \overline{TxC} . Nếu CPU không cung cấp dữ liệu cho 8251 trước khi bộ đệm phát bị rỗng, thì (các) ký tự SYNC sẽ được chèn vào chuỗi ký tự phát đi đồng thời tín hiệu điện áp chân TxEMPTY sẽ chuyển đổi lên mức “1” để thông báo rằng bộ đệm phát rỗng và ký tự SYNC được phát. Điện áp trên chân TxEMPTY được Reset khi ký tự mới được CPU chuyển tới 8251.

+ *Thu đồng bộ (Synchronous Receive):*

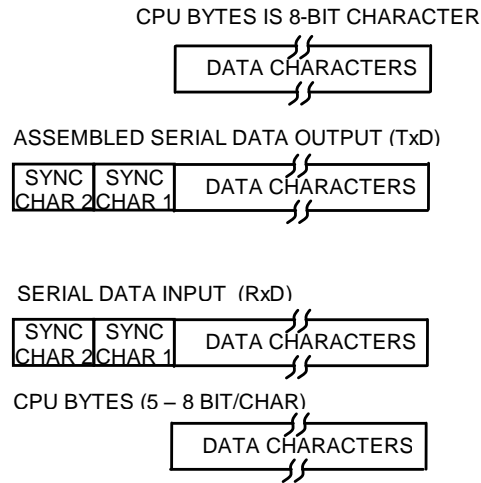
Tín hiệu đồng bộ có thể là tự động do bản thân 8251 tạo ra hoặc thu từ bên ngoài. Khi hoạt động ở chế độ đồng bộ, lệnh “săn tìm” (bit EH - ENTER HUNT) được gộp trong từ lệnh cho 8251. Dữ liệu trên lối vào RxD được “lấy mẫu” qua sườn lên của xung nhịp \overline{RxC} . Nội dung của thanh ghi đệm nhận Rx buFFer được so sánh với ký tự đồng bộ SYNC cho đến khi hoàn toàn phù hợp. Nếu được chọn là chế độ đồng bộ kép, tập các ký tự cũng được so sánh tương tự. Khi cả hai ký tự đồng bộ đã được nhận biết, USART 8251 kết thúc chế độ săn tìm và chuyển sang đồng bộ hoá ký tự. Chân SYNDET chuyển sang trạng thái logic “1”, và sẽ tự động Reset nhờ lệnh đọc trạng thái.

Trong chế độ External Sync, việc đồng bộ đạt được nhờ áp mức cao lên chân SYNDET để loại trừ chế độ “săn tìm” của 8251. Mức cao này sẽ được Reset sau một nhịp \overline{RxC} . Lệnh EH không có tác động gì trong chế độ này. Việc phát hiện lỗi chẵn lẻ và lỗi tràn hoàn toàn tương tự như ở thu dữ liệu.

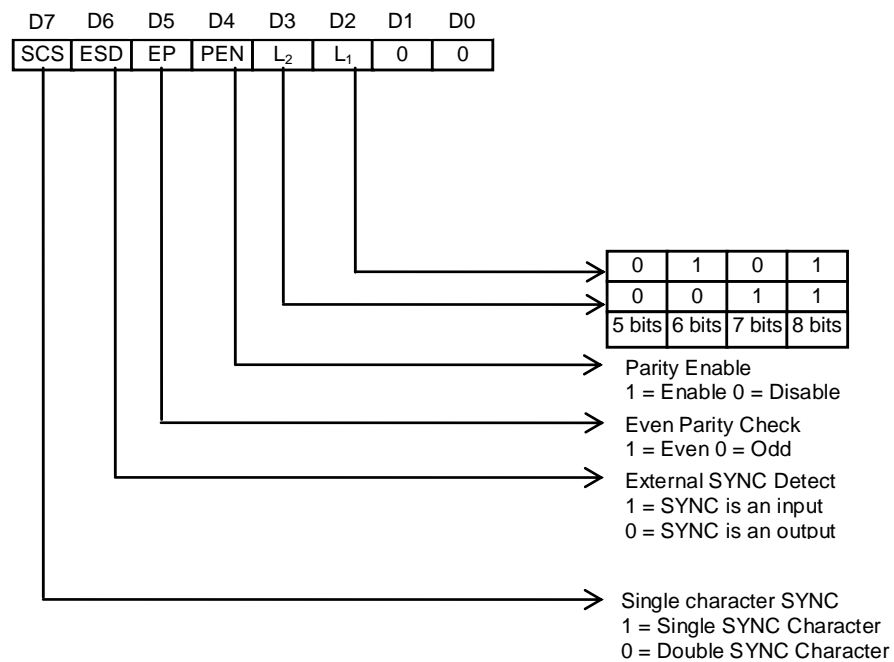
Hình IV.8 là gói dữ liệu trong thu phát đồng bộ.

– Command Instruction (CI)

Dạng của từ lệnh được thể hiện trên Hình IV.11. Sau khi 8251 đã được thiết lập chế độ làm việc, và ký tự đồng bộ đã được tách (nếu là chế độ đồng bộ) nó đã sẵn sàng cho một hoạt động thu phát dữ liệu. Từ lệnh được sử dụng để điều khiển các hoạt động thực tế của 8251 trong chế độ đã đặt. Các hoạt động đó là: Cho phép Thu / cho phép Phát, Reset các lỗi hay điều khiển Modem.

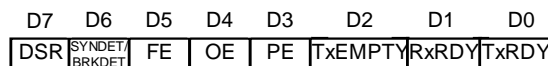


Hình IV. 8 Gói dữ liệu trong thu nhất đồng bộ



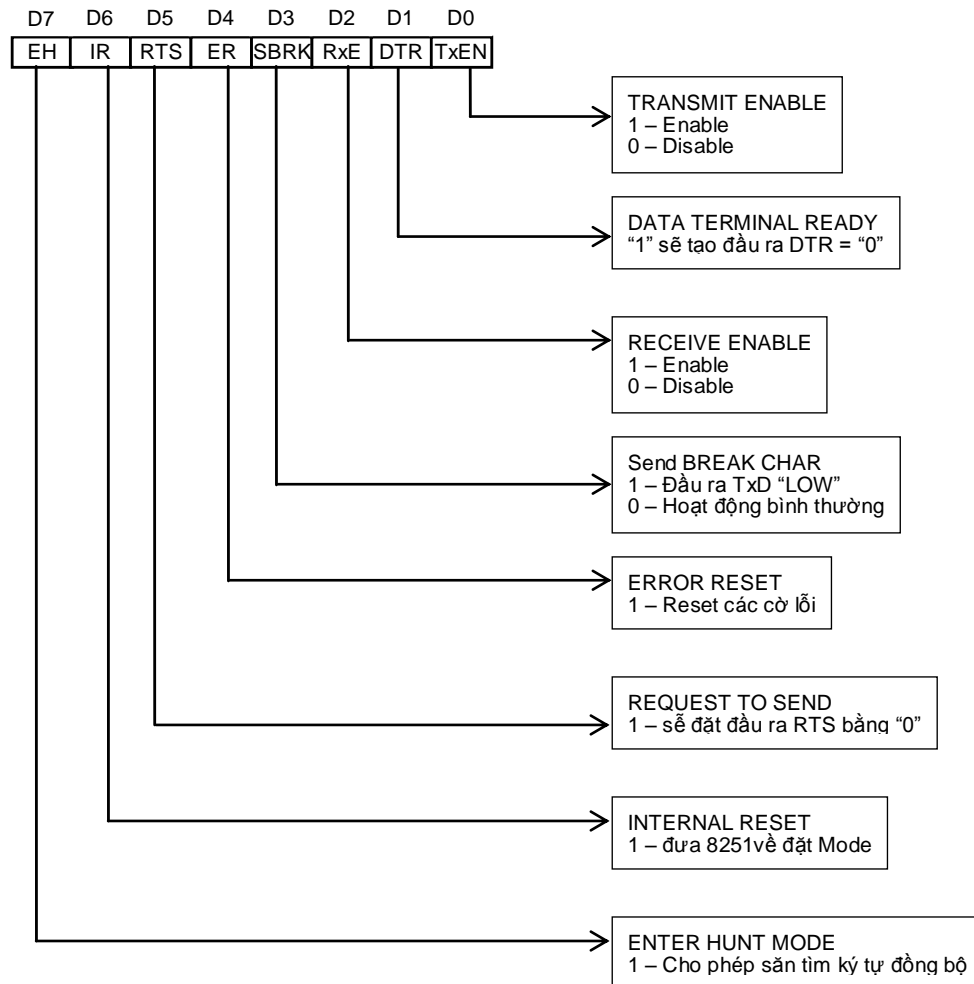
Hình IV. 9 Khuôn dạng từ điều khiển cho 8251, Chế độ đồng bộ (Synchronous Mode)

Cũng cần hiểu thêm về từ trạng thái của 8251. Với lệnh đọc Input vào CPU trong trường hợp C/D = “1”, từ trạng thái được đọc vào. Nội dung từ trạng thái được thể hiện trên Byte đọc vào, các giá trị logic của các bit trạng thái RxRDY, TxEMPTY, SYNDET/BRKDET tại các bit tương ứng D1, D2 và D6 là giá trị trên chính các chân ra tương ứng của 8251. Riêng bit TxRDY (bit D0) thể hiện trạng thái rỗng của thanh ghi đệm dữ liệu vào, còn giá trị trên chân ra TxRDY của 8251 là kết quả của sự phối hợp cùng các giá trị của CTS và TxEN trước đó.



Hình IV.10 Byte trạng thái của 8251

Bit D3 (Parity Error) bằng “1” nghĩa là phát hiện có lỗi trong khi kiểm tra tính chẵn lẻ của byte dữ liệu, bit này được xoá bằng từ lệnh (bit ER).
 Bit D4 (Overrun Error) được Set nếu CPU không kịp đọc dữ liệu trước khi có một byte mới đang được thu về. còn bit D7 (Data Set Ready) thông báo DSR đang ở mức “0”.



Hình IV.11- Khuôn dạng của từ lệnh 8251

Ví dụ:

1. Thiết kế thiết bị đầu cuối bằng cách ghép nối CPU 8085 với các mạch PPI 8255 và mạch USART 8251 để thu thập dữ liệu vào song song từ 6 cổng và khả năng trao đổi thông tin với một máy tính chủ qua đường truyền dẫn nối tiếp.

Lời giải:

Vi mạch PPI 8255 là mạch khả trình cho phép mở rộng số lượng cổng vào ra song song, mỗi mạch có 24 lối vào/ra nhóm thành 3 cổng Port A, Port B và Port C, như vậy, cần 2 vi mạch PPI 8255 để tạo ra 6 cổng thu thập dữ liệu song song, mỗi cổng bao gồm 8 bit tương ứng từ $D_7 \div D_0$.

Vi mạch USART 8251 là mạch thu phát dữ liệu nối tiếp vạn năng, có thể sử dụng để trao đổi dữ liệu với máy tính qua cổng COM.

CPU 8085 sử dụng byte địa chỉ thấp từ $A_7 \div A_0$ cùng các lệnh IN port và OUT port để trao đổi dữ liệu với các thiết bị ngoại vi, do vậy, ta cần giải mã địa chỉ cho các chip PPI 8255 và chip USART 8251, kết hợp các tín hiệu RD và WR để thực hiện yêu cầu đề bài.

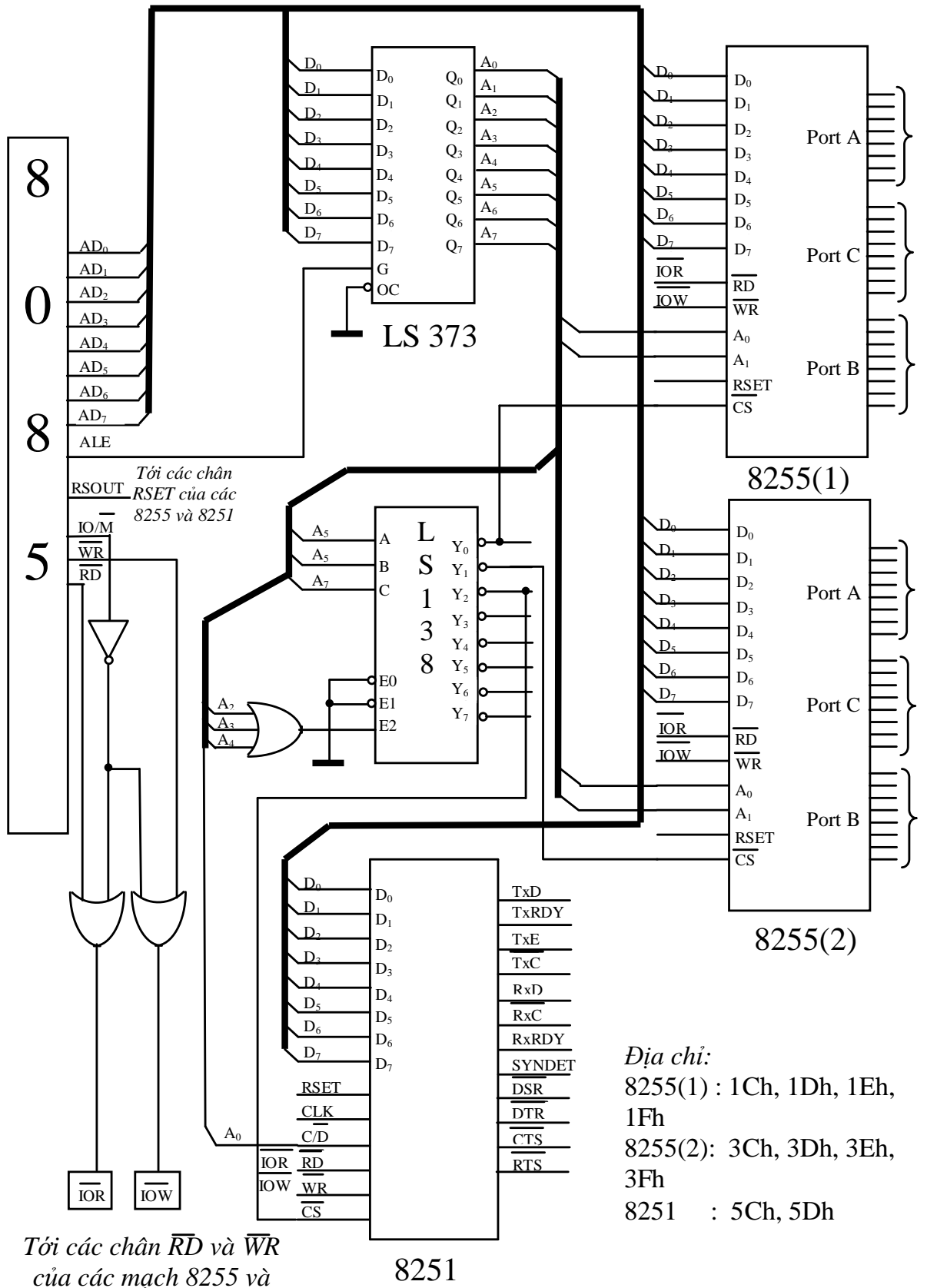
Một phương án tổ chức địa chỉ là:

8255(1) : Port A 1Ch, Port B 1Dh, Port C 1Eh, Thanh ghi từ điều khiển 1Fh

8255(2) : Port A 3Ch, Port B 3Dh, Port C 3Eh, Thanh ghi từ điều khiển 3Fh

8251: Thanh ghi từ điều khiển và ghi lệnh 5Ch, Cổng dữ liệu 5Dh

Sử dụng mạch giải mã địa chỉ 74LS138, ta có sơ đồ nguyên lý thiết bị đầu cuối thu thập dữ liệu song song từ 6 cổng, trao đổi với máy tính chủ qua cổng nối tiếp như hình vẽ .



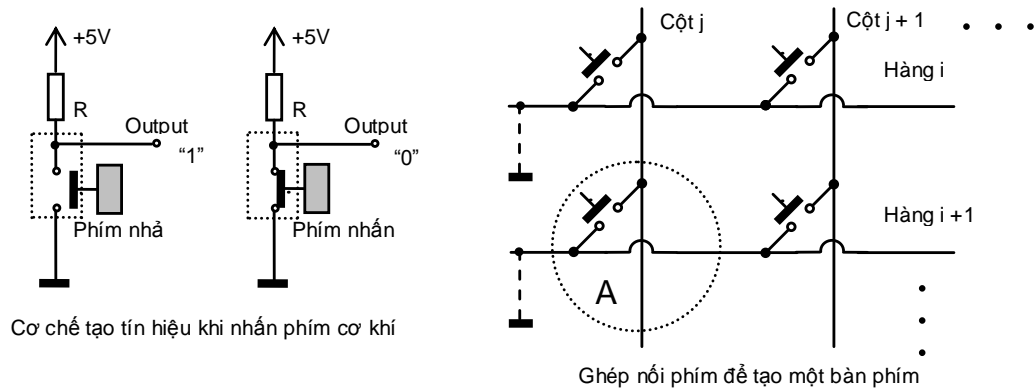
CÂU HỎI ÔN TẬP CHƯƠNG IV.

1. Khái niệm chip khả lập trình và những tiện lợi trong ứng dụng
2. Nêu ví dụ một vài chip khả lập trình tiêu biểu trong thu phát thông tin dữ liệu song song và nối tiếp, khả năng ứng dụng của các chip đó.
3. Cấu hình và chức năng các khối cơ bản của PPI 8255.
4. Cách tính địa chỉ các cổng PA, PB, PC và địa chỉ thanh ghi từ điều khiển CWR cho vi mạch PPI 8255.
5. Phân tích các chế độ và cách thiết lập từ điều khiển để PPI 8255 làm việc trong các chế độ sau:
 - a. Chế độ vào/ra cơ bản
 - b. Chế độ vào/ra có xung chốt dữ liệu
 - c. Chế độ vào/ra hai chiều có xung chốt dữ liệu.
 - d. Chế độ thiết lập và xóa các bit trên cổng C.
6. Phân tích và thiết lập từ điều khiển để PPI 8255 làm việc theo cấu hình: Port A, nửa cao port C: OUT; port B, nửa thấp port C: IN.
7. Sơ đồ khối các thành phần chức năng và chức năng của vi mạch khả lập trình điều khiển ngắt cứng PIC 8259
8. Trình bày cơ chế ngắt cứng và hoạt động của PIC 8259 khi xuất hiện tín hiệu yêu cầu ngắt cứng IRQ_i
9. Lưu đồ chương trình thiết lập PIC 8259
10. Khái niệm số ngắt nền trong từ điều khiển PIC 8259.
11. Trình bày các phương thức xét ưu tiên ngắt của PIC 8259.
12. Sơ đồ khối cấu trúc vi mạch định thời PIT 8253/54. Sự khác nhau giữa PIT 8253 với PIT 8254.
13. Phân tích nội dung từ điều khiển PIT 8253/54
14. Trình bày sơ lược các chế độ (modes) làm việc của PIT 8253/54
15. Đề xuất phương thức sử dụng PIT 8253/54 tạo ngắt cứng theo chu kỳ thời gian cố định đối với CPU.
16. So sánh ưu nhược điểm của hai phương pháp phát/thu dữ liệu đồng bộ và không đồng bộ.
17. Sơ đồ khối cấu trúc và chức năng của vi mạch USART 8251
18. Trình bày các từ điều khiển chế độ làm việc của USART 8251.

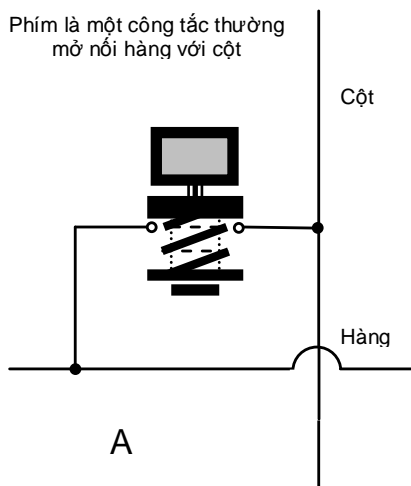
19. Trình bày và giải thích cấu trúc gói dữ liệu và khung dữ liệu trong các chế độ làm việc của USART 8251.
20. Thiết kế thiết bị đầu cuối bằng cách ghép nối CPU 8085 các mạch PPI 8255 và mạch USART 8251 để thu thập dữ liệu song song từ 6 cổng vào/ra song song, trao đổi thông tin với một máy tính chủ qua đường truyền dẫn nối tiếp.
21. Thay đổi CPU 8085 ở bài tập 20 bằng cách sử dụng Single Chip MicroComputer 8051 và các mạch ghép nối ngoại vi thích hợp.
22. Trình bày sự khác nhau giữa vào ra nối tiếp và vào ra song song.

CHƯƠNG V. THIẾT BỊ VÀO RA CỦA HỆ VI XỬ LÝ

V.1 Bàn phím Hex Keyboard



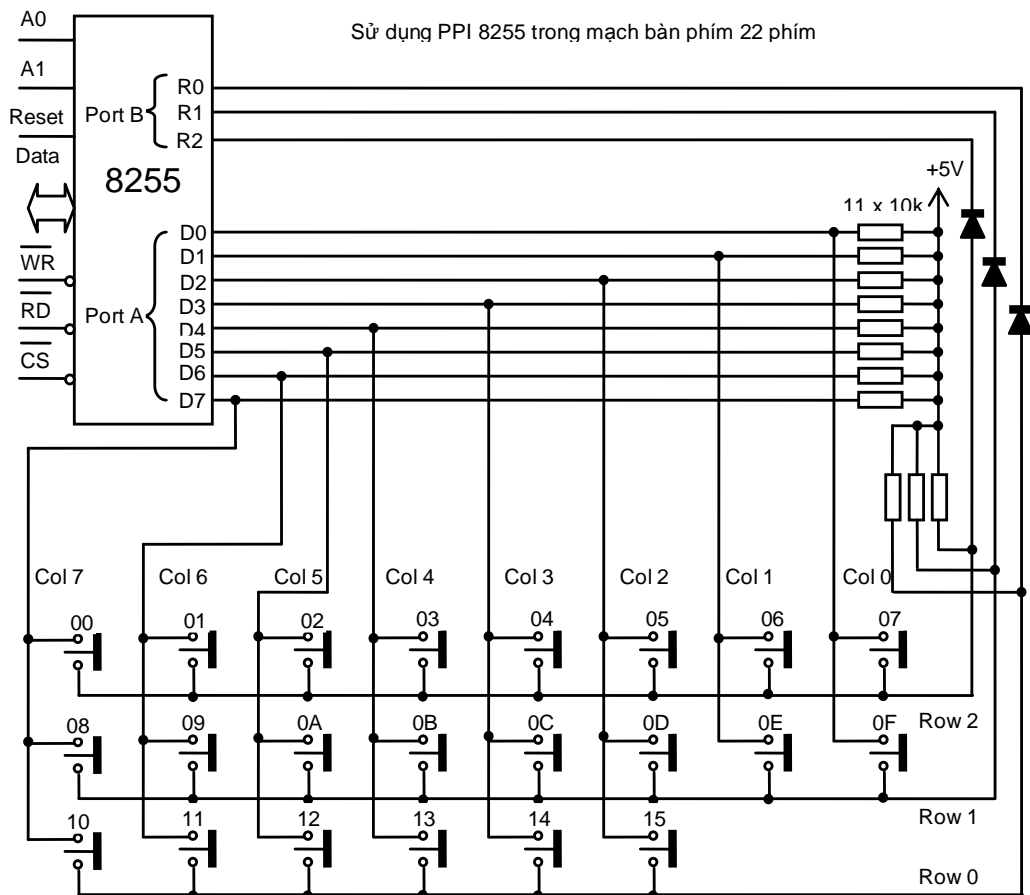
Hình V.1. Phím tiếp xúc và cách tạo bàn phím



Bàn phím được tổ chức theo kiểu ma trận các hàng và các cột, tại vị trí giao nhau không tiếp xúc được ghép một công tắc thường mở nối hàng với cột, chỉ tiếp xúc khi được nhấn. Để xác định có một phím bị nhấn, ta nối đất tất cả các hàng và đọc nội dung các cột. Nếu trên cột nào đó ta đọc được giá trị là “0”, tương ứng với trường hợp có một phím trên cột đó bị nhấn. Dễ dàng thấy rằng, nếu các hàng i và $i + 1$ nối đất, bất cứ phím nào trên cột j (hay $j + 1$) bị nhấn, ta đều đọc được giá trị “0” trên cột j (hay $j + 1$).

Hình V.2 là một bàn phím Hexa gồm 22 phím được tạo từ một ma trận 3 hàng và 8 cột. Giả sử rằng ta dùng vi mạch vào ra song song PPI-8255 để xây dựng nên bàn phím như trên Hình V.2. Ba lối ra của port B gồm R0, R1, R2 (tương ứng với các dây PB0, PB1 và PB2) được dùng ở chế độ Output, 8 lối vào của port A dùng D0 ÷ D7 (tương ứng với các dây PA0 ÷ PA7) ở chế độ Input. Như vậy chu trình đọc phím theo chế độ dò tìm (polling) được thực hiện như sau:

1. Để đảm bảo phím nhấn trước đó đã được nhả ra, các giá trị “0” cùng lúc được áp lên tất cả các hàng và đọc các giá trị trên các cột. Nếu các cột đều ở mức “1”, chương trình tiếp tục đọc giá trị các cột
2. Quét các cột, tức là đọc giá trị tại các cột để phát hiện có phím bị nhấn. Để tăng độ tin cậy khi đọc phím, tránh tác động của nhiễu cơ học khi phím bị nhấn và các loại nhiễu khác, sau khi phát hiện có phím bị nhấn, chương trình chờ khoảng 20msec rồi đọc tiếp giá trị tại các cột. Giá trị “0” đọc được ở cột nào sẽ được ghi nhớ để sử dụng cho việc xác định phím ở vị trí nào bị nhấn
3. Quét hàng để xác định vị trí của phím bị nhấn. Số vòng lặp này là không cố định, nhưng nhiều nhất là bằng số hàng có trong cấu trúc của bàn phím
4. Gán mã cho phím. Mã cho phím là do thiết kế phần cứng quy định, tùy theo chức năng và yêu cầu của người dùng.



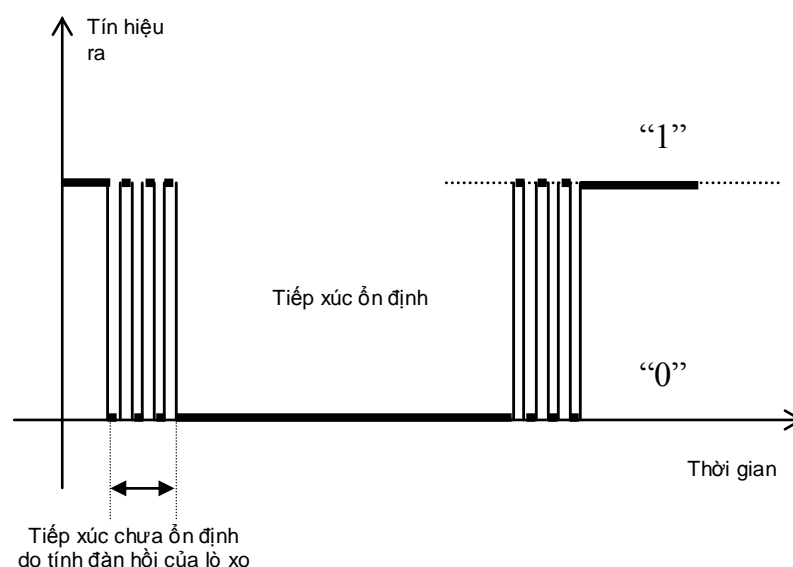
Hình V.2 – Bàn phím 22 phím sử dụng giao tiếp qua PPI8255

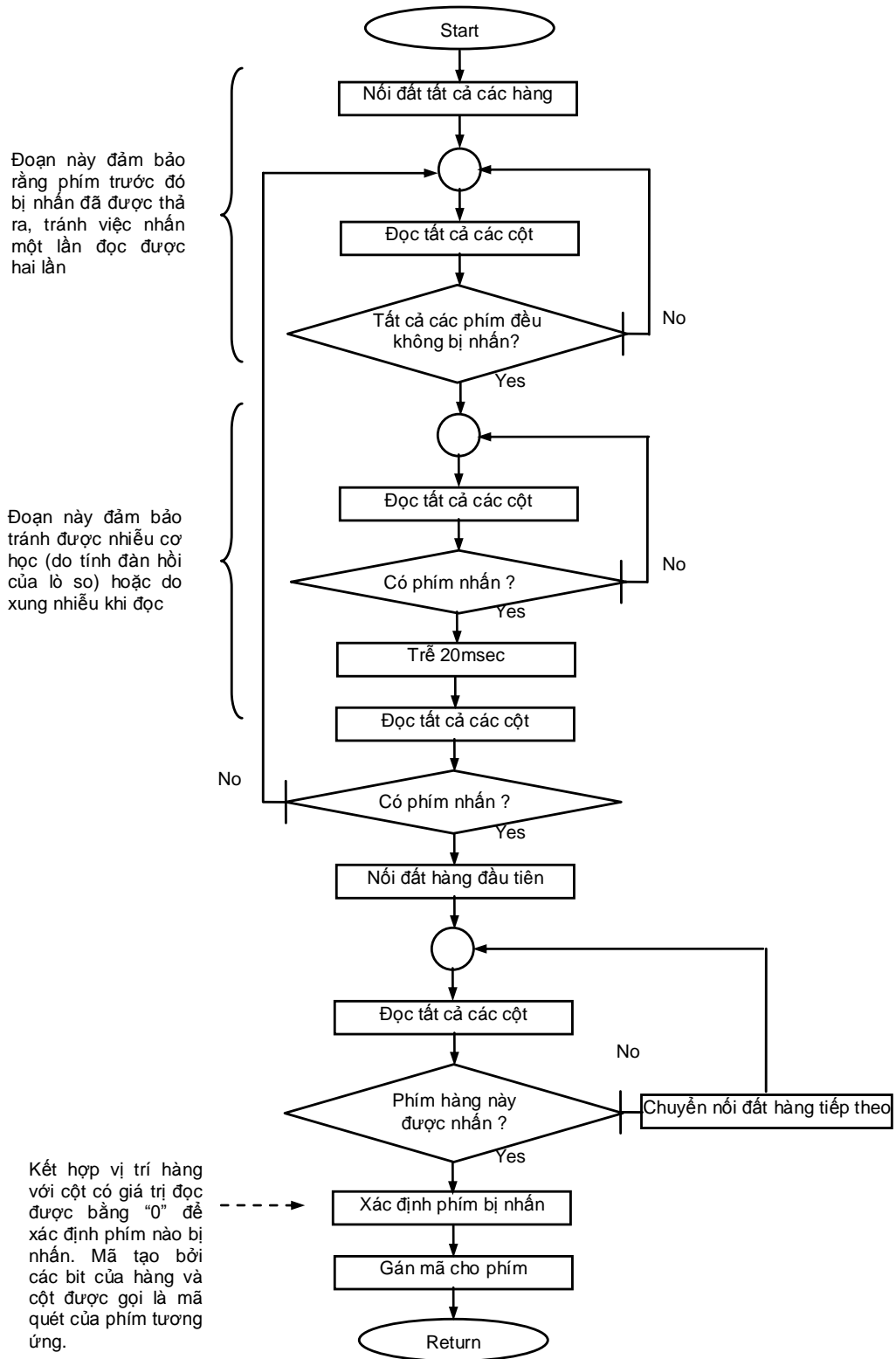
Trong ví dụ này giả sử rằng các phím được gán mã như sau:

- Từ phím 00 đến phím 0F (toàn bộ các phím trong Row 1 và Row 2) được gán mã Hexa từ “0_H” đến “F_H”
- Các phím ở Row 0 có thể gán các chức năng sau”
 - Phím 10 là phím chức năng “GO” - thực hiện chương trình
 - Phím 11 là phím chức năng “INS” - thực hiện chức năng thay đổi nội dung các thanh ghi của CPU
 - Phím 12 là phím “REP” - thực hiện chức năng sửa nội dung thanh ghi của CPU
 - Phím 13 là phím “DISP” - thực hiện chức năng hiển thị nội dung các thanh ghi của CPU
 - Phím 14 là phím “STEP” - thực hiện chức năng chạy chương trình theo từng lệnh
 - Phím 14 là phím “ENTER” - thực hiện chức năng kết thúc nhập dữ liệu hoặc lệnh từ bàn phím

Lưu đồ chương trình đọc và xác định phím bị nhấn được thể hiện trên Hình V.3 Chương trình có thể được viết dưới dạng một chương trình con.

Do tính đàn hồi của lò xo trong phím nên sự tiếp xúc của phím sau khi bị nhấn có thể mô tả như hình sau:





Hình V.3 Lưu đồ chương trình đọc bàn phím

V.2 Ghép nối bàn phím với hệ Vi xử lý

Bàn phím là thiết bị ngoại vi cho phép đưa thông tin vào máy tính dưới dạng mã ký tự. Bàn phím thực hiện chức năng chuyển thông tin dạng lực nhấn phím và vị trí của phím được nhấn thành mã phím và chuyển cho máy tính. Bàn phím gồm hai bộ phận chính là ma trận phím và mạch điện tử quét phím. Ma trận phím là tổ hợp các phím nhấn được sắp xếp theo các hàng và cột.

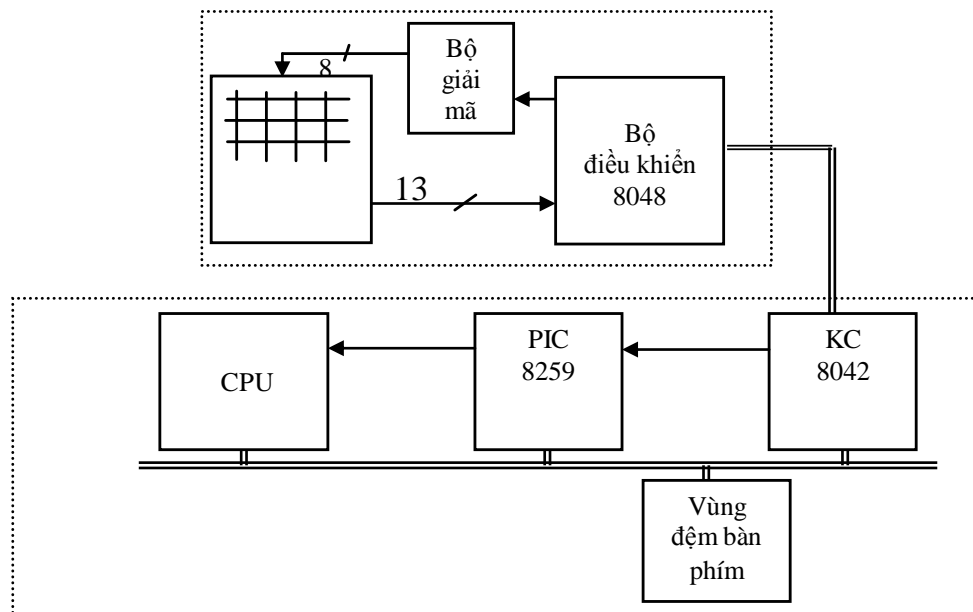
Bình thường phím luôn ở trạng thái nhả, khi phím nhả thì hai tiếp điểm không được nối với nhau, đầu ra có mức điện áp dương tương ứng với mức logic “1”. Khi phím được nhấn thì hai tiếp điểm được nối với nhau qua công tắc phím và đầu ra có mức điện áp bằng 0V tương ứng mức logic “0”.

Để mỗi lần nhấn phím có một mã phím tương ứng được tạo ra, cần sắp xếp hệ thống phím dưới dạng ma trận phím.

Ma trận phím gồm các dây hàng và các dây cột giao nhau nhưng không tiếp xúc với nhau. Các công tắc phím được đặt ở chỗ giao của hàng và cột. Hai tiếp điểm của công tắc nằm ở trên hàng và cột tại chỗ giao nhau đó. Mỗi khi phím được nhấn thì hai dây hàng và cột được nối với nhau qua hai tiếp điểm của công tắc tại chỗ giao nhau.

V.2.1. Hệ thống bàn phím của máy vi tính

Hệ thống bàn phím của máy vi tính gồm hai phần bàn phím và thiết bị giao diện bàn phím, được kết nối và trao đổi thông tin theo kiểu “chủ” “thợ”.



Hình V.4 – Sơ đồ ghép nối bàn phím (keyboard) với hệ thống máy tính

Bàn phím là tổ hợp của ma trận 8x13 phím và mạch vi điều khiển μP8048 . Mạch μC8048 là một hệ vi xử lý nhỏ được tích hợp trên một đơn chip. Mạch 8048 bao gồm CPU, bộ nhớ ROM chứa chương trình điều khiển quét và tạo mã phím, RAM chứa dữ liệu của chương trình điều khiển, hai cổng vào/ra P1 và P2, một cổng dữ liệu 8 bit. Mạch 8048 tuần tự đưa mã nhị phân 3 bit ra tại

cổng P2, qua bộ giải mã 3/8 tạo ra tín hiệu quét bàn phím. Tại thời điểm mã 3 bit được đưa ra, mạch $\mu P8048$ thực hiện đọc tín hiệu 13 bit từ ma trận phím vào cổng P1, từ đây tạo ra mã phím (mã quét) của phím được nhấn. Khi phím được nhả một mã phím (mã quét) cũng được tạo ra bằng cách cộng mã phím nhân với 80H.

Mạch $\mu P8048$, được nuôi bằng nguồn từ máy tính, thực hiện trao đổi thông tin với thiết bị giao diện bàn phím KC 8042 theo kiểu nối tiếp đồng bộ. KC 8042 có cấu trúc tương tự mạch $\mu P8048$. KC 8042 đóng vai trò “chủ”, 8048 đóng vai trò “thợ” trong các quá trình truyền tin thông qua hai dây tín hiệu: dây “DATA” và dây “CLOCK”.

Dây “DATA” truyền tín hiệu dữ liệu nối tiếp giữa $\mu P8048$ và KC 8042. Tín hiệu nối tiếp bao gồm: bit START, 8 bit dữ liệu, 1 bit PARITY, 1 bit STOP. Quá trình trao đổi thông tin giữa $\mu P8048$ và KC 8042 được đồng bộ bởi tín hiệu trên dây “CLOCK”.

V.2.2. Quá trình truyền dữ liệu từ bàn phím cho CPU

Mạch $\mu P8048$ luôn phải kiểm tra trạng thái truyền tin qua hai dây “DATA” và “CLOCK” trước khi phát đi mã phím. Khi KC 8042 đặt “DATA” = 0 và “CLOCK” = 1 thì 8048 phải nhận các chỉ lệnh từ KC 8042. Khi KC 8042 đặt “DATA” = 1 và “CLOCK” = 1 thì $\mu P8048$ được quyền truyền mã phím cho máy tính. Quá trình truyền dữ liệu được đồng bộ bằng dây xung đồng bộ do $\mu P8048$ phát ra trên dây “CLOCK”.

Khi KC 8042 nhận được mã phím dạng nối tiếp, nó loại bỏ các bit tạo khung dữ liệu truyền, chuyển mã phím vào thanh ghi tạm và phát ra yêu cầu ngắt IRQ1 cho hệ thống ngắt cứng. Hệ thống ngắt cứng sẽ kích hoạt chương trình phục vụ bàn phím 09H (chương trình phục vụ ngắt 09H) nằm ở BIOS. Chương trình phục vụ bàn phím 09H có chức năng dịch mã phím thành mã hai byte và chứa vào vùng đệm bàn phím.

Chương trình phục vụ bàn phím 09H trước hết kiểm tra (mã) các phím trượt (Shift, Alt, Ctrl) và các phím đặc biệt (ScrollLock, NumLock, CapsLock, Insert) trước khi dịch mã phím sang mã hai byte.

Mã hai byte được chương trình phục vụ bàn phím 09H tạo ra có cấu trúc tùy thuộc mã phím hoặc tổ hợp mã phím nhận được. Nếu nhận được mã của phím ký tự thì byte thấp của mã hai byte chứa mã ASCII của ký tự tương ứng, byte cao chứa mã phím (mã quét phím). Khi chương trình phục vụ bàn phím 09H nhận được mã các phím không phải là ký tự thì byte thấp của mã hai byte có giá trị 0, byte cao chứa mã phím mở rộng.

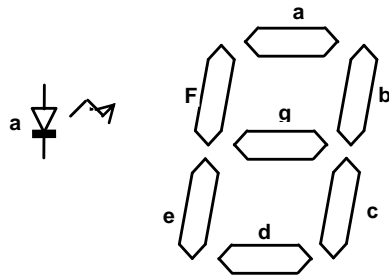
Vùng đệm bàn phím có kích thước 32 byte nằm trên bộ nhớ chính tại địa chỉ 0000H:041EH. Trạng thái của các phím trượt và các phím đặc biệt được chứa ở hai ô nhớ 0000H:0417H và 0000H:0418H. Có thể truy nhập vùng đệm bàn phím để đọc thông tin về bàn phím nhờ chương trình ngắt 16H của BIOS.

Chương trình phục vụ bàn phím 09H cũng xử lý các trường hợp đặc biệt như:

- Khi phím được nhấn quá lâu (ví dụ quá 0.5 giây) và KC 8042 không nhận được mã phím nữa, nó sẽ gửi ra cho đơn vị xử lý trung tâm mã của phím được nhấn.
- Khi nhận được tổ hợp các phím Ctrl+Alt+Del nó sẽ khởi động lại máy tính.
- Khi nhận được mã phím PrintScreen nó sẽ kích hoạt ngắt 05H của BIOS.
- Khi nhận được mã phím Ctrl+Break nó sẽ kích hoạt ngắt 1BH của BIOS.

V.3 Mạch điều khiển và lập trình chỉ thị 7-segments

Hiển thị 7 thanh (7-segment Light Emitting Diode – LED Display) là loại đơn giản nhất nhận tín hiệu ra và hiển thị dưới dạng phát sáng. Có thể sử dụng vì mạch này để hiển thị các ký tự số từ 0 đến 9. Khi có dòng điện chạy qua, diode sẽ phát sáng.

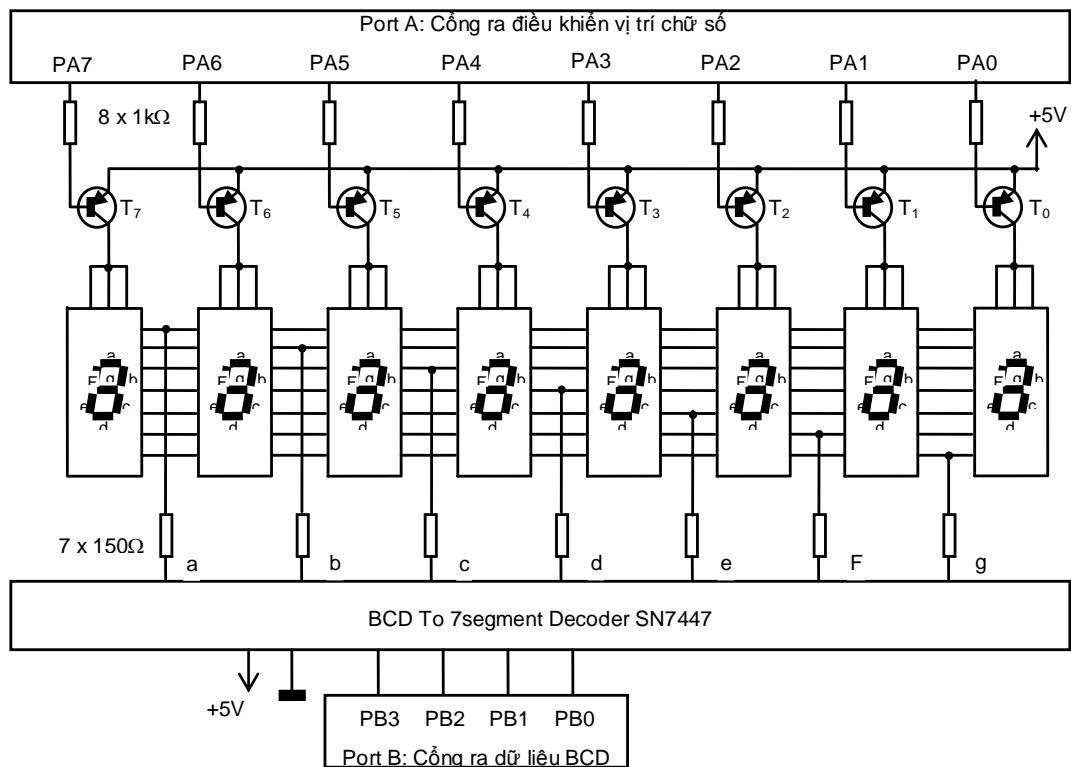


	a	b	c	d	e	F	g
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	0	0	1	1
A	1	1	1	0	1	1	1
B	0	0	1	1	1	1	1
C	1	0	0	1	1	1	0
D	0	1	1	1	1	0	1
E	1	0	0	1	1	1	1
F	1	0	0	0	1	1	1

Hình V.5 là sơ đồ mạch hiển thị 8 digits sử dụng các vi mạch hiển thị 7 segment sử dụng 2 cổng của PPI-8255 theo phương pháp điều khiển hiển thị đa công (Multiplexing) đồng bộ. Các thanh sáng a, b, c, ..., g của các mạch hiển thị 7 thanh được nối song song với nhau và nối với đầu ra của giải mã BCD-7segment SN7447. Việc cấp nguồn nuôi cho mạch hiển thị (1 digit) được đóng ngắt bởi một transistor PNP làm việc ở chế độ khoá đóng mở nhờ xung điều khiển từ một lối ra của cổng A của PPI-8255. Như vậy, tại một thời điểm, bằng cách lập trình cho PPI-8255, ta sẽ điều khiển để duy nhất một mạch hiển thị phát sáng. Nếu tần số của quá

trình phát sáng đạt đến khoảng 15 đến 20 lần/sec, không xảy ra hiện tượng nhấp nháy khi theo dõi.

Dữ liệu cần hiển thị ở dạng mã BCD (4-bit) được đưa ra mạch giả mã hiển thị 7 thanh SN7447 qua 4 dây tương ứng của cổng B, đồng thời vị trí của digit cần hiển thị sẽ được điều khiển phát sáng bằng cách đưa điện áp mức “0” lên lối ra tương ứng trên cổng A để làm thông Transistor cấp nguồn cho mạch 7 segment tương ứng. Như vậy bằng cách lập trình “quét” lần lượt vòng qua tất cả các digit, có thể điều khiển hiển thị một dữ liệu gồm tối đa 8 chữ số.

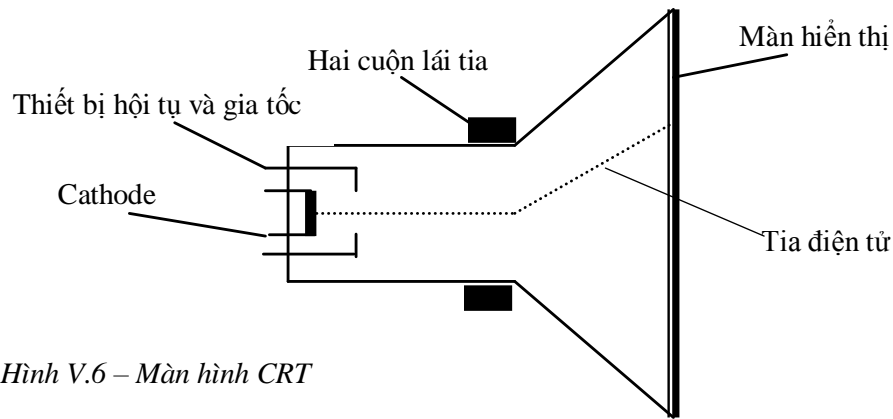


Hình V.5 – Sơ đồ nguyên lý mạch điều khiển bảng hiển thị 8 ký tự số sử dụng PPI 8255 theo phương pháp quét động

V.4 Màn hình (Monitor)

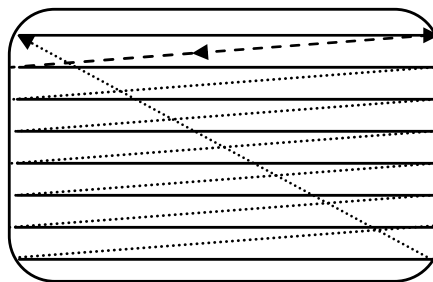
V.4.1 Màn hình ống tia âm cực CRT (Cathode Ray Tube)

Màn hình ống tia âm cực CRT là thiết bị hiển thị thông dụng nhất hiện nay. Màn hình CRT có cấu tạo như sau :



Hình V.6 – Màn hình CRT

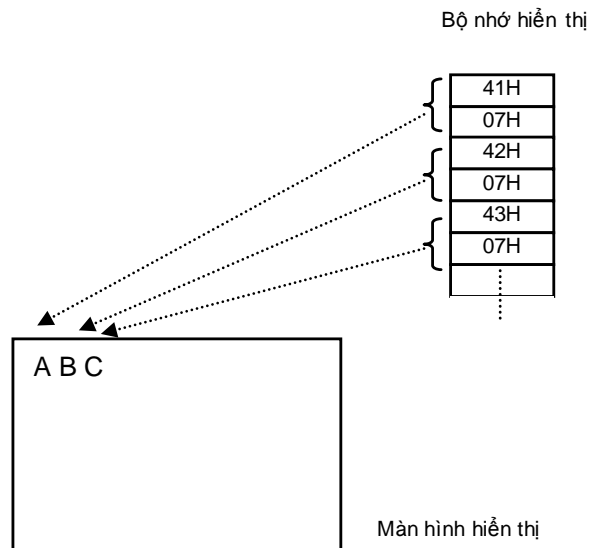
Màn hình CRT là một ống thủy tinh chân không với các bộ phận: cathode phát xạ điện tử, ống phóng tia điện tử, cuộn lái tia và màn hiển thị. Cathode bằng kim loại được nối với điện áp âm, được đốt nóng và tạo ra các điện tử tự do. Màn hiển thị được phủ một lớp chất liệu phát quang và dẫn điện, được nối với điện áp dương và đóng vai trò một anode. Dưới tác dụng của điện trường cường độ cao trong ống phóng, điện tử rời khỏi cathode, được hội tụ thành chùm tia hướng về phía màn hiển thị. Cuộn lái tia có tác dụng lái chùm tia điện tử dịch chuyển theo hai chiều dọc và ngang màn hình. Khi chùm tia điện tử đập vào màn hiển thị sẽ tạo nên một điểm phát sáng. Cường độ điểm sáng phụ thuộc vào cường độ chùm tia và chất liệu phát sáng. Khi chùm tia mất đi hoặc chuyển hướng thì điểm vẫn còn lưu sáng một khoảng thời gian ngắn sau đó, thời gian lưu sáng phụ thuộc vào chất liệu phát sáng và cường độ chùm tia.



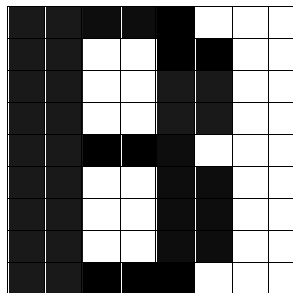
Ảnh trên màn hình CRT được tạo từ các điểm ảnh. Điểm ảnh được tạo ra khi cường độ chùm tia điện tử được tăng lên, điểm ảnh không xuất hiện khi chùm tia bị tắt đi. Các điểm ảnh được tạo theo từng dòng, từ trên xuống dưới. Một ảnh hoàn chỉnh được tạo ra trên màn hiển thị bởi các dòng chứa các điểm ảnh. Các điểm ảnh chỉ tồn tại trong một thời gian rất ngắn. Để có thể quan sát được ảnh cần làm tươi các điểm ảnh theo một chu kỳ xác định. Các điểm ảnh được làm tươi theo từng dòng, bắt đầu từ dòng thứ nhất. Các dòng được làm tươi tuần tự từ trên xuống dưới. Khi dòng cuối cùng được quét xong, quá trình làm tươi được bắt đầu lại từ dòng đầu tiên (hình vẽ).

V.4.2 Ghép nối màn hình với hệ Vi xử lý

Các thiết bị hiển thị được sử dụng ở máy vi tính PC đều là loại ánh xạ bộ nhớ. Bộ nhớ này được cả đơn vị xử lý trung tâm và thiết bị điều khiển màn hình cùng truy nhập và được gọi là bộ nhớ hiển thị. Thông tin cần hiển thị được đưa ra bộ nhớ hiển thị, thiết bị điều khiển màn hình CRTC liên tục đọc bộ nhớ này để đưa ra màn hình. Hình vẽ sau đây minh họa nguyên tắc ánh xạ từ bộ nhớ hiển thị ra màn hình trong chế độ văn bản :



Hình V.7 – Hiển thị ký tự trên màn hình CRT theo nguyên tắc ánh xạ bộ nhớ



Mỗi một ký tự trên màn hình là một ánh xạ của một ô nhớ hai byte trong bộ nhớ hiển thị. Byte đầu chứa mã ASCII của ký tự, byte thứ hai chứa thuộc tính (màu nền, màu chữ, có/không nhấp nháy) của ký tự. Vị trí của mã ký tự trong bộ nhớ xác định vị trí ký tự trên màn hình. Mã ký tự đầu tiên trong bộ nhớ hiển thị (ví dụ : mã 41H) được ánh xạ thành ký tự (ký tự A) lên góc trái trên của màn hình hiển thị, mã ký tự tiếp theo được ánh xạ thành ký tự tiếp theo v.v.

Phương pháp ánh xạ bộ nhớ cho phép chương trình máy tính có thể dễ dàng thay đổi nội dung màn hình hiển thị bằng cách thay đổi nội dung của bộ nhớ hiển thị.

Mỗi ký tự được hiển thị trên màn hình dưới dạng một ma trận 7×9 ^(*) điểm ảnh sáng/tối như trên hình vẽ :

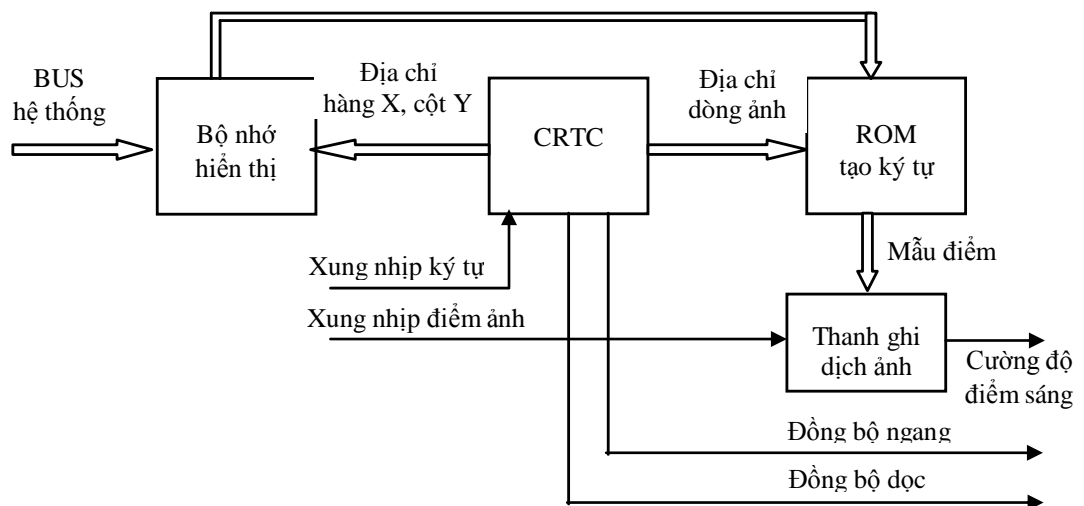
^(*) Cũng có những trường hợp sử dụng ma trận 5×7 , 7×9 , 7×12 và 9×14 điểm

Phương pháp hiển thị ánh xạ bộ nhớ không hoàn toàn phù hợp với việc hiển thị các đối tượng có hình dạng không bình thường và chuyển động nhanh, đáp ứng thời gian thực bị chậm vì cần phải thao tác nhiều điểm ảnh để dịch chuyển đối tượng.

V.4.3 Bộ điều khiển màn hình CRTC

Thiết bị giao diện màn hình (bộ điều khiển màn hình) CRTC thực hiện việc chuyển mã ký tự trong bộ nhớ hiển thị thành ký tự hiện trên màn hình. ở chế độ văn bản các mẫu ký tự chỉ được hiển thị ở các vị trí hàng và cột cố định (25 hàng x 80 cột).

Sơ đồ nguyên lý của thiết bị giao diện màn hình ở chế độ văn bản như sau:



Hình V.8 – Sơ đồ khối điều khiển hiển thị CRTC

Mỗi một ký tự trên màn hình chứa nhiều hàng điểm ảnh. CRTC có nhiệm vụ chuyển mỗi mã ASCII trong bộ nhớ hiển thị thành chuỗi các mẫu điểm ảnh, đưa mỗi mẫu nằm lên một dòng màn hình. Điều này được thực hiện nhờ bộ ROM tạo ký tự. ROM tạo ký tự chứa các hộp mẫu ký tự, mỗi hộp mẫu ký tự có kích thước 8 byte mang thông tin về ma trận điểm ảnh của một ký tự. Ví dụ hộp mẫu ký tự B có thể có dạng sau :

```
00000111
00110011
00110011
00110011
```

```

00000111
00110011
00110011
00110011
00000111

```

Nếu cần hiển thị 256 ký tự ASCII cần một ROM 2Kbyte, đủ chứa 256 hộp mẫu ký tự, mỗi hộp mẫu chiếm 8 ô nhớ liền nhau. Các hộp mẫu ký tự trong bộ ROM tạo ký tự được định vị bằng địa chỉ 11 bit, trong đó 8 bit địa chỉ cao xác định vị trí của hộp trong ROM, 3 bit địa chỉ thấp xác định vị trí của từng byte mẫu điểm ảnh trong hộp đó. Các mẫu ký tự được đặt trong ROM theo trật tự của bảng mã ASCII.

Nguyên lý hoạt động của thiết bị giao diện màn hình trong chế độ văn bản như sau: Giả sử cần hiển thị hai ký tự A và B tại các vị trí hàng 0 - cột 0 và hàng 0 - cột 1 trên màn hình. Mã ASCII của hai ký tự được đặt tại hai vị trí tương ứng trong bộ nhớ hiển thị (xem hình vẽ ở mục 2.2).

CRTC gửi địa chỉ hàng và cột màn hình cho bộ nhớ hiển thị (hàng=0, cột=0). Bộ nhớ hiển thị gửi mã ASCII của ký tự (ký tự A) cho ROM, mã ASCII của ký tự mang thông tin về địa chỉ của hộp mẫu ký tự trong ROM (8 bit địa chỉ cao). Tại cùng thời điểm này CRTC gửi địa chỉ của dòng mẫu điểm ảnh (dòng mẫu điểm 0) cho ROM (3 bit địa chỉ thấp). Hai địa chỉ này được kết hợp lại tạo thành địa chỉ (11 bit) cho phép truy nhập vào dòng mẫu điểm ảnh đầu tiên của ký tự (ký tự A) trong ROM và xuất nó ra thanh ghi dịch ảnh. Từ thanh ghi dịch ảnh, từng bit mẫu ảnh tuần tự được đưa ra màn hình.

Khi tất cả các bit mẫu ảnh từ thanh ghi dịch được đẩy ra màn hình, CRTC tiếp tục gửi địa chỉ hàng-cột (hàng=0, cột=1) cho bộ nhớ hiển thị và gửi địa chỉ dòng mẫu điểm ảnh (dòng mẫu điểm 0) cho ROM, bộ nhớ hiển thị gửi mã ASCII của ký tự (ký tự B) cho ROM. Dòng mẫu điểm ảnh đầu tiên của ký tự (ký tự B) được xuất ra thanh ghi dịch ảnh. Tương tự như thế các dòng mẫu điểm đầu tiên của tất cả các ký tự trên cùng một hàng màn hình được hiển thị, cho đến ký tự cuối cùng trên hàng.

CRTC tiếp tục gửi địa chỉ hàng-cột (hàng=0, cột=0) đến bộ nhớ hiển thị, nhưng địa chỉ dòng mẫu điểm ảnh bây giờ là 1 (dòng mẫu điểm 1) cho ROM. Bộ nhớ hiển thị gửi mã ASCII của ký tự A cho ROM, ROM xuất ra dòng mẫu điểm ảnh 1 của ký tự A. Dòng 1 của ký tự B được xuất ra theo cách tương tự. Các dòng điểm ảnh tiếp theo của ký tự lần lượt được hiển thị lên màn hình cho đến khi tất cả các dòng điểm ảnh của hàng văn bản đầu tiên (hàng 0) được hiển thị trên màn hình.

Các hàng văn bản tiếp theo cũng được hiển thị theo phương pháp nói trên.

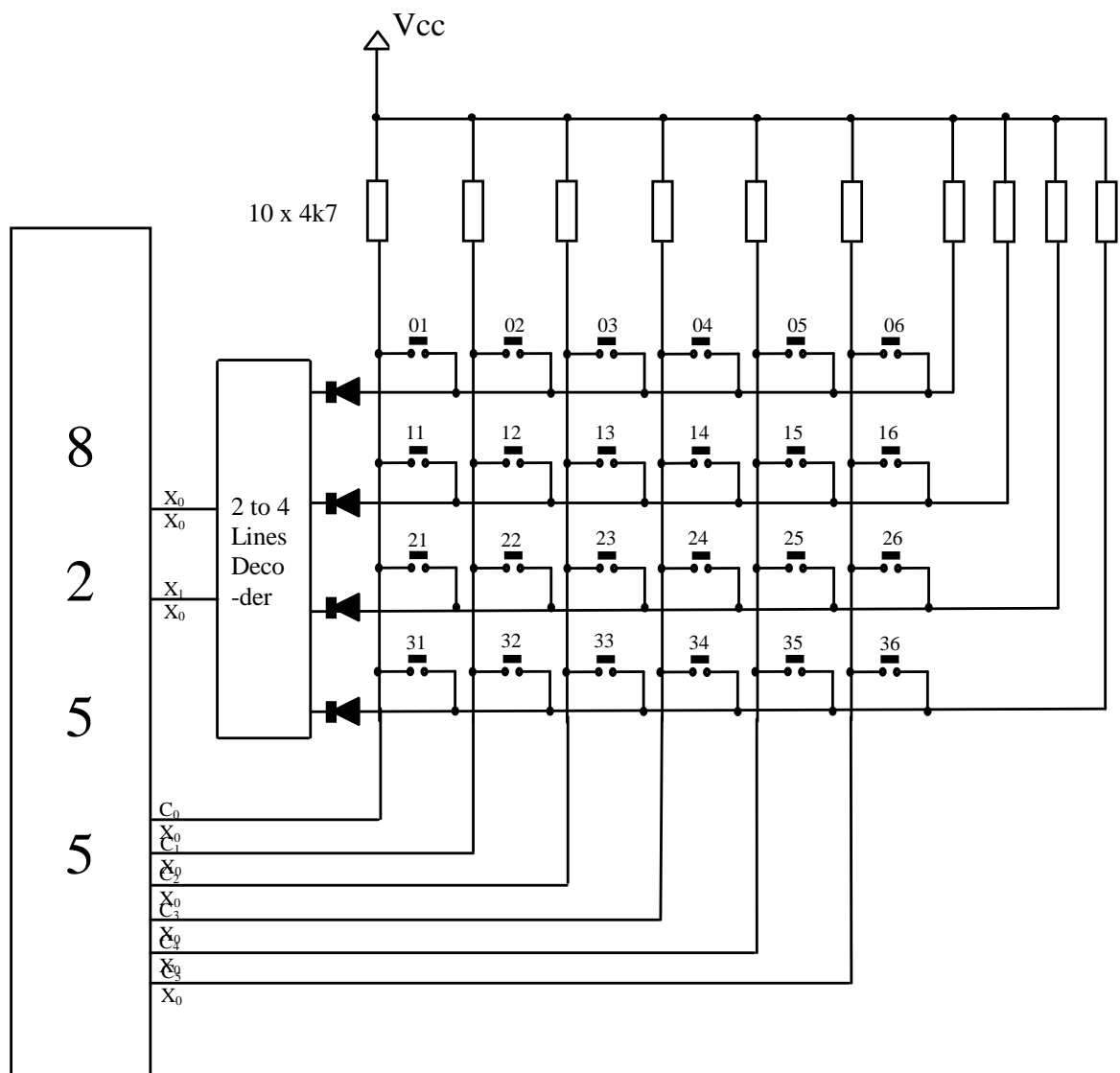
Trên thực tế hoạt động của CRTC phức tạp hơn. CRTC phải có khả năng hiển thị ở chế độ đồ họa. CRTC phải theo dõi thông tin về thuộc tính của ký tự hiển thị, phải tạo ra điểm nháy. CRTC cũng phải tạo ra hai

tín hiệu đồng bộ ảnh ngang - dọc và làm tươi màn hình. Tần số làm tươi tối thiểu là 50 Hz.

Ví dụ:

Thiết kế phân cứng mạch bàn phím 24 phím cho một hệ Vi xử lý +5V chuyên dụng sử dụng vi mạch PPI 8255, trình bày phương pháp tạo mã quét cho các phím.

Lời giải:



Bàn phím 24 phím sử dụng vi mạch 8255 có thể thiết kế như hình vẽ, sinh viên sử dụng kiến thức về mạch khả lập trình PPI8255 và phương pháp tạo mã quét cho phím để lập trình hợp ngữ chương trình đọc phím. Mã quét của 1 phím bất kỳ được tạo bởi 8 bit, $X_0X_1C_5C_4C_3C_2C_1C_0$.

CÂU HỎI ÔN TẬP CHƯƠNG V.

1. Mô tả nguyên lý tạo tín hiệu của phím cơ khí
2. Nguyên lý xác định phím bị nhấn trong bàn phím
3. Giải thích lưu đồ đọc và gán mã cho phím bị nhấn
4. Ghép nối bàn phím máy tính, sơ đồ cấu trúc và hoạt động đọc phím
5. Nguyên lý chỉ thị số của LED 7 thanh
6. Giải thích hoạt động hiển thị của bảng số theo phương pháp quét động
7. Nguyên lý hiển thị văn bản lên màn hình

PHỤ LỤC

PHỤ LỤC A

Bảng tóm tắt hệ lệnh của Trung tâm Vi xử lý họ x86

Từ gọi nhớ	Chức năng	Từ gọi nhớ	Chức năng
Các lệnh họ 80x86			
AAA	Chỉnh sau phép cộng 2 số dạng ASCII	CMP	So sánh toán hạng đích và gốc
AAD	Chỉnh hai số mã ASCII trước phép chia	CMPS	So sánh chuỗi Byte hay từ
AAM	Chỉnh sau phép nhân 2 số mã ASCII	CMPSB	So sánh xâu (byte)
AAS	Chỉnh sau phép trừ 2 số mã ASCII	CMPSW	So sánh xâu (từ)
ADC	Cộng có cờ nhớ	CWD	Biến đổi từ thành từ kép
ADD	Cộng 2 toán hạng	DAA	Hiệu chỉnh thập phân sau phép cộng
AND	Và từng bit tương ứng của 2 toán hạng	DAS	Hiệu chỉnh thập phân sau phép trừ
CALL	Gọi chương trình con	DEC	Giảm toán hạng đích đi 1
CBW	Chuyển byte thành từ	DIV	Chia không dấu
CLC	Xoá cờ nhớ	ESC	Thoát
CLD	Xoá cờ hướng	HLT	Treo
CLI	Xoá cờ ngắt	IDIV	Chia số nguyên
CMC	Lấy bù cờ nhớ	IMUL	Nhân số nguyên
IN	Đọc cổng vào ra	JS	Nhảy nếu có cờ dấu
INC	Tăng toán hạng đích lên 1	JZ	Nhảy nếu bằng 0
INT	Gọi ngắt	LAHF	Nạp 8 bit thấp của cờ vào AH
INTO	Ngắt nếu bị tràn	LDS	Nạp ô nhớ từ kép vào thanh ghi đoạn dữ liệu
IRET	Trở về chỗ bị ngắt	LEA	Nạp địa chỉ hiệu dụng
JA	Nhảy nếu ở trên	LES	Nạp con trỏ khi dùng ES
JAE	Nhảy nếu ở trên hoặc bằng	LOCK	Khoá bus
JB	Nhảy nếu thấp hơn	LODS	Nạp xâu
JBE	Nhảy nếu thấp hơn hoặc bằng	LODSB	Nạp xâu (byte)
JC	Nhảy nếu có cờ nhớ	LODSW	Nạp xâu (từ)
JCXZ	Nhảy nếu CX = 0	LOOP	Vòng lặp
JE	Nhảy nếu bằng	LOOPE	Lặp lại trong khi bằng
JG	Nhảy nếu lớn hơn	LOOPNE	Lặp lại khi không bằng
JGE	Nhảy nếu lớn hơn hoặc bằng	LOOPNZ	Lặp khi không bằng 0
JL	Nhảy nếu nhỏ hơn	LOOPZ	Lặp khi bằng 0
JLE	Nhảy nếu nhỏ hơn hoặc bằng	MOV	Chuyển nguồn tới đích
JMP	Nhảy không điều kiện	MOVS	Chuyển xâu
JNA	Nhảy nếu không ở trên	MOVSB	Chuyển xâu (byte)
JNAE	Nhảy nếu không ở trên hoặc bằng	MOVSW	Chuyển xâu (từ)
JNB	Nhảy nếu không ở dưới	MUL	Phép nhân
JNBE	Nhảy nếu không ở dưới hoặc bằng	NEG	Đảo dấu hay lấy bù 2

JNC	Nhảy nếu không có cờ nhớ	NOP	Không hành động
JNE	Nhảy nếu không bằng	NOT	Đảo dấu (lấy bù 1)
JNG	Nhảy nếu không lớn hơn	OR	Hoặc các bit tương ứng của 2 toán hạng
JNGE	Nhảy nếu không lớn hơn hoặc bằng	OUT	Viết cổng vào/ra
JNL	Nhảy nếu không nhỏ hơn	POP	Hồi phục nội dung (các thanh ghi, ...)
JNLE	nhảy nếu không nhỏ hơn hoặc bằng	POPF	Hồi phục nội dung các cờ
JNO	Nhảy nếu không tràn	PUSH	Đẩy nội dung (thanh ghi, ...) vào ngăn xếp
JNP	Nhảy nếu không có cờ chặn lẻ	PUSHF	Đẩy nội dung cờ vào ngăn xếp
JNS	Nhảy nếu không có cờ dấu	RCL	Quay trái qua cờ nhớ
JNZ	Nhảy nếu không bằng 0	RCR	Quay phải qua cờ nhớ
JO	Nhảy nếu có cờ tràn	REP	Lặp lại
JP	Nhảy nếu có cờ chặn lẻ	REPE	Lặp lại khi bằng
JPE	Nhảy nếu có cờ lẻ chặn	REPNE	Lặp lại khi không bằng
JPO	Nhảy nếu lẻ lẻ	REPZ	Lặp lại khi không bằng 0
REPZ	Lặp lại trong khi bằng 0	STC	Đặt cờ nhớ
RET	Trở về	STD	Đặt cờ hướng
ROL	Quay trái	STI	Đặt cờ ngắt
ROR	Quay phải	STOS	Lưu trữ xâu
SAHF	Lưu trữ AH vào byte thấp của cờ	STOSB	Lưu trữ xâu (byte)
SAL	Dịch trái số học	STOSW	Lưu trữ xâu (từ)
SAR	Dịch phải số học	SUB	Phép trừ
SBB	Trừ có mượn	TEST	Kiểm tra (nhân logic đích với gốc)
SCAS	Quét xâu	WAIT	Đợi
SCASB	Quét xâu (byte)	XCHG	Tráo đổi
SCANW	Quét xâu (từ)	XLAT	Chuyển đổi bảng
SHL	Dịch trái	XOR	Hoặc tuyệt đối tương ứng 2 số
SHR	Dịch phải		
Các lệnh chỉ có trong 80286, 80386 và 80486			
ARPL	Chỉnh trường RPL của bộ chọn	LSL	Nạp độ dài đoạn nhớ
BOUND	Kiểm tra biên của trường	LTR	Nạp thanh ghi nhiệm vụ
CLTS	Xoá cờ chuyển nhiệm vụ	OUTS	Xuất xâu ra cổng vào/ra
ENTER	Tạo khối các thông số để vào CTC	POPA	Phục hồi tất cả các thanh ghi đa năng
INS	Nhập xâu từ cổng vào/ra	PUSHA	Đẩy vào ngăn xếp các thanh ghi đa năng
LAR	Nạp quyền thâm nhập	SGDT	Lưu trữ thanh ghi bảng bộ mô tả toàn cục
LEAVE	RA khỏi CTC (chương trình con)	SIDT	Lưu trữ thanh ghi bảng bộ mô tả ngắt
LGDT	Nạp thanh ghi bảng bộ mô tả toàn cục	SLDT	Lưu trữ thanh ghi bảng bộ mô tả cục bộ
LIDT	Nạp thanh ghi bảng bộ mô tả ngắt	SMSW	Lưu trữ từ trạng thái máy
LLDT	Nạp thanh ghi bảng bộ mô tả cục bộ	STR	Lưu trữ thanh ghi nhiệm vụ
LMSW	Nạp từ trạng thái máy	VERR	Kiểm tra một bộ chọn đoạn để đọc
		VERW	Kiểm tra một bộ chọn đoạn để viết
Các lệnh chỉ có trong 80386 và 80486			

BSF	Quét bit về phía trước	SETL	Đặt byte nếu nhỏ hơn
BSR	Quét bit về phía sau	SETLE	Đặt byte nếu nhỏ hơn hoặc bằng
BT	Kiểm tra bit	SETNA	Đặt byte nếu không ở trên
BTC	Kiểm tra và đảo bit	SETNAE	Đặt byte nếu không ở trên hoặc bằng
BTR	Kiểm tra và xoá bit	SETNB	Đặt byte nếu không ở dưới
BTS	Kiểm tra và đặt bit	SETNBE	Đặt byte nếu không ở dưới hoặc bằng
CDQ	Biến đổi từ kép thành từ kép bốn	SETNC	Đặt byte nếu không nhớ
CMPSD	So sánh xâu (từ kép)	SETNE	Đặt byte nếu không bằng
CWDE	Biến đổi từ thành từ kép trong EAX	SETNG	Đặt byte nếu không lớn hơn
JECXZ	Nhảy nếu ECX bằng 0	SETNGE	Đặt byte nếu không lớn hơn hoặc bằng
LFS	Nạp con trỏ khi dùng FS	SETNL	Đặt byte nếu không nhỏ hơn
LGS	Nạp con trỏ khi dùng GS	SETNLE	Đặt byte nếu không nhỏ hơn hoặc bằng
LSS	Nạp con trỏ khi dùng SS	SETNO	Đặt byte nếu không tràn
LODSD	Nạp xâu (từ kép)	SETNP	Đặt byte nếu không có chặn lẻ
MOVSD	Chuyển xâu (từ kép)	SETNS	Đặt byte nếu không dấu
MOVSB	Chuyển với Sigh-eXtend	SETNZ	Đặt byte nếu không bằng 0
MOVZB	Chuyển với Zero-eXtend	SETO	Đặt byte nếu tràn
SCASD	Quét xâu (từ kép)	SETP	Đặt byte nếu có chặn lẻ
SETA	Đặt byte nếu ở trên	SETPE	Đặt byte nếu chặn lẻ chặn
SETAE	đặt byte nếu ở trên hoặc bằng	SETPO	Đặt byte nếu có chặn lẻ lẻ
SETB	Đặt byte nếu ở dưới	SETS	Đặt byte nếu có dấu
SETBE	Đặt byte nếu ở dưới hoặc bằng	SETZ	Đặt byte nếu bằng 0
SETC	Đặt byte nếu có cờ nhớ	SHLD	Dịch trái (từ kép)
SETE	Đặt byte nếu bằng	SHRD	Dịch phải (từ kép)
SETG	Đặt byte nếu lớn hơn	STOSD	Lưu trữ xâu (từ kép)
SETGE	Đặt byte nếu lớn hơn hoặc bằng		
Các lệnh chỉ có trong 80486			
BSWAP	Hoán chuyển byte	INVLPG	Vô hiệu hoá TLB (cho chế độ trang)
CMPXCHG	So sánh và trao đổi	WBINVD	Ghi trở lại bộ nhớ chính vào nhớ ngầm
INVD	Vô hiệu hoá bộ nhớ ngầm	XADD	Hoán chuyển và cộng

PHỤ LỤC B**Bảng lũy thừa 2ⁿ**

2 ⁿ	n	2 ⁻ⁿ
1	0	1
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.0625
32	5	0.03125
64	6	0.015625
128	7	0.0078125
256	8	0.00390625
512	9	0.001953125
1,024	10	0.000976563
2,048	11	0.0004882815
4,096	12	0.00024414125
8,192	13	0.000122070625
16,384	14	0.000061035156
32,768	15	0.000030517578
65,536	16	0.000015258789
131,072	17	0.000007629395
262,144	18	0.000003814697
524,288	19	0.000001907349
1,048,576	20	0.000000953674
2,097,152	21	0.000000476837
4,194,304	22	0.000000238419
8,388,608	23	0.000000119209
16,777,216	24	0.000000059605
33,554,432	25	0.000000029802
67,108,864	26	0.000000014901
134,217,728	27	0.000000007451
268,435,456	28	0.000000003725
536,870,912	29	0.000000001863
1,073,741,824	30	0.000000000931
2,147,483,648	31	0.000000000466
4,294,967,296	32	0.000000000233

Từ 2⁻¹⁴ là giá trị đã làm tròn lấy 10 số sau dấu phẩy

PHỤ LỤC C

Bảng mã ASCII (American Standard Code for Information Interchange)**Kí tự ASCII in được**

Nhị phân	Thập phân	Hệ Hexa	Đồ hoa	Nhị phân	Thập phân	Hệ Hexa	Đồ hoa	Nhị phân	Thập phân	Hệ Hexa	Đồ hoa
010 0000	32	20		100 0000	64	40	@	110 0000	96	60	`
010 0001	33	21	!	100 0001	65	41	A	110 0001	97	61	a
010 0010	34	22	"	100 0010	66	42	B	110 0010	98	62	b
010 0011	35	23	#	100 0011	67	43	C	110 0011	99	63	c
010 0100	36	24	\$	100 0100	68	44	D	110 0100	100	64	d
010 0101	37	25	%	100 0101	69	45	E	110 0101	101	65	e
010 0110	38	26	&	100 0110	70	46	F	110 0110	102	66	f
010 0111	39	27	'	100 0111	71	47	G	110 0111	103	67	g
010 1000	40	28	(100 1000	72	48	H	110 1000	104	68	h
010 1001	41	29)	100 1001	73	49	I	110 1001	105	69	i
010 1010	42	2A	*	100 1010	74	4A	J	110 1010	106	6A	j
010 1011	43	2B	+	100 1011	75	4B	K	110 1011	107	6B	k
010 1100	44	2C	,	100 1100	76	4C	L	110 1100	108	6C	l
010 1101	45	2D	-	100 1101	77	4D	M	110 1101	109	6D	m
010 1110	46	2E	.	100 1110	78	4E	N	110 1110	110	6E	n
010 1111	47	2F	/	100 1111	79	4F	O	110 1111	111	6F	o
011 0000	48	30	0	101 0000	80	50	P	111 0000	112	70	p
011 0001	49	31	1	101 0001	81	51	Q	111 0001	113	71	q
011 0010	50	32	2	101 0010	82	52	R	111 0010	114	72	r
011 0011	51	33	3	101 0011	83	53	S	111 0011	115	73	s
011 0100	52	34	4	101 0100	84	54	T	111 0100	116	74	t
011 0101	53	35	5	101 0101	85	55	U	111 0101	117	75	u
011 0110	54	36	6	101 0110	86	56	V	111 0110	118	76	v
011 0111	55	37	7	101 0111	87	57	W	111 0111	119	77	w
011 1000	56	38	8	101 1000	88	58	X	111 1000	120	78	x
011 1001	57	39	9	101 1001	89	59	Y	111 1001	121	79	y
011 1010	58	3A	:	101 1010	90	5A	Z	111 1010	122	7A	z
011 1011	59	3B	;	101 1011	91	5B	[111 1011	123	7B	{
011 1100	60	3C	<	101 1100	92	5C	\	111 1100	124	7C	
011 1101	61	3D	=	101 1101	93	5D]	111 1101	125	7D	}
011 1110	62	3E	>	101 1110	94	5E	^	111 1110	126	7E	~
011 1111	63	3F	?	101 1111	95	5F	_				

Kí tự điều khiển ASCII

Nhị phân	Thập phân	Hệ Hexa	Viết tắt	Truy nhập bàn phím	Tên/Ý nghĩa
000 0000	0	00	NUL	^@	Null character
000 0001	1	01	SOH	^A	Start of Header
000 0010	2	02	STX	^B	Start of Text
000 0011	3	03	ETX	^C	End of Text
000 0100	4	04	EOT	^D	End of Transmission
000 0101	5	05	ENO	^E	Enquiry
000 0110	6	06	ACK	^F	Acknowledgement
000 0111	7	07	BEL	^G	Bell
000 1000	8	08	BS	^H	Backspace
000 1001	9	09	HT	^I	Horizontal Tab
000 1010	10	0A	LF	^J	Line feed
000 1011	11	0B	VT	^K	Vertical Tab
000 1100	12	0C	FF	^L	Form feed
000 1101	13	0D	CR	^M	Carriage return
000 1110	14	0E	SO	^N	Shift Out
000 1111	15	0F	SI	^O	Shift In
001 0000	16	10	DLE	^P	Data Link Escape
001 0001	17	11	DC1	^Q	Device Control 1 — oft.
001 0010	18	12	DC2	^R	Device Control 2
001 0011	19	13	DC3	^S	Device Control 3 — oft.
001 0100	20	14	DC4	^T	Device Control 4
001 0101	21	15	NAK	^U	Negative
001 0110	22	16	SYN	^V	Synchronous Idle
001 0111	23	17	ETB	^W	End of Trans. Block
001 1000	24	18	CAN	^X	Cancel
001 1001	25	19	EM	^Y	End of Medium
001 1010	26	1A	SUB	^Z	Substitute
001 1011	27	1B	ESC	^[_ hay ESC	Escape
001 1100	28	1C	FS	^[\	File Separator
001 1101	29	1D	GS	^[_	Group Separator
001 1110	30	1E	RS	^^	Record Separator
001 1111	31	1F	US	^	Unit Separator
111 1111	127	7F	DEL	DEL hay Backspace	Delete

Bảng mã EBCDIC

(Extended Binary Coded Decimal Interchange Code)

Dec	Hex	Code	Dec	Hex	Code	Dec	Hex	Code	Dec	Hex	Code
0	00	NUL	32	20		64	40	space	96	60	-
1	01	SOH	33	21		65	41		97	61	/
2	02	STX	34	22		66	42		98	62	
3	03	ETX	35	23		67	43		99	63	
4	04		36	24		68	44		100	64	
5	05	HT	37	25	LF	69	45		101	65	
6	06		38	26	ETB	70	46		102	66	
7	07	DEL	39	27	ESC	71	47		103	67	
8	08		40	28		72	48		104	68	
9	09		41	29		73	49		105	69	
10	0A		42	2A		74	4A	[106	6A	
11	0B	VT	43	2B		75	4B	.	107	6B	,
12	0C	FF	44	2C		76	4C	<	108	6C	%
13	0D	CR	45	2D	ENQ	77	4D	(109	6D	_
14	0E	SO	46	2E	ACK	78	4E	+	110	6E	>
15	0F	SI	47	2F	BEL	79	4F	!	111	6F	?
16	10	DLE	48	30		80	50	&	112	70	
17	11		49	31		81	51		113	71	
18	12		50	32	SYN	82	52		114	72	
19	13		51	33		83	53		115	73	
20	14		52	34		84	54		116	74	
21	15		53	35		85	55		117	75	
22	16	BS	54	36		86	56		118	76	
23	17		55	37	EOT	87	57		119	77	
24	18	CAN	56	38		88	58		120	78	
25	19	EM	57	39		89	59		121	79	'
26	1A		58	3A		90	5A	!]	122	7A	:
27	1B		59	3B		91	5B	\$	123	7B	#
28	1C	IFS	60	3C		92	5C	*	124	7C	@
29	1D	IGS	61	3D	NAK	93	5D)	125	7D	'
30	1E	IRS	62	3E		94	5E	;	126	7E	=
31	1F	IUS	63	3F	SUB	95	5F	^	127	7F	"

Bảng mã EBCDIC (tiếp theo)

Dec	Hex	Code	Dec	Hex	Code	Dec	Hex	Code	Dec	Hex	Code
128	80		160	A0		192	C0	{	224	E0	\
129	81	a	161	A1	~	193	C1	A	225	E1	
130	82	b	162	A2	s	194	C2	B	226	E2	S
131	83	c	163	A3	t	195	C3	C	227	E3	T
132	84	d	164	A4	u	196	C4	D	228	E4	U
133	85	e	165	A5	v	197	C5	E	229	E5	V
134	86	f	166	A6	w	198	C6	F	230	E6	W
135	87	g	167	A7	x	199	C7	G	231	E7	X
136	88	h	168	A8	y	200	C8	H	232	E8	Y
137	89	i	169	A9	z	201	C9	I	233	E9	Z
138	8A		170	AA		202	CA		234	EA	
139	8B		171	AB		203	CB		235	EB	
140	8C		172	AC		204	CC		236	EC	
141	8D		173	AD		205	CD		237	ED	
142	8E		174	AE		206	CE		238	EE	
143	8F		175	AF		207	CF		239	EF	
144	90		176	B0		208	D0	}	240	F0	0
145	91	j	177	B1		209	D1	J	241	F1	1
146	92	k	178	B2		210	D2	K	242	F2	2
147	93	l	179	B3		211	D3	L	243	F3	3
148	94	m	180	B4		212	D4	M	244	F4	4
149	95	n	181	B5		213	D5	N	245	F5	5
150	96	o	182	B6		214	D6	O	246	F6	6
151	97	p	183	B7		215	D7	P	247	F7	7
152	98	q	184	B8		216	D8	Q	248	F8	8
153	99	r	185	B9		217	D9	R	249	F9	9
154	9A		186	BA		218	DA		250	FA	
155	9B		187	BB		219	DB		251	FB	
156	9C		188	BC		220	DC		252	FC	
157	9D		189	BD		221	DD		253	FD	
158	9E		190	BE		222	DE		254	FE	
159	9F		191	BF		223	DF		255	FF	

PHỤ LỤC D

Các nhóm lệnh của μC8051

1. Tạo vòng lặp và lệnh nhảy

a. Tạo vòng lặp

- Quá trình lặp lại chuỗi lệnh với một số lần nhất định gọi là vòng lặp.

Vòng lặp trong 8051 được thực hiện bằng lệnh “DJNZ thanh ghi, nhãn“

Để tổ chức vòng lặp lồng nhau cần sử dụng 2 thanh ghi để lưu số đếm.

b. Lệnh nhảy

Lệnh nhảy có điều kiện

Lệnh	Ý nghĩa
JZ	Nhảy nếu A=0
JNZ	Nhảy nếu A khác 0
DJNZ	Giảm và nhảy nếu A khác 0
CJNE A,byte	Nhảy nếu A khác byte
CJNE re,#data	Nhảy nếu byte khác data
JC	Nhảy nếu CY=1
JNC	Nhảy nếu CY=0
JB	Nhảy nếu bit=1
JNB	Nhảy nếu bit=0
JBC	Nhảy nếu bit=1 và xoá nó

Lệnh nhảy không điều kiện :

Lệnh nhảy dài :

Là lệnh 3 byte. Byte đầu là mã lệnh, 2 byte còn lại là địa chỉ 16 bit của đích. Địa chỉ đích 2 byte cho phép lệnh có thể nhảy đến bất kỳ vị trí nhớ nào trong không gian nhớ 0000 đến FFFF

Lệnh nhảy ngắn :

Lệnh nhảy ngắn là lệnh 2 byte. Byte đầu tiên là mã lệnh, byte thứ 2 là địa chỉ tương đối của địa chỉ đích. Địa chỉ đích tương đối có nghĩa là so với giá trị của bộ đếm chương trình.

2. Lệnh gọi Call

Lệnh call dùng để gọi chương trình con

Lệnh gọi dài Lcall (longcall)

Đây là lệnh 3 byte. Byte đầu là mã lệnh, 2 byte còn lại là địa chỉ của chương trình con đích. sau khi thực hiện xong 1 chương trình con, để 8051 biết được chỗ quay trở về thì địa chỉ của lệnh đứng ngay sau lệnh gọi Lcall sẽ được tự động cất vào ngăn xếp.

Lệnh gọi tuyệt đối Acall

Lệnh Acall là lệnh 2 byte, địa chỉ đích của chương trình con phải nằm trong khoảng 2 Kbyte địa chỉ vì chỉ có 11 bit của 2 byte được dùng để xác định địa chỉ. Thực tế một số biến thể 8051 chỉ có 1 Kbyte ROM trên chip. Trong những trường hợp đó, sử dụng lệnh Acall có thể tiết kiệm được một số byte bộ nhớ của không gian ROM chương trình so với lệnh Lcall.

3. Nhóm lệnh cơ bản của 8051

Tập lệnh của 8051 được chia thành 5 nhóm:

- Số học.
- Luận lý.
- Chuyển dữ liệu.
- Chuyển điều khiển.
- Rẽ nhánh.

Các chi tiết thiết lập lệnh:

Rn	Thanh ghi R0 đến R7 của bank thanh ghi được chọn.
Data	8 bit địa chỉ vùng dữ liệu bên trong. Nó có thể là vùng RAM dữ liệu trong (0-127) hoặc các thanh ghi chức năng đặc biệt.
@Ri	8 bit vùng RAM dữ liệu trong (0-125) được đánh giá địa chỉ gián tiếp qua thanh ghi R0 hoặc R1.
#data	Hằng 8 bit chức trong câu lệnh.
#data 16	Hằng 16 bit chứa trong câu lệnh.
Addr16	16 bit địa chỉ đích được dùng trong lệnh LCALL và LJMP.
Addr11	11 bit địa chỉ đích được dùng trong lệnh LCALL và AJMP.
Rel	Byte offset 8 bit có dấu được dùng trong lệnh SJMP và những lệnh nhảy có điều kiện.
Bit	Bit được định địa chỉ trực tiếp trong RAM dữ liệu nội hoặc các thanh ghi chức năng đặc biệt.

a. Nhóm lệnh xử lý số học

ADD A,Rn	(1byte 1 chu kỳ máy) : cộng nội dung thanh ghi Rn vào thanh ghi A.
ADD A,data	(21): Cộng trực tiếp 1 byte vào thanh ghi A.
ADD A,@Ri	(11): Cộng gián tiếp nội dung RAM chứa tại địa chỉ được khai báo trong Ri vào thanh ghi A.
ADD A,#data	(21): Cộng dữ liệu tức thời vào A.
ADD A,Rn	(11): Cộng thanh ghi vào A.
ADD A,data	(21): Cộng trực tiếp byte dữ liệu và cờ nhớ vào A.
ADDC A,@Ri	(11): Cộng gián tiếp nội dung RAM và cờ nhớ vào A.
ADDC A,#data	(21): Cộng dữ liệu tức thời và cờ nhớ vào A.

SUBB A,Rn	(11): Trừ nội dung thanh ghi A cho nội dung thanh ghi Rn và cờ nhớ.
SUBB A,data	(21): Trừ trực tiếp A cho một số và cờ nhớ.
SUBB A,@Ri	(11): Trừ gián tiếp A cho một số và cờ nhớ.
SUBB A,#data	(21): Trừ nội dung A cho một số tức thời và cờ nhớ.
INC A	(11): Tăng nội dung thanh ghi A lên 1.
INC Rn	(11): Tăng nội dung thanh ghi Rn lên 1.
INC data	(21): Tăng dữ liệu trực tiếp lên 1.
INC @Ri	(11): Tăng gián tiếp nội dung vùng RAM lên 1.
DEC A	(11): Giảm nội dung thanh ghi A xuống 1.
DEC Rn	(11): Giảm nội dung thanh ghi Rn xuống 1.
DEC data	(21): Giảm dữ liệu trực tiếp xuống 1
DEC @Ri	(11): Giảm gián tiếp nội dung vùng RAM xuống 1.
INC DPTR	(12): Tăng nội dung con trỏ dữ liệu lên 1.
MUL AB	(14): Nhân nội dung thanh ghi A với nội dung thanh ghi B.
DIV AB	(14): Chia nội dung thanh ghi A cho nội dung thanh ghi B.
DA A	(11): hiệu chỉnh thập phân thanh ghi A.

b. Nhóm lệnh logic

ANL A,Rn	(11): AND nội dung thanh ghi A với nội dung thanh ghi Rn.
ANL A,data	(21): AND nội dung thanh ghi A với dữ liệu trực tiếp.
ANL A,@Ri	(11): AND nội dung thanh ghi A với dữ liệu gián tiếp trong RAM.
ANL A,#data	(21): AND nội dung thanh ghi với dữ liệu tức thời.
ANL data,A	(21): AND một dữ liệu trực tiếp với A.
ANL data,#data	(32): AND một dữ liệu trực tiếp với A một dữ liệu tức thời.

ANL C,bit	(22): AND cờ nhớ với 1 bit trực tiếp.
ANL C,/bit	(22): AND cờ nhớ với bù 1 bit trực tiếp.
ORL A,Rn	(11): OR thanh ghi A với thanh ghi Rn.
ORL A,data	(21): OR thanh ghi A với một dữ liệu trực tiếp.
ORL A,@Ri	(11): OR thanh ghi A với một dữ liệu gián tiếp.
ORL A,#data	(21): OR thanh ghi A với một dữ liệu tức thời.
ORL data,A	(21): OR một dữ liệu trực tiếp với thanh ghi A.
ORL data,#data	(31) :OR một dữ liệu trực tiếp với một dữ liệu tức thời.
ORL C,bit	(22): OR cờ nhớ với một bit trực tiếp.
ORL C,/bit	(22): OR cờ nhớ với bù của một bit trực tiếp.
XRL A,Rn	(11): XOR thanh ghi A với thanh ghi Rn.
XRL A,data	(21): XOR thanh ghi A với một dữ liệu trực tiếp.
XRL A,@Ri	(11): XOR thanh ghi A với một dữ liệu gián tiếp.
XRL A,#data	(21): XOR thanh ghi A với một dữ liệu tức thời.
XRL data,A	(21): XOR một dữ liệu trực tiếp với thanh ghi A.
XRL data,#data	(31): XOR một dữ liệu trực tiếp với một dữ liệu tức thời.
SETB C	(11): Thiết lập cờ nhớ.
SETB bit	(21): Thiết lập một bit trực tiếp.
CLR A	(11): Xóa thanh ghi A.
CLR C	(11): Xóa cờ nhớ.
CPL A	(11): Bù nội dung thanh ghi A.
CPL C	(11): Bù cờ nhớ.
CPL bit	(21): Bù một bit trực tiếp.
RL A	(11): Quay trái nội dung thanh ghi A.
RLC A	(11): Quay trái nội dung thanh ghi A qua cờ nhớ.
RR A	(11): Quay phải nội dung thanh ghi A.

RRC A	(11): Quay phải nội dung thanh ghi A qua cờ nhớ.
SWAP	(11): Quay trái nội dung thanh ghi A 1 nibble (1/2byte).

c. Nhóm lệnh chuyển dữ liệu

MOV A,Rn	(11): Chuyển nội dung thanh ghi Rn vào thanh ghi A.
MOV A,data	(21): Chuyển dữ liệu trực tiếp vào thanh ghi A.
MOV A,@Ri	(11): Chuyển dữ liệu gián tiếp vào thanh ghi A.
MOV A,#data	(21): Chuyển dữ liệu tức thời vào thanh ghi A.
MOV Rn,data	(22): Chuyển dữ liệu trực tiếp vào thanh ghi Rn.
MOV Rn,#data	(21): Chuyển dữ liệu tức thời vào thanh ghi Rn.
MOV data,A	(21): Chuyển nội dung thanh ghi A vào một dữ liệu trực tiếp.
MOV data,Rn	(22): Chuyển nội dung thanh ghi Rn vào một dữ liệu trực tiếp.
MOV data,data	(32): Chuyển một dữ liệu trực tiếp vào một dữ liệu trực tiếp.
MOV data,@Ri	(22): Chuyển một dữ liệu gián tiếp vào một dữ liệu gián tiếp.
MOV data,#data	(32): Chuyển một dữ liệu tức thời vào một dữ liệu trực tiếp.
MOV @Ri,A	(11): Chuyển nội dung thanh ghi A vào một dữ liệu gián tiếp.
MOV @Ri,data	(22): Chuyển một dữ liệu trực tiếp vào một dữ liệu gián tiếp.
MOV @Ri,#data	(21): Chuyển dữ liệu tức thời vào dữ liệu gián tiếp.
MOV DPTR,#data	(32): Chuyển một hằng 16 bit vào thanh ghi con trỏ dữ liệu.
MOV C,bit	(21): Chuyển một bit trực tiếp vào cờ nhớ.

MOV bit,C	(22): Chuyển cờ nhớ vào một bit trực tiếp.
MOV A,@A+DPTR	(12): Chuyển byte bộ nhớ chương trình có địa chỉ là @A+DPTR vào thanh ghi A.
MOVC A,@A+PC	(12): Chuyển byte bộ nhớ chương trình có địa chỉ là @A+PC vào thanh ghi A.
MOVX A,@Ri	(12): Chuyển dữ liệu ngoài (8 bit địa chỉ) vào thanh ghi A.
MOVX A,@DPTR	(12): Chuyển dữ liệu ngoài (16 bit địa chỉ) vào thanh ghi A.
MOVX @Ri,A	(12): Chuyển nội dung A ra dữ liệu ngoài (8 bit địa chỉ).
MOVX @DPTR,A	(12): Chuyển nội dung A ra dữ liệu bên ngoài (16 bit địa chỉ).
PUSH data	(22): Chuyển dữ liệu trực tiếp vào ngăn xếp và tăng SP.
POP data	(22): Chuyển dữ liệu trực tiếp vào ngăn xếp và giảm SP.
XCH A,Rn	(11): Trao đổi dữ liệu giữa thanh ghi Rn và thanh ghi A.
XCH A,data	(21): Trao đổi giữa thanh ghi A và một dữ liệu trực tiếp.
XCH A,@Ri	(11): Trao đổi giữa thanh ghi A và một dữ liệu gián tiếp.
XCHD A,@R	(11): Trao đổi giữa nibble thấp (LSN) của thanh ghi A và LSN của dữ liệu gián tiếp.

d. Nhóm lệnh chuyển điều khiển

ACALL addr11	(22): Gọi chương trình con dùng địa chỉ tuyệt đối.
LCALL addr16	(32): Gọi chương trình con dùng địa chỉ dài.

RET	(12): Trở về từ lệnh gọi chương trình con.
RETI	(12): Trở về từ lệnh gọi ngắt.
AJMP addr11	(22): Nhảy tuyệt đối.
LJMP addr16	(32): Nhảy dài.
SJMP rel	(22): Nhảy ngắn.
JMP @A+DPTR	(12): Nhảy gián tiếp từ con trỏ dữ liệu.
JZ rel	(22): Nhảy nếu A=0.
JNZ rel	(22): Nhảy nếu A không bằng 0.
JC rel	(22): Nhảy nếu cờ nhớ được đặt.
JNC rel	(22): Nhảy nếu cờ nhớ không được đặt.
JB bit,rel	(32): Nhảy tương đối nếu bit trực tiếp được đặt.
JNB bit,rel	(32): Nhảy tương đối nếu bit trực tiếp không được đặt.
JBC bit,rel	(32): Nhảy tương đối nếu bit trực tiếp được đặt, rồi xóa bit.
CJNE A,data,rel	(32): So sánh dữ liệu trực tiếp với A và nhảy nếu không bằng.
CJNE A,#data,rel	(32): So sánh dữ liệu tức thời với A và nhảy nếu không bằng.
CJNE Rn,#data,rel	(32): So sánh dữ liệu tức thời với nội dung thanh ghi Rn và nhảy nếu không bằng.
CJNE @Ri,#data,rel	(32): So sánh dữ liệu tức thời với dữ liệu gián tiếp và nhảy nếu không bằng.
DJNZ Rn,rel	(22): Giảm thanh ghi Rn và nhảy nếu không bằng 0.
DJNZ data	(32): Giảm dữ liệu trực tiếp và nhảy nếu không bằng 0.

TÀI LIỆU THAM KHẢO

- [1] Nguyễn Tăng Cường, Phan Quốc Khánh: *Cấu trúc và lập trình họ Vi điều khiển 8051*. NXB KH&KT Hà Nội-2004
- [2] Vũ Chấn Hưng: *Giáo trình Kiến trúc máy tính* NXB Giao thông vận tải - Hà Nội 2003
- [3] Văn Thế Minh: *Kỹ thuật Vi xử lý* – NXB Thống kê – Hà Nội 1983
- [4] Phòng Kỹ thuật số - Viện Khoa học Tính toán và Điều khiển: *Kỹ thuật Vi xử lý* - Nhà Xuất bản Thống kê – Hà Nội 1983
- [5] Alan Clements: *Principles of Computer Hardware* – PWS-KENT Publishing Company – Boston 1992
- [6] Intel[®] Corporation: *Component Data Catalog 1982*
- [7] David Hergert, Nancy Thibeault: *PC Architecture from Assembly Language To C*. Prentice-Hall, Inc. 1997
- [8] Christopher L. Morgan and Mitchell Waite: *8086/8088 16-bit Micro-Processor Primer* – McGraw-Hill, Inc. 1982
- [9] V.M. Rooney: *Microprocessors and Microcomputers* - McMilan Publishing Company – New York 1983
- [10] James L., Turley: *Advanced 80386 programming techniques* - Osborne Mc Graw-Hill 1988

