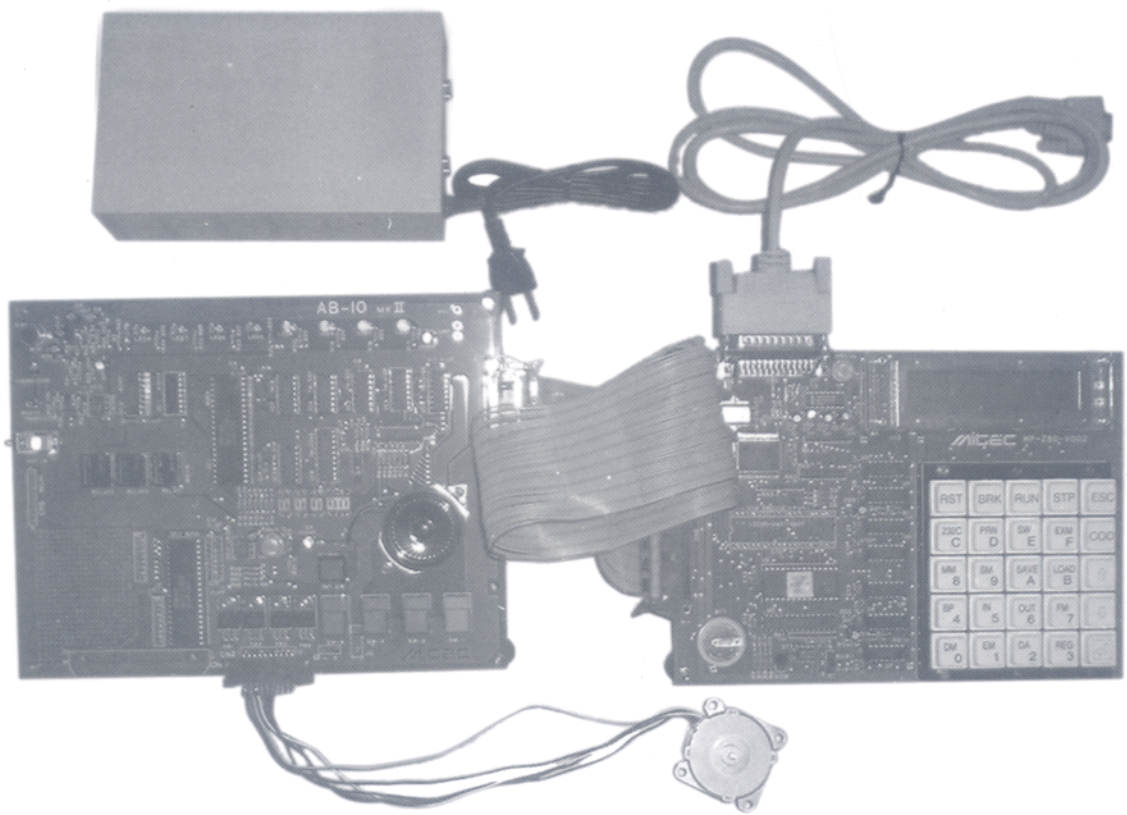
JICA-HIC, DỰ ÁN TĂNG CƯỜNG KHẢ NĂNG ĐÀO TẠO CÔNG NHÂN KỸ THUẬT
TRƯỜNG CAO ĐẲNG CÔNG NGHIỆP HÀ NỘI
BAN ĐIỀU KHIỂN ĐIỆN

Chủ biên: KIỀU XUÂN THỰC
Sửa lần II: NGUYỄN THANH HÀ
Cùng tập thể giáo viên ban Điều Khiển Điện

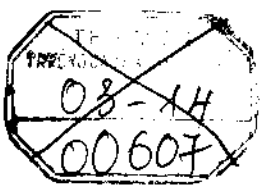
KỸ THUẬT VI XỬ LÝ



NHÀ XUẤT BẢN LAO ĐỘNG XÃ HỘI

JICA-HIC, DỰ ÁN TĂNG CƯỜNG KHẢ NĂNG ĐÀO TẠO CÔNG NHÂN KỸ THUẬT
TRƯỜNG CAO ĐẲNG CÔNG NGHIỆP HÀ NỘI
BAN ĐIỀU KHIỂN ĐIỆN

KỸ THUẬT VI XỬ LÝ



NHÀ XUẤT BẢN LAO ĐỘNG XÃ HỘI

LỜI NÓI ĐẦU

Khoa học và công nghệ ngày càng phát triển, chúng ta cần cung cấp kiến thức khoa học công nghệ cho công nhân trẻ, những người mong muốn được học tập và nghiên cứu để phục vụ sự nghiệp công nghiệp hóa – hiện đại hóa đất nước.

Để đáp ứng nhu cầu trên, Dự án **"Tăng cường Khả năng Đào tạo Công nhân kỹ thuật tại trường Cao đẳng Công nghiệp Hà Nội"** đã được thành lập và bắt đầu hoạt động từ ngày 1 tháng 4 năm 2000 theo thoả thuận hợp tác kỹ thuật giữa hai chính phủ Việt Nam và Nhật Bản. Đây là dự án hợp tác kỹ thuật về dạy nghề trên 3 lĩnh vực: gia công kim loại tấm, điều khiển điện và gia công cơ khí. Việc biên soạn giáo trình phục vụ cho đào tạo, chuyển giao công nghệ là một hoạt động quan trọng của dự án.

Cuốn sách **"Kỹ thuật Vi xử lý"** được viết nhằm mục đích cung cấp cho học sinh, sinh viên những kiến thức cơ bản nhất về kỹ thuật vi xử lý nói chung và về bộ vi xử lý Z80 của hãng Zilog nói riêng, đồng thời để phục vụ cho việc giảng dạy thực hành môn học Vi xử lý chúng tôi cũng đưa ra các bài tập ứng dụng trên modul MP-85MKII và modul I/O AB-10MKII của hãng Mitec.

Trong quá trình biên soạn, mặc dù đã rất cố gắng nhưng chắc chắn không tránh khỏi những sai sót, chúng tôi rất mong nhận được những ý kiến đóng góp của bạn đọc.

Hà Nội, tháng 3 năm 2003

Các tác giả

KS. KIỀU XUÂN THỰC

KS. NGUYỄN THANH HÀ

MỤC LỤC

Chương 1

THÔNG TIN VÀ BIỂU DIỄN THÔNG TIN TRONG MÁY TÍNH

1. Các hệ đếm

1.1. Hệ thập phân

Hàng ngày con người thường dùng hệ đếm cơ số mười (Hệ thập phân) để biểu diễn các giá trị số. Trong hệ này chúng ta dùng các số trong phạm vi từ 0 đến 9 để biểu diễn các giá trị. Hệ thập phân là một hệ đếm phụ thuộc vị trí, có nghĩa là mỗi chữ số gắn liền với một luỹ thừa 10 với số mũ phụ thuộc vào vị trí của con số đó trong số được biểu diễn. Ví dụ số 1234 sẽ bằng 1 nghìn, 2 trăm, 3 chục và 4 đơn vị:

$$1234 = 1x10^3 + 2x10^2 + 3x10^1 + 4x10^0$$

Nhưng trong máy tính các vi mạch chỉ có thể xử lý thông tin dưới dạng mã nhị phân nên chúng ta cần phải xem xét cách biểu diễn các số dưới dạng mã nhị phân như thế nào.

1.2. Hệ nhị phân

Trong hệ nhị phân (hay hệ 2), cơ số đếm là 2 nên chỉ sử dụng hai số là 0 và 1 để biểu diễn các giá trị số.

Ví dụ: chuỗi số nhị phân 101011 dùng để biểu diễn số:

$101011 = 1x2^5 + 0x2^4 + 1x2^3 + 0x2^2 + 1x2^1 + 1x2^0 = 43$ trong hệ thập phân.

Bảng sau chỉ ra tương quan giữa số thập phân và số nhị phân:

Hệ thập phân	Hệ nhị phân
0	0
1	1
2	10
3	11
4	100
5	101
6	110

7	111
8	1000
9	1001
...	...

Mỗi số 0 và 1 được gọi là một bit (viết tắt của binary digit); một số ở hệ 2 gồm các bít được kết thúc bởi chữ B để phân biệt với các hệ khác. Một cụm 4 bít gọi là 1 nibble, một cụm 8 bít gọi là byte, cụm 16 bit gọi là 1 word, cụm 32 bít gọi là double word...

Bít đầu tiên bên trái được gọi là bít có ý nghĩa lớn nhất (MSB: most significant bit), còn bít cuối cùng ở bên phải được gọi là bít có ý nghĩa nhỏ nhất (LSB: least significant bit).

2. Chuyển đổi giữa các hệ đếm

Vì con người quen sử dụng các số ở hệ thập phân trong khi đó các bộ vi xử lý chỉ thao tác với các số ở hệ nhị phân nên để đảm bảo việc giao tiếp giữa người và máy thì phải có phép chuyển đổi giữa hai hệ này.

2.1. Chuyển từ hệ hai sang hệ mười

Để đổi một số từ hệ hai sang hệ mười ta áp dụng công thức sau:

$$(b_n b_{n-1} \ldots . b_1 b_0)\mathbf{B} = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_1 \times 2^1 + b_0 \times 2^0$$

Ví dụ:

$100011\mathbf{B} = 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 35$

2.2. Chuyển từ hệ mười sang hệ hai

Để chuyển từ hệ mười sang hệ hai ta sử dụng phương pháp đơn giản sau: lấy số cần chuyển chia cho 2 và ghi nhớ phần dư, tiếp theo lấy thương của phép chia trước đó chia cho 2 và ghi nhớ phần dư ... cứ tiếp tục cho đến khi thương bằng 0. Kết quả của phép chuyển đổi chính là dãy các số dư lấy theo thứ tự đảo ngược.

Ví dụ 1: Đổi số 25 sang hệ 2.

$25 \mid \boldsymbol{1}$ 25 chia 2 được 12 dư 1

12	***0***	12 chia 2 được 6 dư 0
6	***0***	6 chia 2 được 3 dư 0
3	***1***	3 chia 2 được 1 dư 1
1	***1***	1 chia 2 được 0 dư 1
0		

Vậy kết quả là ***11001***.

Ví dụ 2: Đổi số 42 sang hệ 2.

42	***0***	42 chia 2 được 21 dư 0
21	***1***	21 chia 2 được 10 dư 1
10	***0***	10 chia 2 được 5 dư 0
5	***1***	5 chia 2 được 2 dư 1
2	***0***	2 chia 2 được 1 dư 0
1	***1***	1 chia 2 được 0 dư 1
0		

Vậy kết quả là ***101010***.

Ta thấy rằng một số biểu diễn ở hệ hai thì rất dài và khó nhớ nên trong thực tế người ta thường sử dụng hệ mười sáu. Hệ mười sáu là hệ đếm cơ số 16 nên người ta sử dụng các số từ 0 đến 9 và các chữ cái từ A đến F để biểu diễn các số. Bảng sau chỉ ra tương quan giữa số hệ mười và hệ mười sáu:

Hệ 10	Hệ 16
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8

9	9
10	A
11	B
12	C
13	D
14	E
15	F
16	10
17	11
.....	.....

3. Các phép toán số học với số hệ hai

3.1. Phép cộng

Phép cộng với các số hệ hai được thực hiện tương tự như với các số hệ mười theo quy tắc sau:

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 nhớ 1

Ví dụ:

0010	100011
+ 1011	+ 011111
= 1101	= 1000010

3.2. Phép trừ

Phép trừ với các số hệ hai được thực hiện tương tự như với các số hệ mười.

Ví dụ:

1111	1000010
- 1011	- 01010
= 0100	= 111000

Trong máy tính người ta thực hiện phép trừ bằng phép cộng với số bù hai.

3.3. Phép nhân

Phép nhân hai số hệ hai được thực hiện tương tự như nhân hai số hệ mười. Quy tắc thực hiện như sau:

0 x 0 = 0

0 x 1 = 0

1 x 0 = 0

1 x 1 = 1

Ví dụ:

```
      1001
    x 1101
  ---------
      1001
+   0000
    1001
   1001
---------
= 1110101
```

4. Các mã thường dùng

4.1. Mã BCD

Mã BCD thường được sử dụng để mã hoá các chữ số từ 0 đến 9 trong vi xử lý, PLC...

Chữ số của hệ 10	Mã BCD
0	0000
1	0001

2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Ví dụ: Mã BCD của 5 là 0101, của 9 là 1001

Trong khi làm toán với các số BCD ta thường kết hợp 2 số BCD tạo thành 1 byte, dạng biểu diễn này gọi là dạng BCD chuẩn hay BCD đóng gói (packed BCD). Ví dụ mã BCD đóng gói (BCD chuẩn) của 59 là 0101 1001B, của 62 là 0110 0010B.

Số BCD không gói là số dài 1 byte trong đó 4 bit cao bằng 0 còn 4 bit thấp là số BCD chuẩn mã hóa số cần biểu diễn. Ví dụ mã BCD không gói của 5 là 0000 0101B, của 4 là 0000 0100B.

4.2. Mã ASCII

Mã ASCII (American Standard Code for International Interchange – Mã chuẩn của Mỹ dùng cho trao đổi thông tin) là bộ mã rất thông dụng trong máy tính và truyền thông.

Trong bảng mã ASCII tiêu chuẩn người ta sử dụng 7 bit để mã hóa các ký tự thông dụng, như vậy bảng này sẽ có 128 ký tự ứng với các mã số từ 0 đến 127.

Trong bảng mã ASCII mở rộng, người ta bổ xung thêm 128 ký tự đặc biệt với mã từ 128 đến 255.

Ví dụ mã ASCII của 'A' là 65 (01000001), của 'a' là 97 (01100001)

Chương 2

GIỚI THIỆU VỀ MÁY TÍNH VÀ BỘ VI XỬ LÝ

1. Sơ lược về lịch sử phát triển của máy tính và bộ vi xử lý

** Thế hệ máy tính cơ khí:*

Ý tưởng về một hệ thống tính toán đã có từ rất lâu, khoảng 500 năm trước công nguyên, người Babylon

đã chế tạo được mẩy tính đầu tiên có tên là Abacus. Năm 1643 Blaise Pascal chế tạo thành công một máy tính tạo bởi các bánh răng trong đó số răng của bánh nọ gấp 10 lần số răng của bánh kia, nguyên lý này sau này được sử dụng để tạo các đồng hồ đo quãng đường của motor, đồng hồ đo nước...

** Thế hệ máy tính cơ điện-điện tử:*

Năm 1889, Herman Holerith phát minh ra card đục lỗ dùng để lưu trữ dữ liệu, sau đó ông chế tạo thành công máy tính cơ khí được điều khiển bởi một môtơ điện, nó có thể thực hiện được các phép đếm, sắp xếp và so sánh thông tin lưu trong card đục lỗ.

Năm 1942, nhà phát minh người Đức Konrad Zure chế tạo ra máy tính điện tử Z3 dùng cho không quân Đức. Năm 1943, Alan Turing phát minh ra hệ thống máy tính điện tử có tên là Collossus được thiết kế từ các đèn điện tử chân không, đây là một máy tính chuyên dụng thực hiện theo một chương trình cố định để giải mã các bí mật quân sự của Đức quốc xã.

Máy tính điện tử đa dụng đầu tiên – hệ máy tính có thể lập trình được - được phát triển bởi Đại học Pennnsylvania có tên là ENIAC (Electronics Numerical Integrator and Calculator). Đây là một máy tính lớn chứa hơn 17.000 đèn điện tử, nặng khoảng 30 tấn và có thể thực hiện được 100.000 thao tác trong 1 giây. ENIAC được lập trình bằng cách nối lại mạch điện nhờ các rơle điện từ.

** Thế hệ máy tính dùng vi xử lý:*

Năm 1948 Transistor được phát minh, đến năm 1958 Jack Kilby phát minh ra mạch tổ hợp nó là cơ sở để phát triển các mạch số tích hợp. Cùng thời gian này Marcian T.Hoff một kỹ sư của Intel đã thiết kế ra bộ vi xử lý 4004 mở đầu cho thời kỳ sử dụng vi xử lý trong máy tính.

4004 là bộ vi xử lý 4 bit có thể quản lý được bộ nhớ có 4096 ô nhớ 4 bit, tập lệnh của 4004 gồm 45 lệnh khác nhau, nó được chế tạo theo công nghệ MOSFET kênh p cho phép thực hiện được 50 KIPS (KIPS: kilo-instruction per second – nghìn lệnh/giây). 4004 được dùng để thiết kế các hệ thống video game, hệ thống điều khiển nhỏ dùng vi xử lý. Trên cơ sở 4004 hãng Intel sản xuất bộ vi xử lý 4040, đây cũng là bộ vi xử lý 4 bit nhưng có tốc độ cao hơn 4004.

Sau năm 1971, Intel sản xuất bộ vi xử lý 8 bit tên là 8008, nó có thể quản lý được 16 KB bộ nhớ, tập lệnh gồm 48 lệnh khác nhau. Năm 1973 hãng Intel giới thiệu bộ vi xử lý 8 bit hiện đại đầu tiên có tên là 8080, nó có thể thực hiện được 500 KIPS, quản lý được 64 KB bộ nhớ. Năm 1977 Intel phát triển bộ vi xử lý 8085 (tương thích với Z80 của hãng Zilog) là bộ vi xử lý 8 bit đa dụng nhanh hơn 8080.

Năm 1978 Intel giới thiệu bộ vi xử lý 16 bit tên là 8088 có thể thực hiện được 2,5 MIPS (MIPS: millions-instruction per second – triệu lệnh/giây) và có thể quản lý được 1 MB bộ nhớ. Bộ vi xử lý 80286 cũng là bộ vi xử lý 16 bit nhưng nó có thể quản ý được tới 16 MB bộ nhớ.

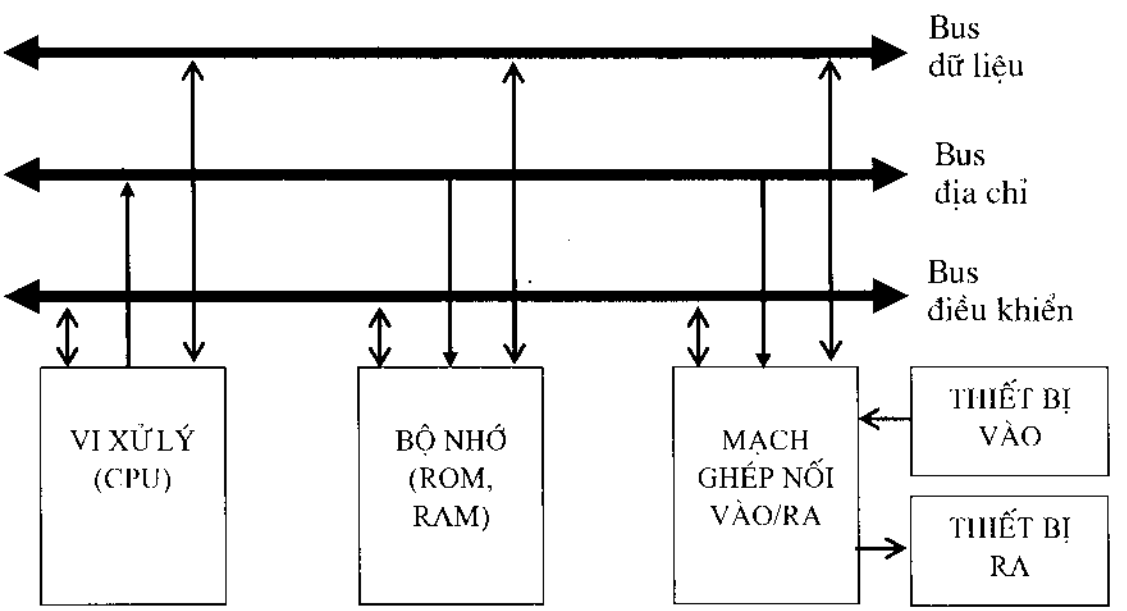
Các ứng dụng đòi hỏi tốc độ ngày càng cao, bộ nhớ và độ rộng bus dữ liệu (số chân của vi xử lý để truyền/nhận dữ liệu) ngày càng lớn. Năm 1986 Intel giới thiệu bộ vi xử lý 32 bit có tên là 80386 có thể quản lý được 4 GB bộ nhớ. Năm 1989 Intel giới thiệu bộ vi xử lý 80486 có thể thực hiện được 50 MIPS. Năm 1993 Intel giới thiệu bộ vi xử lý 80585 (còn gọi là Pentium hay P5) có thể thực hiện được 110 MIPS, nó có thể thực hiện được đồng thời một

lúc 2 lệnh (hai lệnh độc lập nhau) vì bên trong có tới 2 bộ xử lý số nguyên (công nghệ superscalar). Các bộ vi xử lý sau này của Intel sản xuất theo kiến trúc IE-64 (kiến trúc 64 bít của Intel). Đầu năm 2003 Intel đã xuất xưởng các bộ vi xử lý Pentium IV có tốc độ 3,06 MHZ với kiến trúc đa luồng để nâng cao hiệu suất sử dụng các khối chức năng bên trong vi xử lý, với công nghệ đa luồng một bộ vi xử lý vật lý tương đương như hai bộ vi xử lý logic.

2. Sơ lược cấu trúc của máy tính dùng vi xử lý (hệ vi xử lý)

Vi xử lý là một thành phần cơ bản không thể thiếu của máy tính, ngoài ra để tạo ra một hệ hoàn chỉnh cần phải có các bộ phận khác như bộ nhớ, các thiết bị vào/ra như bàn phím, màn hình...

Hình 2.1 giới thiệu sơ đồ khối tổng quát của một máy tính:



Hình 2.1. Sơ đồ khối cấu trúc của máy tính

Trong sơ đồ này ta thấy các khối chức năng của một máy tính (hay hệ vi xử lý) bao gồm:

+ Bộ vi xử lý (CPU)

+ Bộ nhớ bán dẫn (ROM, RAM)

+ Mạch ghép nối vào ra

+ Bus hệ thống để truyền thông tin bao gồm bus điều khiển, bus địa chỉ và bus dữ liệu.

Sau đây chúng ta sẽ tìm hiểu nguyên lý làm việc của máy tính bằng cách xem xét chức năng của các khối đó.

2.1.Bộ vi xử lý

Bộ vi xử lý (Microprocessor) hay còn được gọi là CPU (Central Processing Unit - đơn vị xử lý trung tâm) đóng vai trò là bộ não của máy tính. Đây là một vi mạch số với mức độ tích hợp cực lớn (VLSI) bên trong nó bao gồm nhiều khối chức năng khác nhau như đơn vị số nguyên để thao tác tính toán với các số nguyên, đơn vị xử lý dấu phẩy động để thực hiện các phép tính với số thực... Khi hoạt động, nó đọc mã lệnh (mã lệnh được ghi dưới dạng chuỗi các bít 0, 1) từ bộ nhớ, đưa vào trong vi xử lý để giải mã thành các vi lệnh là những xung điều khiển để điều khiển hoạt động của các đơn vị chức năng bên trong vi xử lý.

Các thông số quan trọng của một bộ vi xử lý gồm:

+ Tần số làm việc: là tần số xung nhịp (clock) cung cấp cho vi xử lý, tần số này quyết định đến tốc độ làm việc của vi xử lý.

+ Độ rộng bus dữ liệu m: là số đường dây dùng để truyền dữ liệu kí hiệu từ D_0 đến D_{m-1}. Các giá trị của m thường là 4, 8, 16, 32 và 64.

+ Độ rộng bus địa chỉ n: quyết định đến dung lượng bộ nhớ cực đại mà vi xử lý có thể quản lý được. Một bộ vi xử lý có n đường địa chỉ từ A_0 đến A_{n-1} có thể quản lý được 2^n ô nhớ (mỗi ô nhớ thường là 1 byte). Các giá trị của n thường là 16, 20, và 32.

2.2.Bộ nhớ

Bộ nhớ (hay còn gọi là bộ nhớ trong, bộ nhớ chính) được tạo từ các vi mạch nhớ ROM và RAM.

ROM (Read Only Memory – bộ nhớ chỉ đọc, nội dung bên trong của ROM không bị mất khi mất nguồn nuôi) dùng để chứa các chương trình điều khiển hệ thống như chương trình để kiểm tra các thiết bị mỗi khi bật nguồn, chương trình khởi động máy, các chương trình điều khiển trao đổi tin với các thiết bị ngoại vi như bàn phím, màn hình, ...

RAM (Random Access Memory – bộ nhớ truy cập ngẫu nhiên) là bộ nhớ có thể ghi/đọc được, có nghĩa là ta có thể đọc thông tin từ bộ nhớ, xoá thông tin cũ trong bộ nhớ hoặc ghi thông tin mới vào bộ nhớ. Nội dung thông tin ghi trong bộ nhớ RAM sẽ bị mất khi mất nguồn cung cấp. RAM được dùng để lưu trữ mã lệnh, toán hạng và kết quả của chương trình khi nó đang được thực hiện. Trong máy tính bộ nhớ RAM là các module dạng thanh cắm trên bảng mạch chính của máy, trên mỗi module thường gắn nhiều vi mạch RAM, các vi mạch này thường có dung lượng 1, 2, 4, 8, 16, 32, hoặc 64 MB.

2.3.Mạch ghép nối vào/ra

Mạch ghép nối vào ra có nhiệm vụ tạo ra khả năng giao tiếp giữa hệ vi xử lý với thế giới bên ngoài. Các thiết bị vào/ra (hay còn gọi là thiết bị ngoại vi) bao gồm các thiết bị vào (bàn phím, chuột, máy quét...), thiết bị ra (màn hình, máy in, máy vẽ...). Ngoài ra còn có các thiết bị lưu trữ (bộ nhớ ngoài) dùng để lưu thông tin với khối lượng lớn như đĩa mềm, đĩa cứng, đĩa CD...

2.4.Bus hệ thống

Bus hệ thống là tập hợp tất cả các đường dây dùng để liên lạc giữa các khối chức năng trong hệ thống. Dựa vào chức năng của các đường dây người ta chia chúng làm 3 nhóm:

- Bus điều khiển là các đường dây mang các tín hiệu điều khiển hoạt động của các khối như RD (Read - đọc bộ nhớ hoặc thiết bị vào), WR (Write - ghi dữ liệu vào bộ nhớ hoặc xuất dữ liệu ra thiết bị ra)...

- Bus dữ liệu là các đường dây mang số liệu mà vi xử lý đang trao đổi với bộ nhớ hoặc thiết bị vào/ra.

- Bus địa chỉ mang thông tin về địa chỉ của ô nhớ hay một thiết bị vào/ra mà vi xử lý đang trao đổi tin. Thông tin về địa chỉ là do vi xử lý phát ra để chọn ra một ô nhớ hoặc một thiết bị vào/ra mà nó cần trao đổi tin.

3. Khái niệm về phần cứng, phần mềm

3.1.Phần cứng

Phần cứng (Hardware) là thuật ngữ dùng để chỉ toàn bộ những thiết bị cơ khí, điện tử tạo nên máy tính như các ổ đĩa, màn hình...

3.2.Phần mềm

Phần mềm (Software) là thuật ngữ dùng để chỉ các chương trình máy tính, nó được thực thi trên phần cứng bằng cách điều khiển sự hoạt động của phần cứng. Người ta còn sử dụng khái niệm phần dẻo (Firmware) để chỉ những chương trình tồn tại gắn chặt với phần cứng như các chương trình lưu trong bộ nhớ ROM...

Phần mềm được chia thành các loại sau:

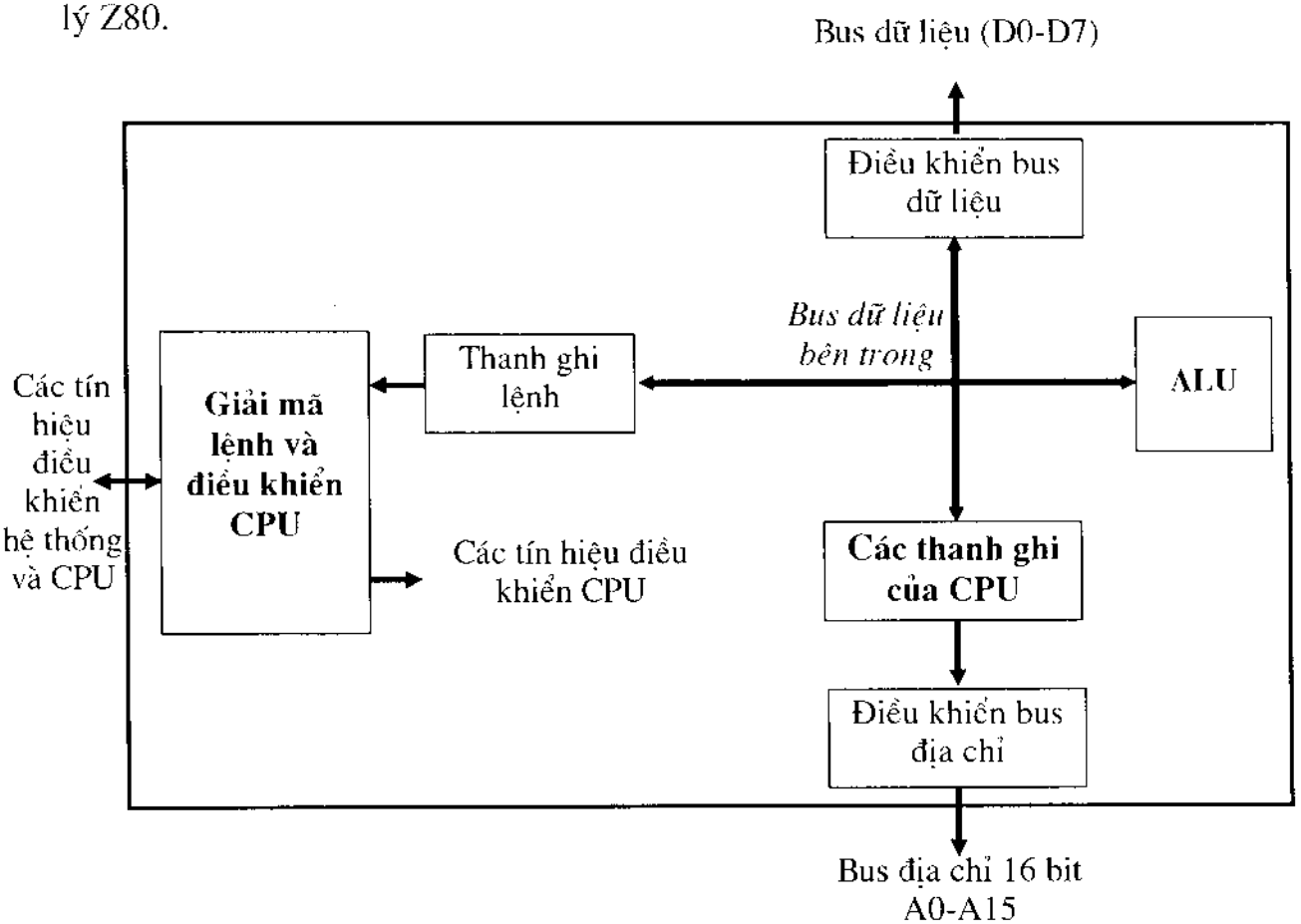
- Hệ điều hành (Operating system) như DOS, Windows, Linux ...
- Trình tiện ích như NC, NU ...
- Chương trình ứng dụng như MS Word, Multisim ...
- Ngôn ngữ lập trình.

Chương 3

BỘ VI XỬ LÝ Z80 CỦA ZILOG

1. Kiến trúc của Z80 CPU

Vi xử lý Z80 của hãng Zilog là bộ vi xử lý 8 bit sử dụng nguồn nuôi 5V, có tần số hoạt động nằm trong khoảng 6 đến 20 MHz nên rất phù hợp với các hệ thống nhỏ. Các thanh ghi bên trong của Z80 gồm 208 bít có thể đọc/ghi bởi người lập trình. Các thanh ghi này được chia làm 2 tập mỗi tập gồm 6 thanh ghi đa dụng, chúng có thể làm việc độc lập như những thanh ghi 8 bit hoặc làm việc theo một cặp tạo thành thanh ghi 16 bit. Ngoài ra còn có hai tập các thanh ghi chứa và thanh ghi cờ, một thanh ghi con trỏ ngăn xếp, một thanh ghi đếm chương trình, hai thanh ghi chỉ số, một thanh ghi REFRESH và một thanh ghi ngắt INTERRUPT. Hình 3.1 minh họa sơ đồ khối bên trong vi xử lý Z80.



Hình 3.1. Sơ đồ khối bên trong Z80

Sau đây chúng ta sẽ xem xét chức năng của các khối bên trong CPU Z80.

1.1. Các thanh ghi của CPU

Các thanh ghi của Z80 CPU là một bộ nhớ gồm 208 bit đọc/ghi. Bộ nhớ này có thể được truy xuất bởi người lập trình. Hình 3.2 cho thấy bộ nhớ này gồm 8 thanh ghi 8 bit và 4 thanh ghi 16 bit. Tất cả các thanh ghi của Z80 CPU đều dùng RAM tĩnh. Các thanh ghi bao gồm 2 tập 6 thanh ghi đa dụng, tập các thanh ghi này có thể được sử dụng độc lập như là các thanh ghi 8 bit hoặc từng cặp như là các thanh ghi 16 bit. Có hai tập (chính và phụ) thanh ghi tích lũy, thanh ghi cờ và sáu thanh ghi đặc biệt.

Tập thanh ghi chính		Tập thanh ghi phụ	
A	F	A'	F'
B	C	B'	C'
D	E	D'	E
H	L	H'	L'

Thanh ghi vector ngắt I	Các thanh ghi đặc biệt
Thanh ghi làm tươi bộ nhớ R	
Thanh ghi chỉ số IX	
Thanh ghi chỉ số IY	
Thanh ghi con trỏ ngăn xếp SP	
Thanh ghi đếm chương trình PC	

Hình 3.2. Các thanh ghi bên trong CPU Z80

* *Tập các thanh ghi đặc biệt.*

- Thanh ghi đếm chương trình PC (Program counter): giữ địa chỉ mười sáu bit của lệnh hiện tại đang được lấy về từ bộ nhớ. PC tự động tăng sau khi nội dung của nó được đặt lên bus địa chỉ. Khi có 1 lệnh nhảy xảy ra, 1 giá trị mới được tự động đặt vào trong PC thay cho giá trị sẽ được tăng của nó.

- Thanh ghi con trỏ ngăn xếp SP (Stack pointer): Giữ địa chỉ 16 bit của đỉnh ngăn xếp hiện hành trong bộ nhớ RAM ngoài. Bộ nhớ ngăn xếp bên ngoài được tổ chức như là 1 file vào sau ra trước (LIFO). Dữ liệu có thể được đẩy vào ngăn xếp từ các thanh ghi xác định của CPU hay được lấy ra khỏi ngăn xếp để đưa vào các thanh ghi xác định của CPU qua việc thi hành các lệnh PUSH và POP. Dữ liệu được lấy ra từ ngăn xếp luôn luôn là dữ liệu được đẩy vào ngăn xếp sau cùng. Ngăn xếp cho phép đơn giản hóa việc thi hành các loại ngắt không giới hạn các chương trình con lồng nhau và đơn giản hóa nhiều kiểu thao tác trên dữ liệu.

- Hai thanh ghi chỉ số IX và IY: Hai thanh ghi chỉ số độc lập giữ địa chỉ cơ bản 16 bit được dùng trong chế độ định địa chỉ chỉ số. Trong chế độ này, một thanh ghi chỉ số được dùng như là một vị trí cơ bản chỉ đến một vùng trong bộ nhớ nơi mà dữ liệu bắt đầu được chứa hay được khôi phục. Byte được thêm trong câu lệnh chỉ số để xác định khoảng cách đến vị trí cơ bản. Khoảng cách này được xác định bằng phép bù 2 một số nguyên có dấu. Chế độ này làm đơn giản đi rất nhiều cho chương trình, đặc biệt cho sử dụng các bảng dữ liệu.

- Thanh ghi vector ngắt I (Interrupt): Z80 CPU có thể được điều khiển trong 1 chế độ mà ở đó một lệnh gọi trực tiếp đến bất kỳ vùng nào của bộ nhớ có thể được thực hiện bằng cách đáp ứng ngắt. Thanh ghi I được dùng cho mục đích này để chứa 8 bit cao của địa chỉ trực tiếp trong khi thiết bị ngắt cung cấp 8 bit địa chỉ thấp. Đặc điểm này cho phép chương trình phục vụ ngắt định vị nhanh tới bất kỳ vùng nhớ nào với thời gian truy xuất đến chương trình là nhỏ nhất.

- Thanh ghi làm tươi bộ nhớ R: Z80 CPU chứa 1 bộ đếm làm tươi bộ nhớ để cho phép bộ nhớ động (DRAM) được dùng dễ dàng như bộ nhớ tĩnh (SRAM). 7 trong 8 bit của thanh ghi này được tự động tăng sau mỗi lần lấy lệnh. Bit thứ 8 còn lại được lập trình như là kết quả của lệnh LD R,A. Dữ liệu trong bộ đếm làm tươi được gửi lên phần thấp của bus địa

chỉ kèm theo một tín hiệu làm tươi trong khi CPU giải mã và thực thi lệnh vừa được lấy về. Cách thức làm tươi này hoàn toàn dễ hiểu đối với người lập trình và không làm chậm hoạt động của CPU. Người lập trình có thể lấy thanh ghi R cho mục đích kiểm tra, nhưng thanh ghi này thường không được người lập trình dùng. Trong thời gian làm tươi, nội dung của thanh ghi I được đặt lên 8 bit cao của bus địa chỉ.

- Các thanh ghi tích lũy A và thanh ghi cờ F: CPU bao gồm hai thanh ghi tích lũy độc lập A và A' và được kết hợp với các thanh ghi cờ 8 bit F và F'. Thanh ghi tích lũy A và A' lưu giữ kết quả của phép toán logic và số học, thanh ghi cờ (F và F') cung cấp thông tin cho người dùng về trạng thái Z80 ở bất kỳ thời điểm nào. Vị trí bit cho mỗi cờ được chỉ ra như sau:

S	Z	X	H	X	P/V	N	C

Trong đó :

C: Cờ Carry

N: Cộng hay trừ

P/V: Cờ Parity hay tràn

H: Cờ nửa Carry

Z: Cờ zero

S: Cờ dấu

X: Không dùng

Một trong hai thanh ghi cờ Z80 bao gồm 6 bit mang thông tin về các trạng thái được Set hay Reset bởi các phép toán (bit 3 và bit 5 không được dùng), 4 bit có thể kiểm tra (C, P/V, Z và S) dùng làm điều kiện cho các lệnh Jump, Call hay Return. Hai cờ không kiểm tra được là H và N được dùng cho các phép toán số học BCD.

+ Cờ Carry (C): Bit cờ Carry được set hay reset phụ thuộc vào phép toán được thi hành. Lệnh 'ADD' và lệnh "SUB" có thể set cờ Carry.

Trong các lệnh RLA, RRA, RLS, và RRS bit Carry được dùng như là mối liên kết giữa LSB và MSB cho các thanh ghi hay bộ nhớ bất kỳ. Trong các lệnh RLCA, RLC và SLA cờ Carry chứa giá trị cuối cùng được dịch ra từ bit 7 của thanh ghi hay bộ nhớ. Trong các lệnh RRCA, RRC, SRA, SRL Carry chứa giá trị cuối cùng được dịch ra từ bit 0 của thanh ghi hay bộ nhớ. Trong các lệnh logic AND, OR, XOR Carry sẽ bị reset.

Cờ Carry cũng có thể được set (SFC) và bù (CCF).

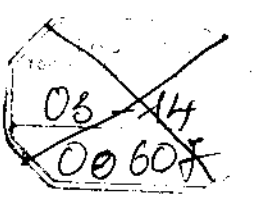
+ Cờ cộng/trừ (N): Cờ này được dùng khi hiệu chỉnh thập phân thanh ghi tích lũy (DAA) để phân biệt lệnh ADDER và SUBTRACT. Khi dùng lệnh ADD, N sẽ được đặt ở mức logic 0. Khi dùng lệnh SUBTRACT, N sẽ được đặt ở mức logic 1.

+ Cờ chẵn lẻ/tràn (P/V): Cờ này được set phụ thuộc vào phép toán được thi hành.

Với các phép toán số học, cờ này báo tràn khi kết quả trong thanh ghi tích lũy lớn hơn +127 hay nhỏ hơn -128. Điều kiện tràn có thể được xác định nhờ vào các bit dấu của các toán hạng.

Thêm vào đó các phép toán khác dấu sẽ không gây tràn. Khi cộng các toán hạng cùng dấu và kết quả là dấu khác, cờ tràn được set.

Ví dụ :



+120 = 0111 1000 ADDEND

+105 = 0110 1001 AUGEND

+225 = 1110 0001 (-95) SUM

Hai số cộng lại và kết quả vượt quá +127 và 2 toán hạng dương có kết quả là số âm (-95), đây là kết quả sai. Cờ tràn được set.

Đối với phép trừ, tràn có thể xảy ra khi các toán hạng khác dấu. Các toán hạng cùng dấu sẽ không bao giờ gây tràn.

Ví dụ :

	+127	0111 1111	MINUEND
(-)	-64	1100 0000	SUBTRAHEND
	+191	1011 1111	DIFFERENCE

+ Cờ bán Carry (H): Cờ bán Carry (H) sẽ được set hay reset phụ thuộc vào Carry và borrow giữa bit 3 và 4 của phép toán số học 8 bit. Cờ này được dùng khi hiệu chỉnh thập phân thanh ghi tích lũy (DAA) để làm đúng kết quả của phép toán cộng hay trừ số BCD. Cờ H sẽ là set(1) hay reset (0) theo bảng sau:

H	Cộng	Trừ
1	Carry từ bit 3 sang bit 4	Mượn từ bit 4
0	Không có Carry từ bit 3 sang bit 4	Không mượn từ bit 4

+ Cờ zero (Z): Cờ zero được set hay reset nếu kết quả của lệnh là 0.

Đối với các phép toán logic và số học 8 bit, cờ Z sẽ được set lên 1 nếu nội dung thanh ghi tích lũy là 0.

Đối với lệnh so sánh, cờ Z sẽ được set lên 1 nếu nội dung của thanh ghi tích lũy và nội dung của bộ nhớ được chỉ bởi cặp thanh ghi HL giống nhau.

Khi kiểm tra bit trong thanh ghi hay bộ nhớ, cờ Z sẽ chứa trạng thái bù của bit được chỉ định (Xem bit b,s).

Khi đang nhập hay xuất 1 byte giữa bộ nhớ với thiết bị I/O (INI, IND, OUTI, OUTD) nếu kết quả của B-1 là 0 thì cờ Z được set. Đối với byte

được nhập từ thiết bị I/O dùng IN r, (C), cờ Z sẽ được set để cho biết byte nhập bằng zero.

+ Cờ dấu (S): Cờ dấu chứa trạng thái của bit có trọng số lớn nhất của thanh ghi tích lũy (bit 7). Một số dương được phân biệt bởi '0' ở bit 7 và số âm là '1'.

Khi nhập byte từ thiết bị I/O vào thanh ghi, IN r, (C) cờ S sẽ chỉ hoặc số dương (S=0) hoặc số âm (S=1).

* *Các thanh ghi đa dụng:*

Có 2 tập đối xứng các thanh ghi đa dụng. Mỗi tập chứa 6 thanh ghi 8 bit hay như cặp thanh ghi 16 bit. Tập thứ nhất được gọi là BC, DE và HL trong khi tập thứ hai tương ứng được gọi là BC', DE' và HL'. Trong bất kỳ thời điểm nào người lập trình có thể chọn 1 trong các tập thanh ghi để làm việc qua 1 lệnh hoán chuyển đơn. Trong các hệ thống yêu cầu đáp ứng ngắt nhanh, một tập của tập thanh ghi đa dụng và một thanh ghi cờ/tích lũy có thể được dự trữ để giải quyết chương trình rất nhanh này. Chỉ cần một lệnh đơn giản được thực thi để chạy giữa các chương trình. Điều này thu giảm thời gian phục vụ ngắt do việc loại bỏ yêu cầu cất và khôi phục nội dung thanh ghi ở ngăn xếp trong thời gian ngắt hay xử lý chương trình con. Các thanh ghi đa dụng này được dùng cho nhiều ứng dụng bởi người lập trình.

1.2. Đơn vị logic số học ALU

Các lệnh logic và số học 8 bit của CPU được thực hiện trong ALU. Giao tiếp giữa ALU với các thanh ghi và bus dữ liệu bên ngoài được thực hiện trên bus dữ liệu bên trong. Các chức năng ALU gồm:

- Cộng
- Trừ
- Logic AND

- Logic OR
- Logic XOR
- So sánh
- Dịch trái, dịch phải hay quay (Số học và logic)
- Phép toán tăng
- Phép toán giảm
- Set bit
- Reset bit
- Test bit

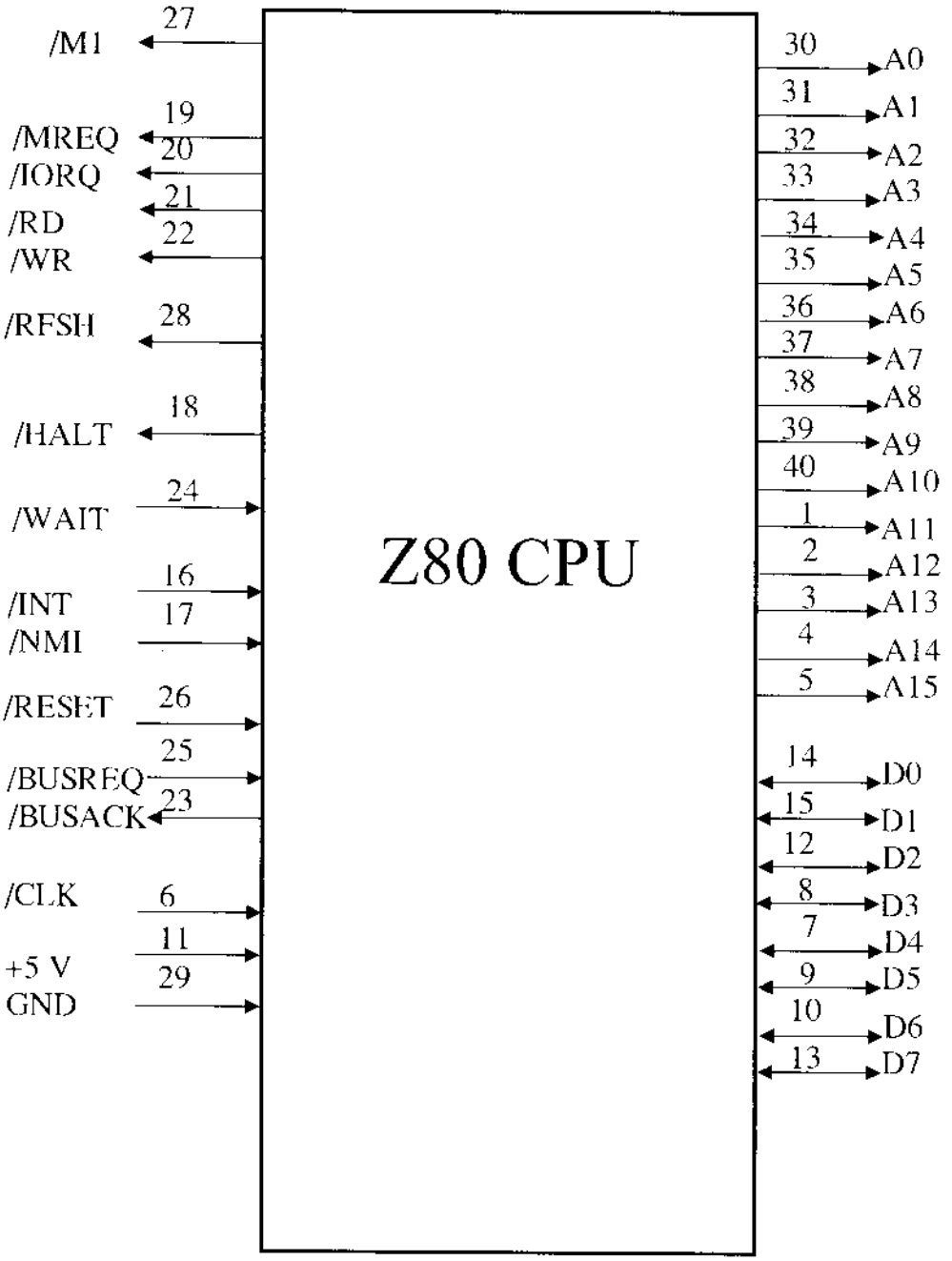
1.3. Điều khiển CPU và thanh ghi lệnh

Mỗi 1 lệnh được lấy về từ bộ nhớ, nó được đặt vào trong thanh ghi lệnh và được giải mã. Các phần điều khiển thi hành chức năng này, phát và cấp các tín hiệu điều khiển cần thiết để đọc hoặc ghi dữ liệu từ (hoặc đến) các thanh ghi, điều khiển ALU và đáp ứng các yêu cầu của tín hiệu điều khiển bên ngoài.

2. Sơ đồ chân và chức năng các chân của Z80

2.1. Sơ đồ chân của Z80

Sơ đồ các chân của Z80 CPU được mô tả trong hình 3.3 ở trang sau:



Hình 3.3. Sơ đồ chân của Z80

2.2. Chức năng các chân

- A15 ÷A0: bus địa chỉ (đầu ra 3 trạng thái), A15 ÷ A0 tạo thành bus địa chỉ 16 bit. Bus địa chỉ cung cấp địa chỉ cho bộ nhớ và cho thiết bị I/O để trao đổi dữ liệu.

- /BUSACK (bus acknowledge, đầu ra tích cực thấp): báo cho thiết bị yêu cầu biết bus địa chỉ, bus dữ liệu và các đường tín hiệu điều khiển: /MREQ, /IORQ, /RD, /WR đã ở trạng thái trở kháng cao. Mạch ngoài có thể dùng những đường này.

- /BUSREQ: Bus request (đầu vào, tích cực thấp) bus request có độ ưu tiên cao hơn /NMI và nó luôn được nhận diện vào cuối chu kỳ máy hiện tại. /Busreq dùng để yêu cầu bus địa chỉ, bus dữ liệu và các tín hiệu điều khiển: /MREQ, /IORQ, /RD, /WR phải ở trạng thái trở kháng cao để các thiết bị khác có thể điều khiển các đường này. Bus request thường được nối vào cổng OR và yêu cầu điện trở kéo lên bên ngoài cho các ứng dụng kiểu này.

- D7 ÷ D0: Data bus (vào/ra, 3 trạng thái) D7 ÷ D0 tạo thành bus dữ liệu 8 bit hai chiều, sử dụng để trao đổi dữ liệu với bộ nhớ và I/O.

- /HALT: Halt (đầu ra, tích cực thấp) báo rằng CPU đang thực thi 1 lệnh Halt và sẽ chờ 1 trong hai tín hiệu: /NMI hoặc /INT trước khi sự điều khiển được khôi phục. Trong thời gian Halt, CPU thực thi các lệnh NOP để duy trì việc làm tươi bộ nhớ.

- /INT: Interrup request (đầu vào, tích cực thấp) – yêu cầu ngắt. Interrup request được thiết bị I/O phát ra. CPU chấp nhận yêu cầu ngắt tại thời điểm cuối của lệnh hiện tại nếu được phần mềm cho phép. /INT thường được nối vào các cổng OR và cần một điện trở kéo lên cho các ứng dụng loại này.

- /IORQ: Input/ Output Request (đầu ra 3 trạng thái, tích cực thấp) báo nửa thấp của bus địa chỉ giữ 1 địa chỉ I/O hợp lệ cho hoạt động đọc hay ghi I/O. /IORQ cũng được phát đồng thời với /M1 trong 1 chu kỳ công nhận ngắt để báo rằng 1 vertor đáp ứng ngắt có thể được đặt lên data bus.

- /M1: machine cycle one (đầu ra, tích cực thấp) /M1 cùng với /MREQ báo rằng chu kỳ máy hiện tại là chu kỳ lấy mã lệnh. /M1 cùng với /IORQ báo cho biết rằng đang ở chu kỳ công nhận ngắt.

- /MREQ - memory request – yêu cầu bộ nhớ (đầu ra 3 trạng thái, tích cực thấp) /MREQ báo rằng bus địa chỉ đang giữ một địa chỉ hợp lệ cho việc đọc ghi bộ nhớ.

- /NMI: Non Maskable Interrup – ngắt không che được (đầu vào, tích cực cạnh xuống) /NMI có độ ưu tiên cao hơn /INT. /NMI luôn luôn được chấp nhận tại chu kỳ cuối của lệnh hiện hành, độc lập với trạng thái của Flip-Flop interrupt và tự động đưa CPU đến vị trí 0066h.

- /RD: Read (đầu ra 3 trạng thái, tích cực thấp) báo rằng CPU muốn đọc dữ liệu từ bộ nhớ hay thiết bị I/O. Thiết bị I/O hay bộ nhớ được chỉ định sẽ dùng tín hiệu này để đưa dữ liệu lên bus dữ liệu.

- /RESET: Reset (đầu vào, tích cực thấp) /RESET khởi động CPU như sau: reset IFF1 & IFF2 xóa PC và các thanh ghi I & R, đặt trạng thái interrupt đến chế độ 0. Trong thời gian reset, bus địa chỉ và bus dữ liệu sẽ ở trạng thái trở kháng cao và tất cả các đầu ra điều khiển sẽ ở trạng thái không tích cực. Chú ý rằng đường /RESET phải tích cực thấp ít nhất là 3 chu kỳ xung clock trước khi hoạt động Reset hoàn tất.

- /RFSH: Refresh (đầu ra, tích cực thấp) /RFSH cùng với /MREQ báo rằng 7 bit thấp của các đường địa chỉ có thể được dùng làm địa chỉ làm tươi cho bộ nhớ DRAM.

- /WAIT: wait (đầu ra, tích cực thấp) /WAIT báo cho CPU biết bộ nhớ hay thiết bị I/O được chỉ định chưa sẵn sàng cho việc truyền dữ liệu. CPU tiếp tục ở trạng thái chờ. Việc kéo dài trạng thái /WAIT có thể ngưng việc làm tươi cho bộ nhớ RAM động.

- /WR: Write (đầu ra 3 trạng thái, tích cực thấp) /WR báo bus dữ liệu đang giữ 1 giá trị dữ liệu hợp lệ để đưa vào bộ nhớ hoặc I/O được chỉ định.

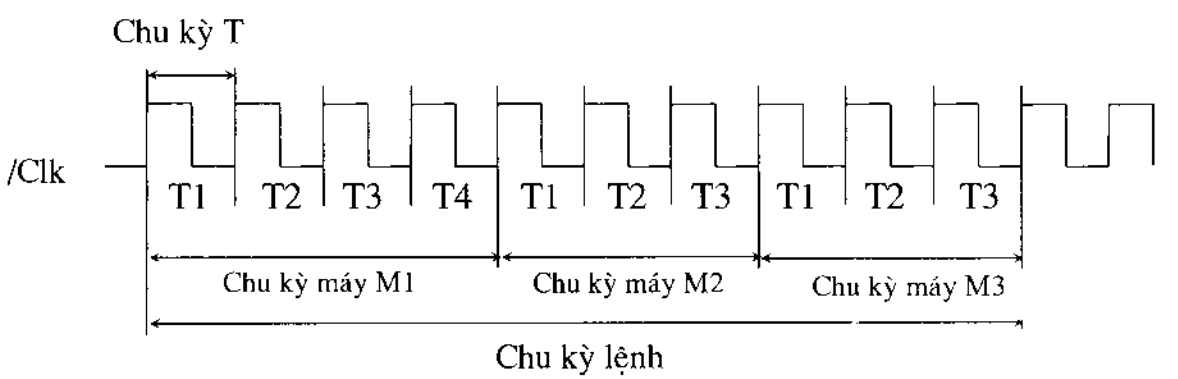
- /CLK: Clock (đầu vào) đầu vào cung cấp xung nhịp cho vi xử lý.

3. Các giản đồ thời gian

Z80 CPU thi hành các lệnh thông qua việc thực hiện từng bước các hoạt động cơ bản, các hoạt động cơ bản gồm:

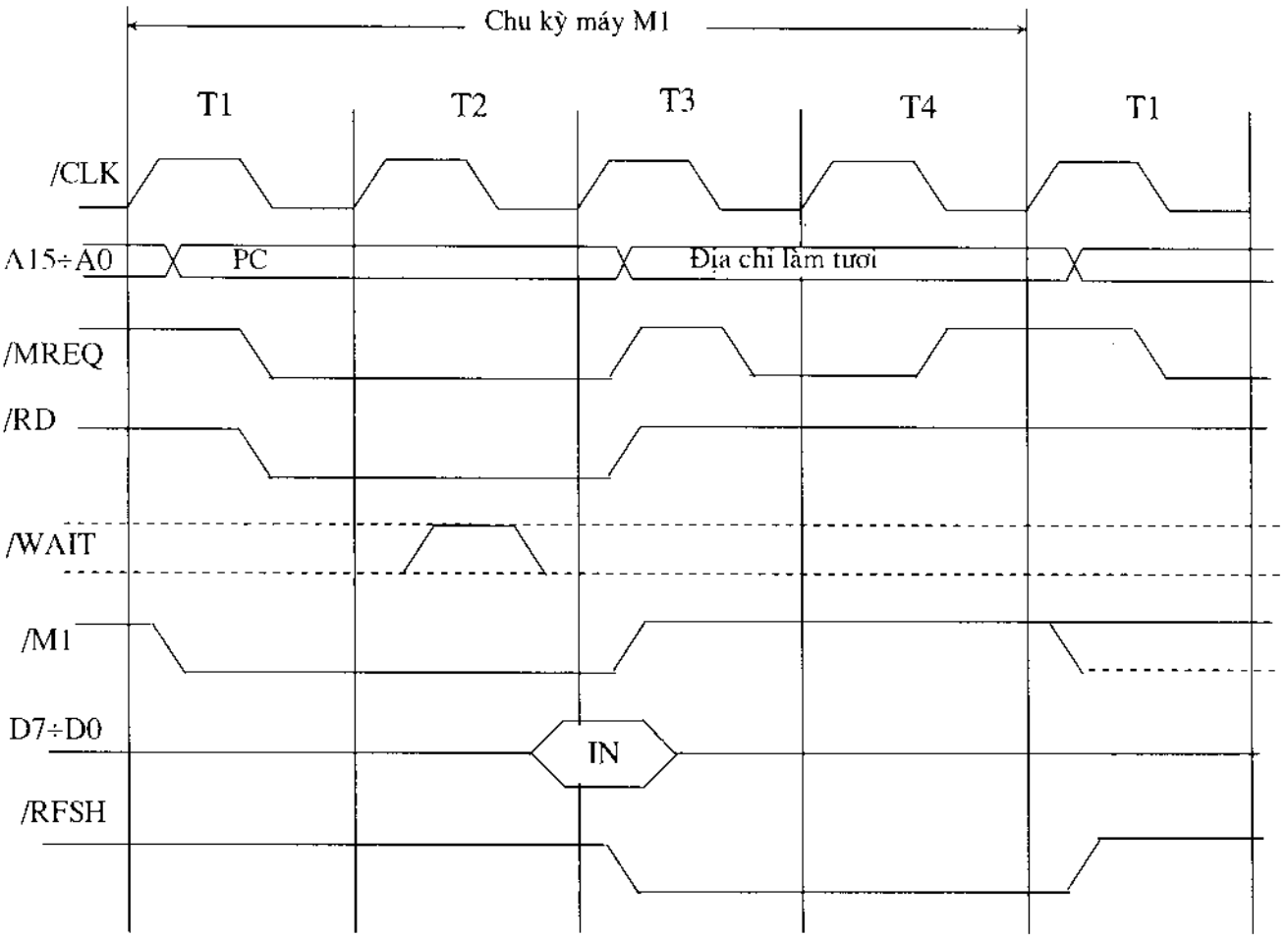
- ♦ Đọc và ghi bộ nhớ
- ♦ Đọc và ghi thiết bị I/O
- ♦ Đáp ứng ngắt

Lệnh là tập hợp các hoạt động cơ bản. Mỗi hoạt động cơ bản có thể chiếm từ 3 đến 6 chu kỳ xung clock để hoàn tất hoặc chúng có thể được kéo dài thêm để đồng bộ vị tốc độ giữa CPU với thiết bị bên ngoài. Chu kỳ xung Clock căn bản được gọi là chu kỳ T và thời gian thực hiện hoạt động cơ bản được gọi là chu kỳ máy M. Hình 3.4 minh họa một chu kỳ lệnh cơ bản. Lưu ý rằng lệnh này gồm 3 chu kỳ máy (M1, M2 và M3). Chu kỳ máy đầu tiên của bất kỳ lệnh nào là chu kỳ lấy mã lệnh (dài 4, 5 hay 6 chu kỳ T trừ khi bị kéo dài bởi tín hiệu Wait). Chu kỳ lấy lệnh (M1) được dùng để lấy mã lệnh sắp được thi hành. Chu kỳ máy sau M2 chuyển dữ liệu giữa CPU và bộ nhớ hay thiết bị I/O và chúng có thể có từ 3 đến 5 chu kỳ T (trừ khi bị kéo dài bởi tín hiệu Wait để đồng bộ với tốc độ thiết bị bên ngoài). Đoạn sau đây mô tả việc định thời (xảy ra trong bất kỳ chu kỳ máy nào). Trong khoảng thời gian T_2 và mỗi TW sau đó, CPU lấy mẫu đường Wait tại sườn xuống của xung clock. Nếu đường Wait được tích cực tại thời điểm này, một trạng thái Wait khác sẽ được thêm vào trong chu kỳ sau. Nhờ sử dụng kỹ thuật này mà việc đọc bộ nhớ có thể kéo dài để phù hợp với thời gian truy xuất của bất kỳ kiểu thiết bị bộ nhớ nào.



Hình 3.4. Ví dụ về giản đồ thời gian cơ bản của CPU

3.1. Chu kỳ lấy mã lệnh từ bộ nhớ

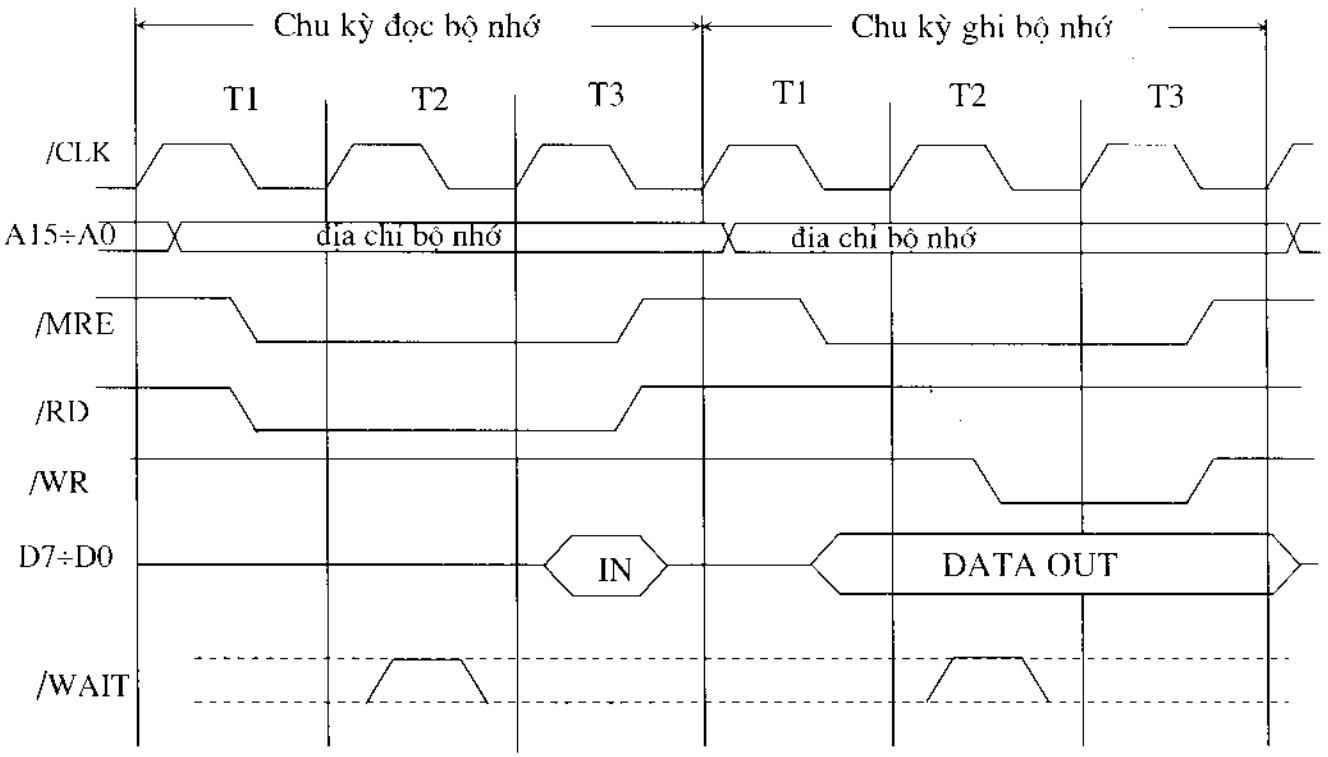


Hình 3.5. Chu kỳ lấy mã lệnh

Hình 3.5 là giản đồ thời gian của chu kỳ M1 (lấy mã lệnh). Nội dung thanh ghi đếm chương trình PC được đặt trên bus địa chỉ tại điểm bắt đầu của chu kỳ M1. Nửa chu kỳ clock sau đó tín hiệu /MREQ được tích cực. Tại thời điểm này địa chỉ tới bộ nhớ có đủ thời gian để ổn định do đó cạnh xuống của /MREQ có thể được dùng như là một tín hiệu cho phép bộ nhớ RAM động. Đường /RD sẽ tích cực để chỉ ra rằng dữ liệu được đọc từ bộ nhớ sẽ được đưa lên bus dữ liệu. CPU lấy dữ liệu bus dữ liệu tại sườn lên xung Clock của chu kỳ T3 và CPU dùng cạnh lên này để tắt tín hiệu /RD và /MREQ. Như vậy, dữ liệu đã sẵn sàng được CPU lấy trước khi tín hiệu /RD trở thành không tích cực. T3, T4 của chu kỳ lấy lệnh được dùng để làm tươi bộ nhớ RAM động. (CPU dùng thời gian này để giải mã và thực thi các lệnh khác đã được lấy về, do đó sẽ không có hoạt

động nào khác được thực hiện trong thời gian này). Trong suốt T3 và T4, bảy bit thấp của đường địa chỉ chứa địa chỉ làm tươi bộ nhớ RAM động. Chú ý rằng tín hiệu /RD không được sinh ra trong suốt thời gian làm tươi để tránh dữ liệu từ phần bộ nhớ khác bị đẩy lên bus dữ liệu. Tín hiệu /MREQ sẽ được sử dụng trong suốt thời gian làm tươi.

3.2. Chu kỳ đọc/ghi bộ nhớ



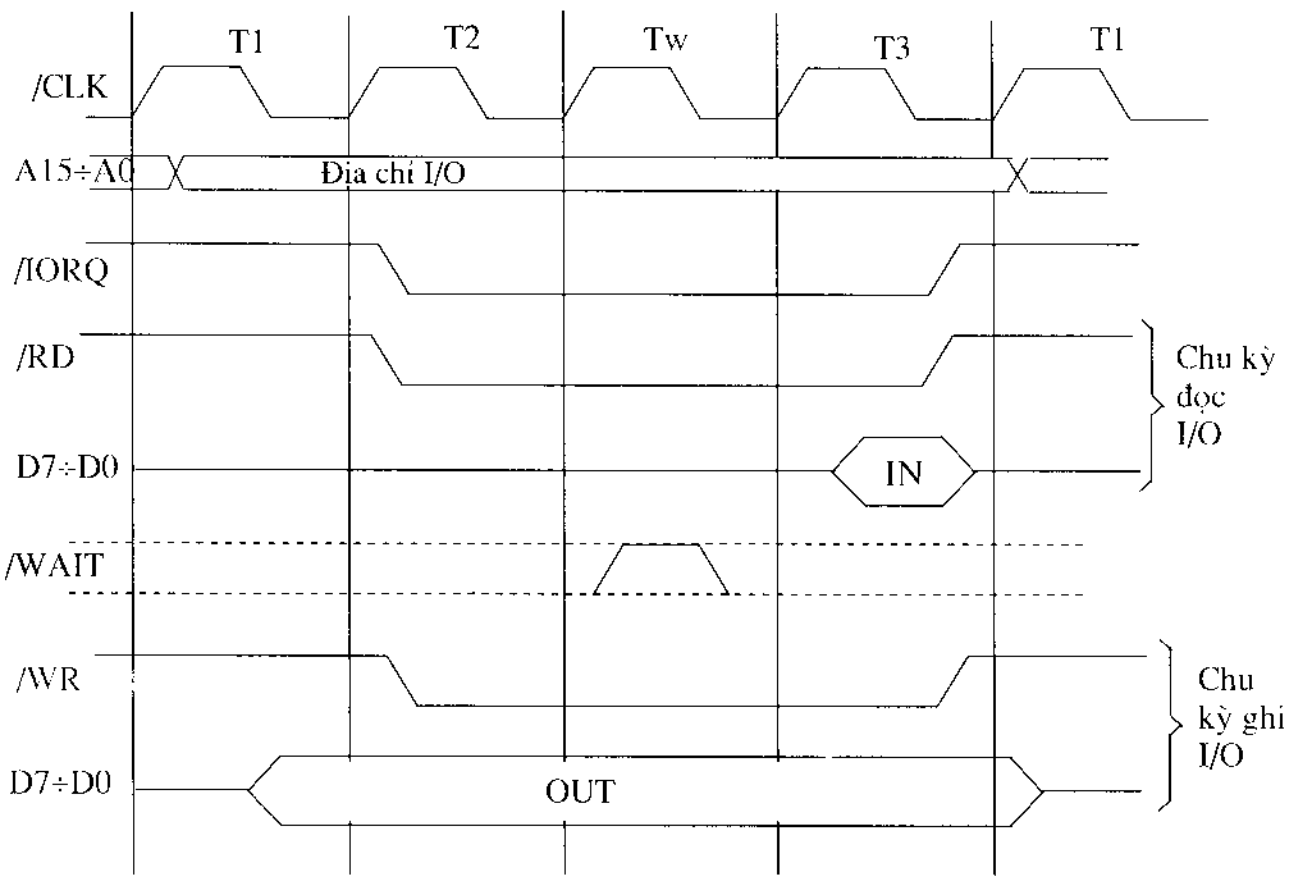
Hình 3.6. Chu kỳ đọc/ghi bộ nhớ

Hình 3.6 minh họa giản đồ thời gian của chu kỳ đọc/ghi bộ nhớ. Chu kỳ này thường được kéo dài trong 3 chu kỳ xung clock trừ phi có trạng thái đợi được bộ nhớ yêu cầu qua tín hiệu /WAIT. Tín hiệu /MREQ và tín hiệu /RD được dùng giống như trong chu kỳ lấy lệnh. Trong trường hợp ghi bộ nhớ, /MREQ cũng trở thành tích cực khi bus địa chỉ đã ở trạng thái ổn định do đó nó có thể được dùng trực tiếp như là tín hiệu chọn chip (/CS: chip select) cho bộ nhớ RAM động. Đường /WR tích cực khi dữ liệu trên bus dữ liệu đã ở trạng thái ổn định do đó nó có thể được dùng trực tiếp như là xung /WR cho các bộ nhớ bán dẫn.

3.3. Chu kỳ đọc/ghi với thiết bị ngoài (I/O)

Hình 3.7 minh họa hoạt động đọc/ghi I/O. Chú ý rằng trong thời gian hoạt động đọc hay ghi I/O thì một trạng thái đợi được tự động thêm vào. Lý do là trong quá trình hoạt động I/O, thời gian từ khi tín hiệu /IORQ trở nên tích cực cho đến khi CPU kiểm tra đường /WAIT thì rất ngắn do đó nếu không thêm trạng thái này vào thì không đủ thời gian để cổng I/O giải mã địa chỉ của nó. Ngoài ra, nếu không có trạng thái đợi thì sẽ khó khăn trong việc thiết kế các linh kiện I/O loại MOS để có thể hoạt động tương ứng với tốc độ cao của CPU. Trong thời gian xảy ra trạng thái đợi, tín hiệu /WAIT luôn được kiểm tra.

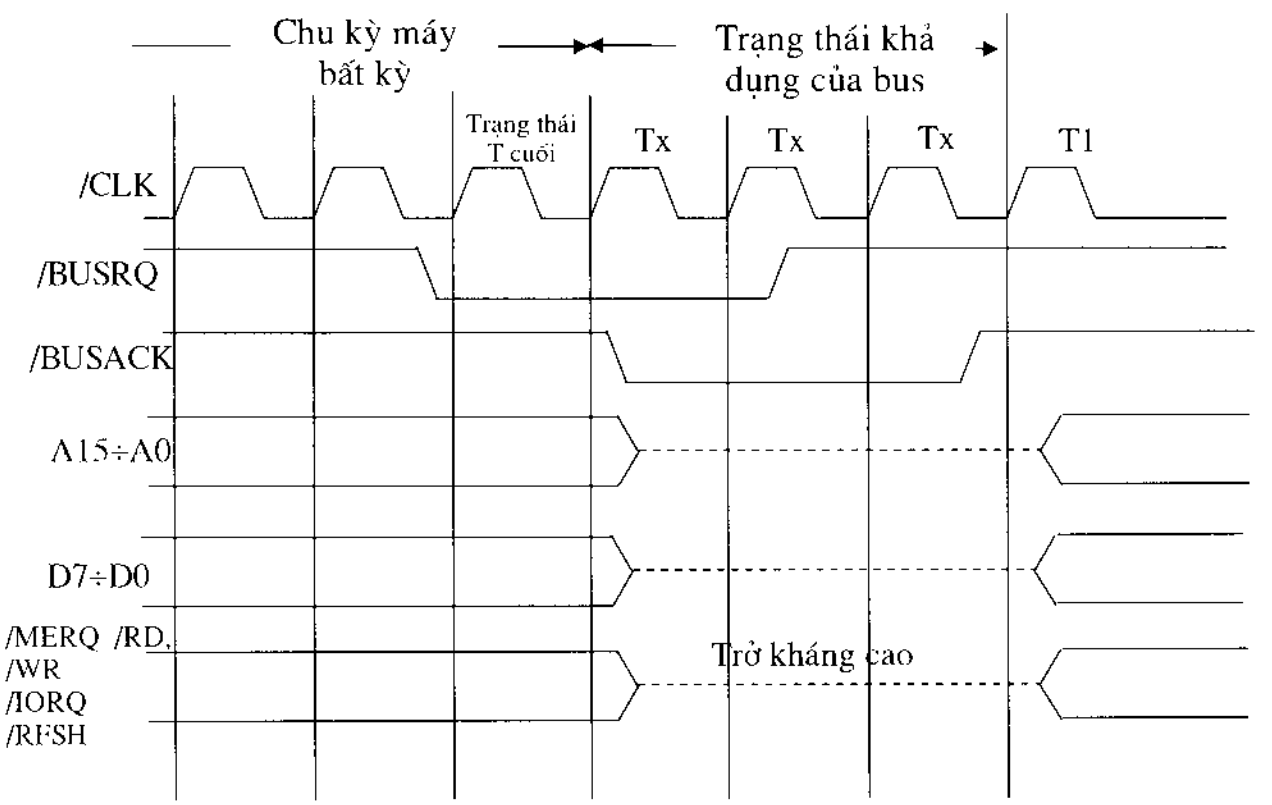
Trong quá trình đọc I/O, đường /RD được dùng để cho phép cổng được chọn đưa dữ liệu lên bus dữ liệu giống như trong trường hợp đọc bộ nhớ. Trường hợp ghi I/O, đường /WR được dùng như là xung Clock tới cổng I/O.



Hình 3.7. Chu kỳ đọc/ghi I/O

3.4. Chu kỳ yêu cầu/chấp nhận sử dụng bus

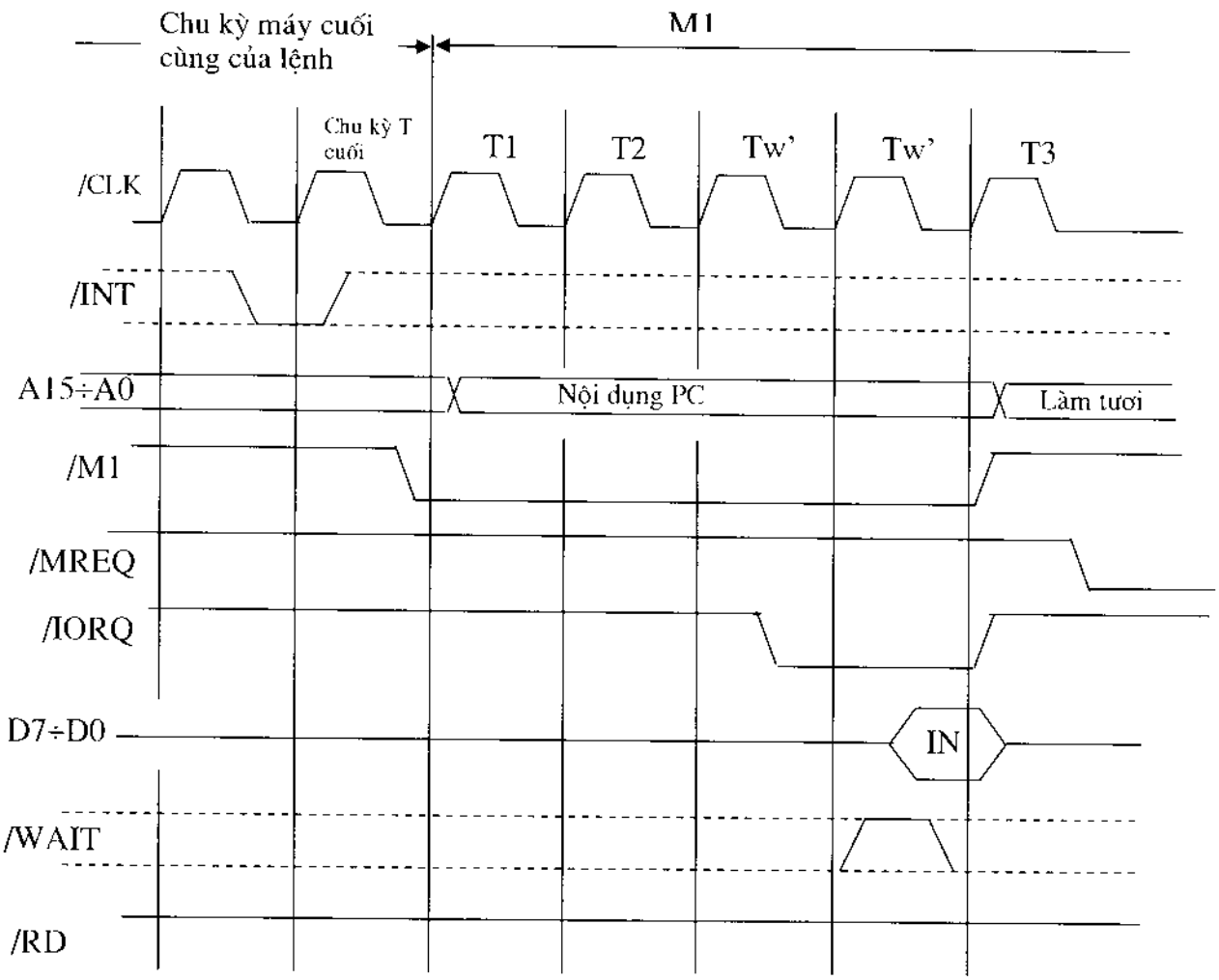
Hình 3.8 minh họa giản đồ thời gian của yêu cầu/chấp nhận sử dụng bus (bus request/ack). Tín hiệu /BUSREQ được CPU lấy mẫu ở cạnh lên của chu kỳ Clock cuối trong một chu kỳ máy. Nếu tín hiệu /BUSREQ tích cực, CPU sẽ đặt các đường dây địa chỉ, dữ liệu và các tín hiệu điều khiển ở trạng thái trở kháng cao tại cạnh lên của xung Clock kế tiếp. Tại thời điểm đó, thiết bị bên ngoài bất kỳ có thể kiểm soát bus để truyền dữ liệu giữa bộ nhớ và thiết bị I/O (ví dụ như quá trình truy nhập trực tiếp bộ nhớ DMA - Direct Memory Access). Thời gian tối đa cho CPU đáp ứng bus request bằng độ dài của 1 chu kỳ máy và thiết bị điều khiển ngoài có thể duy trì sự điều khiển của bus trong nhiều chu kỳ Clock theo yêu cầu. Nếu dùng chu kỳ DMA dài và bộ nhớ động được dùng, thiết bị điều khiển ngoài phải thực hiện chức năng làm tươi. Trường hợp này chỉ xảy ra khi một khối dữ liệu rất lớn được truyền dưới sự điều khiển của DMA. Cần chú ý rằng trong suốt chu kỳ Bus Request, CPU không thể bị ngắt bởi tín hiệu /NMI hay /INT.



Hình 3.8. Chu kỳ yêu cầu/chấp nhận bus

3.4. Chu kỳ yêu cầu/chấp nhận ngắt (Interrupt req/ack)

Hình 3.9 minh họa giản đồ thời gian trong trường hợp có một chu kỳ ngắt.

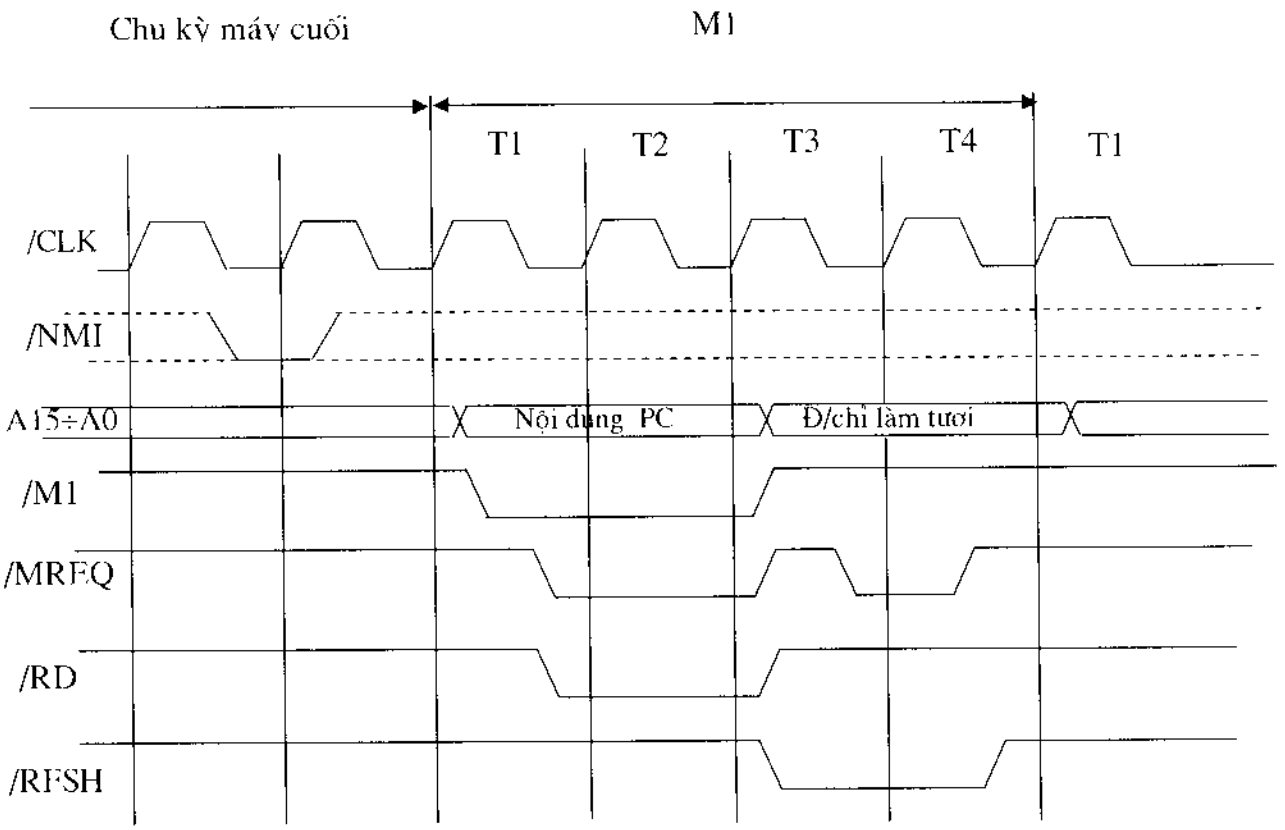


Hình 3.9. Chu kỳ yêu cầu/chấp nhận ngắt

Tín hiệu ngắt /INT được CPU lấy mẫu ở cạnh lên của Clock cuối tại điểm kết thúc của lệnh bất kỳ. Tín hiệu sẽ không được chấp nhận nếu F-F điều khiển ngắt không được phần mềm trong CPU khởi tạo lên mức logic '1' hay nếu tín hiệu /BUSREQ là tích cực. Khi tín hiệu ngắt được chấp nhận, chu kỳ đặc biệt M1 được tạo ra. Trong suốt chu kỳ đặc biệt M1, tín hiệu /IORQ trở nên tích cực (thay vì bình thường là /MREQ) để chỉ ra rằng thiết bị yêu cầu ngắt có thể đặt 1 vector ngắt 8 bit lên bus dữ liệu. Chú ý rằng hai trạng thái đợi được thêm vào chu kỳ này một cách tự động .

3.5. Đáp ứng ngắt không che được

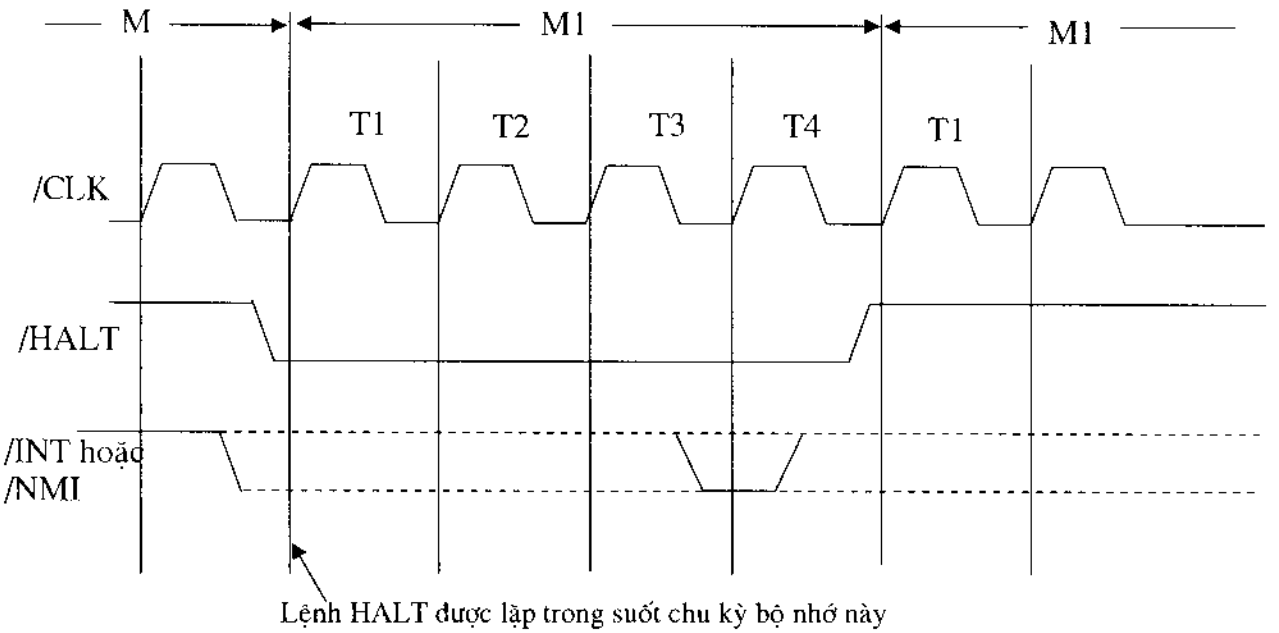
Hình 3.10 minh họa chu kỳ yêu cầu/chấp nhận cho ngắt không che được NMI (Non-Maskable Interrupt). Tín hiệu này được lấy mẫu cùng thời điểm với đường /INT, nhưng đường này có độ ưu tiên cao hơn và nó không thể bị cấm bởi phần mềm. Chức năng thông thường của nó là cung cấp đáp ứng tức thì cho các tín hiệu quan trọng như là lỗi nguồn cung cấp. CPU đáp ứng 1 ngắt không che được tương tự hoạt động đọc bộ nhớ bình thường. Sự khác biệt duy nhất là nội dung của bus dữ liệu bị lờ đi trong khi bộ vi xử lý tự động đẩy PC vào ngăn xếp (trên RAM) và nhảy tới địa chỉ 0066H. Do đó chương trình phục vụ ngắt không che được phải bắt đầu tại vị trí này.



Hình 3.10. Hoạt động yêu cầu ngắt không che được

3.6. Thoát khỏi trạng thái HALT

Khi một lệnh HALT của phần mềm được thực thi, CPU sẽ thực hiện 1 chuỗi lệnh NOP cho đến khi nó nhận được một tín hiệu ngắt (ngắt không che được NMI hoặc ngắt che được INT nếu Flip-Flop ngắt được cho phép). Hai đường ngắt này được lấy mẫu ở cạnh lên của chu kỳ T4 như được minh họa trong hình 3.11. Khi nhận được ngắt không che được hay ngắt che được (nếu Flip-Flop ngắt đang được thiết lập ở mức logic '1'), thì CPU sẽ thoát khỏi trạng thái HALT ở cạnh lên của chu kỳ clock kế tiếp. Chu kỳ sau sẽ là một chu kỳ đáp ứng ngắt tương ứng với ngắt nó nhận được. Nếu cả hai được thu tại cùng một thời điểm, thì ngắt không che được sẽ được đáp ứng vì nó có độ ưu tiên cao nhất. Mục đích của việc thực thi lệnh NOP trong khi CPU ở trạng thái HALT là để giữ cho việc làm tươi bộ nhớ RAM động được thực hiện. Mỗi chu kỳ trong trạng thái HALT là chu kỳ M1 bình thường (lấy lệnh). Tín hiệu báo HALT tích cực trong thời gian này để chỉ ra rằng vi xử lý đang ở trong trạng thái



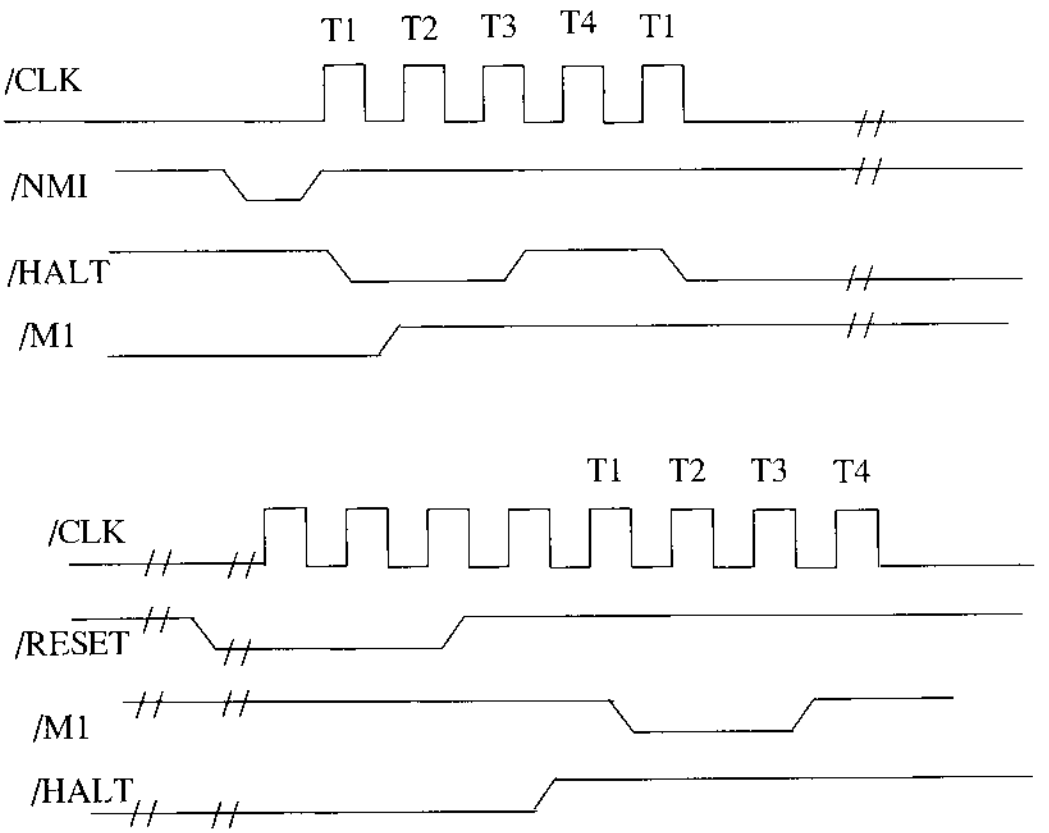
HALT.

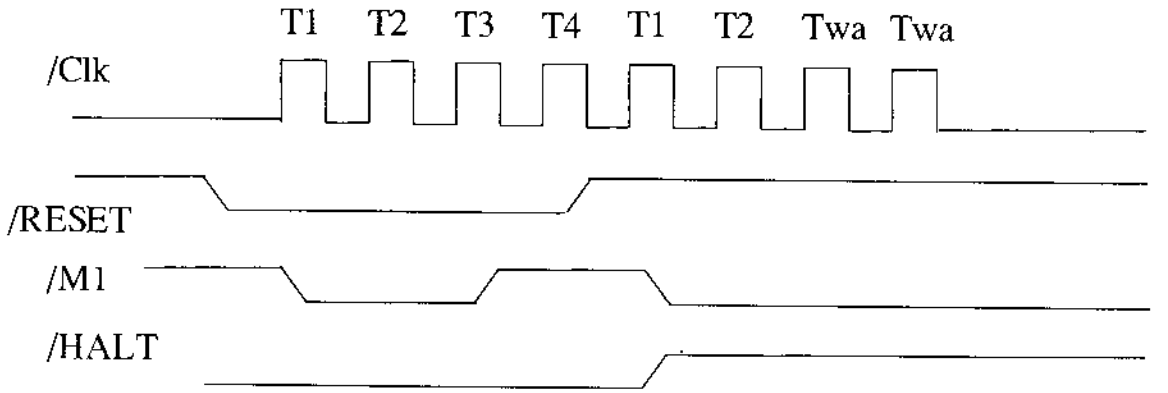
Hình 3.11. Chấp nhận mất xung

Khi xung nhịp Clock cấp cho CMOS Z80 CPU bị dừng lại ở mức cao hay mức thấp, CMOS Z80 CPU dừng hoạt động của nó và duy trì tất cả các thanh ghi và các tín hiệu điều khiển. Giản đồ thời gian cho chức năng Power-down khi thi hành lệnh HALT được chỉ ra trong hình 3.11.

3.7. Chu kỳ thoát khỏi trạng thái Power-down

Xung Clock hệ thống phải được cung cấp cho CMOS Z80 CPU để thoát khỏi trạng thái Power-down. Khi Clock hệ thống được cung cấp cho ngõ vào CLK, CMOS Z80 CPU sẽ khởi động lại hoạt động từ điểm mà tại đó trạng thái Power-down đã được thi hành. Giản đồ thời gian cho việc thoát khỏi chế độ Power-down được chỉ ra trong các hình 3.12, 3.13 và 3.14. Khi lệnh HALT được thực thi để đi vào trạng thái Power-down, CMOS Z80 CPU sẽ được đưa vào trạng thái HALT. Một tín hiệu ngắt (/NMI hay /INT) hay một tín hiệu /RESET được cấp cho CPU sau khi xung Clock hệ thống được đưa vào để thoát khỏi trạng thái Power-down.





Hình 3.12, 3.13, 3.14. Các chu kỳ thoát khỏi trạng thái Power-down

4. Ví dụ về một hệ vi xử lý đơn giản sử dụng Z80

4.1. Sơ đồ khối hệ thống

Hình 3.16 giới thiệu một hệ thống đơn giản sử dụng Z80, nó bao gồm:

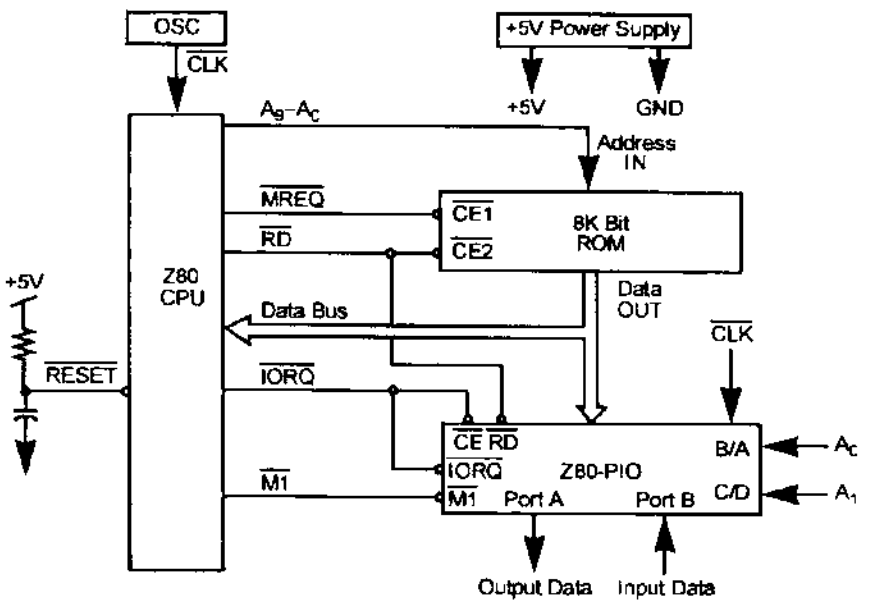
+ Nguồn cung cấp 5V

+ Mạch tạo xung nhịp

+ Các vi mạch nhớ

+ Mạch vào ra Z80-PIO

+ CPU Z80



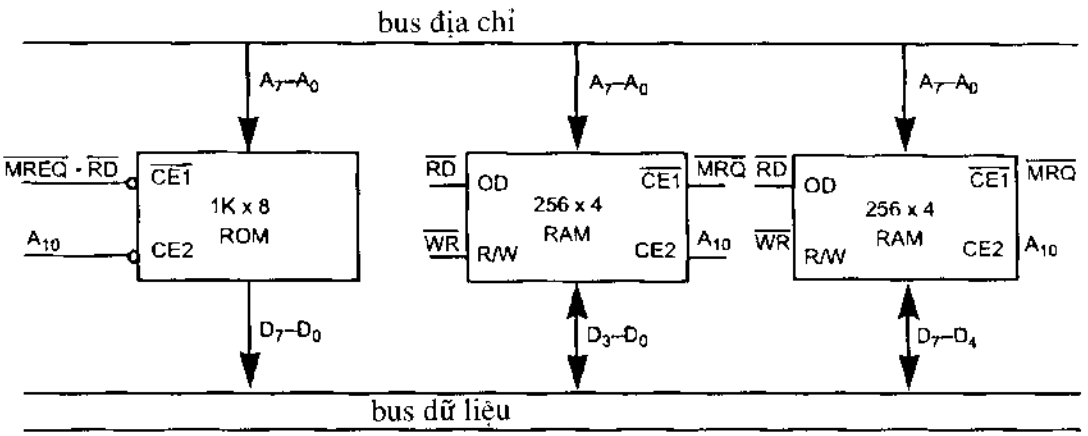
Hình 3.16. Máy tính đơn giản dùng Z80

Bộ nhớ ngoài có thể gồm ROM, RAM. Trong hình 3.16 chỉ sử dụng một vi mạch ROM 8 Kbit vì đây là một hệ thống tối thiểu các thao tác đọc/ghi dữ liệu được thực hiện trên các thanh ghi trong Z80 mà không cần sử dụng RAM ngoài.

Mạch I/O có nhiệm vụ cho phép hệ thống giao tiếp với bên ngoài, trong sơ đồ này ta sử dụng vi mạch vào ra song song 8 bit Z80 PIO, nếu muốn thực hiện vào ra nối tiếp thì cần sử dụng vi mạch vào ra nối tiếp Z80 SIO.

4.2. Thêm RAM cho hệ thống

Hầu hết các máy tính đều yêu cầu có bộ nhớ RAM để lưu trữ dữ liệu tạo ngăn xếp. Hình 3.17 minh họa cách tạo thêm 256 bytes SRAM cho hệ thống trong hình 3.16.



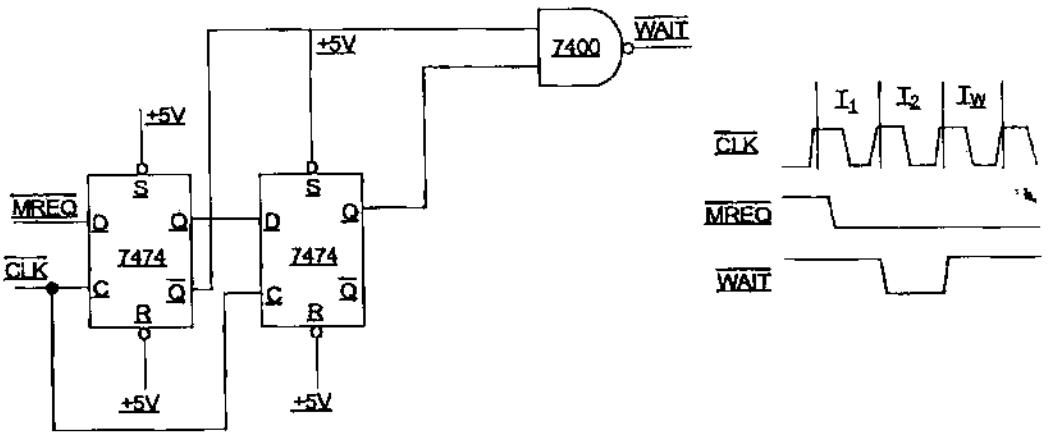
Hình 3.17. Sơ đồ ghép nối ROM, RAM

Không gian bộ nhớ được tổ chức như sau:

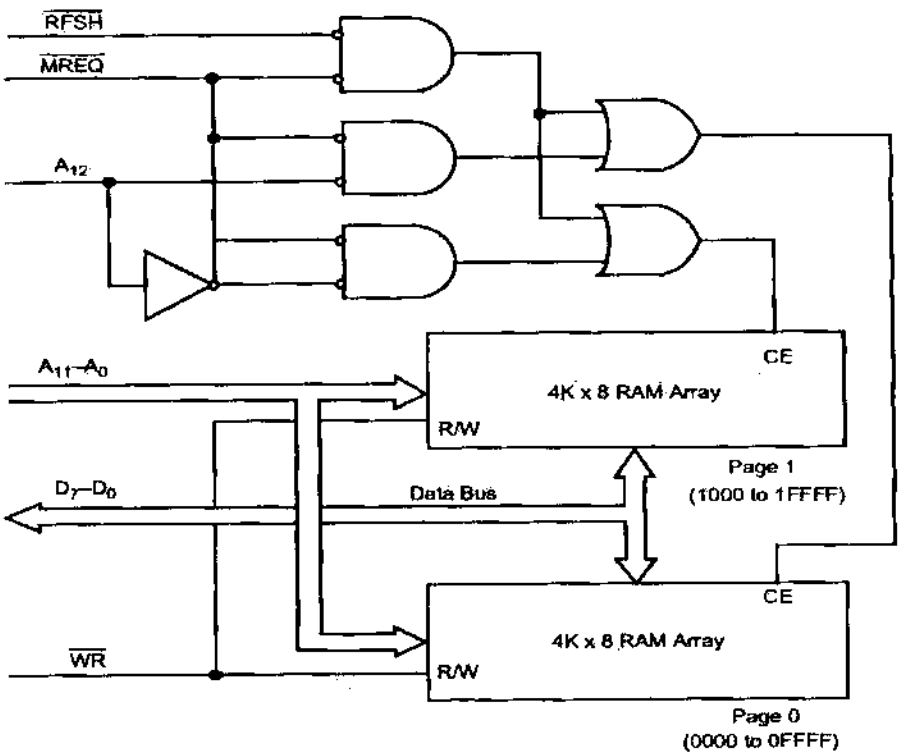
	Địa chỉ
1 Kbyte ROM	0000H
	03FFH
256 bytes RAM	0400H
	04FFH

Bit địa chỉ A10 được sử dụng để phân biệt giữa ROM và RAM, nó được nối với các chân chọn vi mạch (CE: Chip Enable). Trong dải địa chỉ

0000h đến 03FFh thì A10 = 0, do đó vi mạch ROM được chọn còn 2 vi mạch RAM bị cấm, trong dải địa chỉ từ 0400h đến 04FFh thì A10 = 1 nên vi mạch ROM bị cấm, hai vi mạch RAM được chọn, vi mạch RAM thứ nhất cung cấp các bít dữ liệu từ D0 đến D3, vi mạch RAM thứ hai cung cấp các bít dữ liệu từ D4 đến D7, cả hai vi mạch này cung cấp 1 byte dữ liệu hoàn chỉnh D0-D7. Sau đây chúng tôi sẽ giới thiệu mạch tạo trạng thái WAIT để đảm bảo giao tiếp giữa bộ nhớ với CPU (Hình 3.18) và mạch giao tiếp với DRAM để bạn đọc tham khảo.



Hình 3.18. Mạch thêm trạng thái WAIT vào chu kỳ truy cập bộ nhớ



Hình 3.19. Mạch giao tiếp với bộ nhớ DRAM

Chương 4

LẬP TRÌNH ASSEMBLY CHO Z80

1. Mô tả lệnh của Z80 CPU

Z80 CPU có thể thực thi 158 lệnh khác nhau bao gồm cả 78 lệnh của 8080A CPU. Các lệnh có thể được chia thành các nhóm chính sau:

1.1. Nạp và hoán chuyển (LOAD và EXCHANGE)

- Lệnh LOAD chuyển dữ liệu giữa các thanh ghi CPU với nhau hoặc giữa thanh ghi CPU với bộ nhớ ngoài. Tất cả các lệnh phải xác định nguồn và đích. Ví dụ như việc truyền Data từ thanh ghi C đến thanh ghi B. Nhóm này bao gồm cả LOAD tức thời tới thanh ghi CPU hoặc bộ nhớ ngoài bất kỳ. Các kiểu khác của lệnh LOAD cho phép truyền giữa các thanh ghi CPU và bộ nhớ.

- Lệnh EXCHANGE dùng để hoán vị nội dung của hai thanh ghi với nhau.

1.2. Tìm kiếm và chuyển khối dữ liệu

- Lệnh chuyển khối của Z80 được cung cấp để một khối bộ nhớ với kích thước bất kỳ có thể được chuyển đến 1 vị trí khác trong bộ nhớ. Các lệnh chuyển khối có lợi khi phải xử lý một chuỗi dữ liệu lớn.

- Lệnh tìm khối dùng để tìm 1 ký tự 8 bit trong khối bộ nhớ ngoài với chiều dài tùy thích có thể bất kỳ. Khi ký tự được tìm thấy hay gặp cuối khối, lệnh sẽ tự động chấm dứt. Cả hai lệnh chuyển và tìm khối đều có thể bị ngắt trong lúc thực thi, do đó không chiếm giữ CPU trong chuỗi thời gian dài.

1.3. SỐ HỌC và LOGIC

Lệnh LOGIC và SỐ HỌC tác động lên dữ liệu trong thanh ghi tích lũy, thanh ghi đa dụng và bộ nhớ ngoài. Kết quả của các phép toán được đặt

trong thanh ghi tích lũy và cờ đặc trưng được thiết lập tùy theo kết quả của phép toán. Một ví dụ về phép toán số học là cộng nội dung thanh ghi tích lũy với nội dung bộ nhớ ngoài. Kết quả của phép cộng được đặt vào thanh ghi tích lũy. Nhóm này cũng gồm phép cộng và trừ các thanh ghi 16 bit.

1.4. Quay và dịch

Nhóm lệnh quay (ROTATE) và dịch (SHIFT) cho phép quay phải (trái) hay dịch phải (trái) bất kỳ thanh ghi hay bộ nhớ nào có hay không có cờ Carry.

1.5. Thao tác với BIT (Set, Reset, Test)

Lệnh thao tác BIT cho phép bit bất kỳ trong thanh ghi tích lũy, thanh ghi đa dụng hay bộ nhớ ngoài được Set, Reset hay Test với 1 lệnh đơn. Ví dụ, MSB của thanh ghi H có thể Reset. Nhóm này đặc biệt hữu dụng trong các ứng dụng điều khiển.

1.6. Lệnh JUMP, CALL, và RETURN

Lệnh JUMP, CALL, RETURN được dùng để di chuyển giữa hai vị trí khác nhau. Kiểu khác của lệnh CALL là lệnh RESTART. Lệnh này chứa địa chỉ mới như 1 phần của mã lệnh 8 bit. Chương trình nhảy được thi hành bởi sự nạp trực tiếp thanh ghi HL, IX hay IY lên PC.

1.7. Lệnh INPUT/OUTPUT

Nhóm lệnh INPUT/OUTPUT trong Z80 cho phép truyền dữ liệu giữa bộ nhớ ngoài hay thanh ghi đa dụng với thiết bị I/O bên ngoài. Trong mỗi trường hợp, địa chỉ cổng được cung cấp trên 8 bit thấp của bus địa chỉ trong khi thực thi I/O. Sự thuận lợi khi dùng thanh ghi C như con trỏ chỉ tới thiết bị I/O là cho phép các cổng I/O khác nhau chia sẻ driver phần mềm chung. Điều không thể khi địa chỉ là 1 phần của mã lệnh nếu

chương trình được chứa trong ROM. Đặc điểm khác của lệnh INPUT là chúng set thanh ghi cờ tự động do đó không yêu cầu xác định trạng thái của dữ liệu nhập vào. Z80 CPU có một lệnh cho phép chuyển khối dữ liệu (lên đến 256 byte) một cách tự động tới hay từ cổng I/O bất kỳ đến vị trí bộ nhớ bất kỳ, kết hợp với tập các cặp thanh ghi đa dụng, lệnh này cho tốc độ truyền khối I/O cao. Giá trị của tập lệnh I/O này được chứng minh bởi Z80 có thể cung cấp các yêu cầu định dạng đĩa mềm (có nghĩa là CPU cung cấp header, địa chỉ, data, và cho phép mã CRC) trên đĩa mật độ cao dựa trên ngắt căn bản.

1.8. Lệnh điều khiển CPU căn bản

Lệnh điều khiển CPU căn bản cho phép các chức năng và chế độ khác nhau. Nhóm lệnh này gồm SETTING hay RESETTING các Flip-Flop cho phép ngắt hay cài đặt chế độ đáp ứng ngắt.

2. Các chế độ định địa chỉ

Hầu hết các lệnh Z80 hoạt động trên dữ liệu trong thanh ghi CPU, bộ nhớ ngoài, hay cổng I/O. Các chế độ định địa chỉ chỉ ra cách mà địa chỉ của dữ liệu được tạo ra trong mỗi lệnh là như thế nào. Phần dưới cho một tóm tắt sơ lược cho các loại định địa chỉ dùng trong Z80:

2.1. Định địa chỉ tức thời

Trong cách định địa chỉ này, byte theo sau mã lệnh trong bộ nhớ chứa toán hạng thực.

Ví dụ: Nạp thanh ghi tích lũy với hằng số, ở đây hằng số là byte ngay sau mã lệnh.

2.2. Định địa chỉ tức thời mở rộng

Cách định địa chỉ này là sự mở rộng của cách định địa chỉ tức thời trong đó hai byte theo sau mã lệnh là các toán hạng.

Ví dụ: Load 16 bit dữ liệu (2 byte) vào cặp thanh ghi HL.

2.3. Định địa chỉ trang không

Z80 có lệnh 1 byte đặc biệt Call để nhảy tới vị trí bất kỳ trong 8 vị trí trong trang không của bộ nhớ. Lệnh này đặt PC tới địa chỉ hiệu dụng trong trang không. Sự tiện dụng của lệnh này là chỉ cần một byte đơn để xác định 1 địa chỉ 16 bit vị trí (là nơi chứa chương trình con), do đó tiết kiệm được bộ nhớ.

2.4. Định địa chỉ tương đối

Định địa chỉ tương đối dùng một byte dữ liệu theo sau mã lệnh để xác định độ dời từ chương trình hiện tại tới nơi mà chương trình có thể nhảy tới. Độ dời được tính bằng cách cộng số bù hai vào địa chỉ của mã lệnh của lệnh kế tiếp.

Sự tiện lợi của định địa chỉ tương đối là nó cho phép nhảy tới vị trí gần đó mà chỉ cần hai byte bộ nhớ. Do đó, những lệnh này có thể giảm thiểu nhu cầu bộ nhớ. Độ dời có thể từ +127 đến -128 từ địa chỉ A+2. Vì vậy tổng độ dời của lệnh nhảy tương đối là +129 đến -126.

2.5. Định địa chỉ mở rộng

Định địa chỉ mở rộng cung cấp hai byte (16 bit) địa chỉ trong lệnh. Dữ liệu này có thể là địa chỉ đích chương trình có thể nhảy tới hoặc nó có thể là địa chỉ của toán hạng.

Định địa chỉ mở rộng dùng nhảy từ vị trí này tới vị trí khác trong bộ nhớ hay nạp và chứa dữ liệu trong vị trí bộ nhớ bất kỳ. Khi định địa chỉ mở rộng được dùng để xác định địa chỉ nguồn hay đích của toán hạng, ký hiệu (nn) sẽ được dùng để chỉ ra nội dung bộ nhớ tại nn, nn là địa chỉ 16 bit được chỉ ra trong cấu trúc lệnh. Điều này có nghĩa là hai byte bộ nhớ nn được dùng như là con trỏ chỉ đến vị trí bộ nhớ. Dấu ngoặc có nghĩa là giá trị trong chúng được dùng như là con trỏ chỉ đến vị trí bộ nhớ. Ví dụ (1200) chỉ nội dung của bộ nhớ tại địa chỉ 1200.

2.6. Định địa chỉ chỉ số

Trong loại định địa chỉ này, byte dữ liệu theo sau mã lệnh chứa độ dời được cộng với nội dung của thanh ghi chỉ số (mã lệnh xác định thanh ghi chỉ số được dùng) để tạo ra con trỏ tới bộ nhớ.

Ví dụ: Load nội dung của vị trí bộ nhớ (thanh ghi chỉ số + độ dời) vào trong thanh ghi tích lũy. Độ dời là số bù hai có dấu. Định địa chỉ có chỉ số làm đơn giản hóa chương trình dùng bảng dữ liệu có thanh ghi chỉ số chỉ đến đầu bảng. Trong Z80 cung cấp hai thanh ghi chỉ số là IX và IY nên có thể thao tác được hai hay nhiều bảng. Để định địa chỉ chỉ số ta dùng (IX + d) hay (IY + d). Ở đây d là độ dời nằm sau mã lệnh. Dấu ngoặc chỉ ra rằng giá trị được dùng như là con trỏ chỉ đến bộ nhớ ngoài.

2.7. Định địa chỉ thanh ghi

Nhiều mã lệnh của Z80 chỉ rõ thanh ghi nào được dùng.

Ví dụ: Load dữ liệu trong thanh ghi B sang thanh ghi C: LD B,C;

2.8. Định địa chỉ ngầm định

Định địa chỉ ngầm định là ám chỉ đến các phép toán mà mã lệnh tự động ngầm định một hay nhiều thanh ghi CPU như là nơi chứa các toán hạng. Ví dụ như các phép toán số học thanh ghi tích lũy luôn được ngầm định là nơi chứa kết quả.

2.9. Định địa chỉ gián tiếp thanh ghi

Loại định địa chỉ này cần cặp thanh ghi 16 bit (như là HL) để chỉ tới vị trí bất kỳ trong bộ nhớ. Loại lệnh này rất mạnh và nó được dùng trong 1 khoảng rộng các ứng dụng.

Ví dụ: Load thanh ghi tích lũy với dữ liệu trong bộ nhớ được trỏ bởi nội dung thanh ghi HL.

Định địa chỉ chỉ số là một dạng của định địa chỉ thanh ghi gián tiếp, khác nhau ở chỗ trong định địa chỉ chỉ số có cộng thêm độ dời. Định địa

chỉ thanh ghi gián tiếp cho phép truy xuất bộ nhớ tức thời đơn giản nhưng hữu hiệu. Lệnh tìm và truyền khối trong Z80 là loại mở rộng của định địa chỉ này, ở đó thanh ghi được tăng, giảm và so sánh một cách tự động, ký hiệu cho biết việc định địa chỉ thanh ghi gián tiếp là dấu ngoặc đơn bao quanh tên thanh ghi (được dùng như là pointer).

Ví dụ: Ký hiệu (HL) chỉ rằng nội dung thanh ghi HL được dùng như là một pointer chỉ tới ô nhớ.

Thông thường định địa chỉ gián tiếp thanh ghi dùng để xác định các toán hạng 16 bit. Trong trường hợp này, nội dung thanh ghi chỉ đến byte thấp hơn của toán hạng trong khi nội dung thanh ghi cộng 1 chỉ đến byte cao của toán hạng.

Ví dụ: LD A, (HL) dùng để nạp nội dung ô nhí có địa chỉ là HL vào thanh ghi A.

2.10. Định địa chỉ bit

Z80 chứa một lượng lớn tập các lệnh liên quan đến bit: SET, RESET và TEST. Những lệnh này cho phép bất kỳ vị trí bộ nhớ nào hoặc thanh ghi CPU sử dụng phép toán bit thông qua 1 trong 3 chế độ định địa chỉ trước đó (thanh ghi, thanh ghi gián tiếp và chỉ số). 3 bit trong mã lệnh chỉ rõ thanh ghi nào được dùng.

2.11. Sự kết hợp các chế độ định địa chỉ

Nhiều lệnh có nhiều hơn một toán hạng (như là lệnh SỐ HỌC hay LOAD). Trong trường hợp này có thể sử dụng hai kiểu định địa chỉ khác nhau.

Ví dụ: Load có thể dùng định địa chỉ tức thời để xác định nguồn và định địa chỉ thanh ghi gián tiếp hay định địa chỉ chỉ số để xác định đích.

3. Mã hóa lệnh

Phần này mô tả lệnh và bảng mã lệnh cho từng lệnh của Z80. Trong ngôn ngữ Assembly từ lệnh gợi nhớ được dùng cho từng lệnh. Tất cả các mã lệnh được liệt kê ở dạng số Hexa. Những mã lệnh 1 byte cần hai số Hex trong khi 2 byte mã lệnh cần 4 số Hex.

Từ lệnh gợi nhớ của Z80 bao gồm mã lệnh và 0, 1 hay 2 toán hạng. Lệnh không có toán hạng tức là toán hạng của nó được ngầm định. Lệnh LOGIC chỉ gồm 1 toán hạng không đổi (ví dụ lệnh OR) được đại diện bởi 1 từ lệnh gợi nhớ. Lệnh gồm 2 toán hạng khác nhau được đại diện bởi 2 từ lệnh gợi nhớ.

3.1. Load và Exchange

Bảng 4.1 định nghĩa mã lệnh cho tất cả lệnh Load 8 bit trong Z80 CPU. Bảng này cũng chỉ ra kiểu định địa chỉ của mỗi lệnh. Nguồn dữ liệu có thể tra ở hàng trên cùng trong khi đích được tra ở cột bên trái.

Ví dụ: Load thanh ghi C từ thanh ghi B dùng mã lệnh 48H.

Mã lệnh được định ở dạng số Hex và CPU lấy mã lệnh từ bộ nhớ ngoài trong thời gian M1, sau đó CPU tự động giải mã và truyền vào thanh ghi.

Từ lệnh gợi nhớ cho nhóm này là LD (LD đích, nguồn).

Ví dụ: LD (HL), D là nạp nội dung thanh ghi D vào địa chỉ mà thanh ghi HL chỉ tới. Dấu ngoặc quanh HL có nghĩa là nội dung của HL được dùng để chỉ đến 1 vị trí bộ nhớ nào đó. Tất cả các lệnh LOAD của Z80 đích luôn viết trước nguồn.

Chú ý trong bảng một vài mã lệnh LOAD dùng trong Z80 là 2 byte. Đây là phương pháp tận dụng hiệu quả bộ nhớ khi lệnh 8 bit, 16 bit, 24 bit, 32 bit được thi hành trong Z80. Do đó, những lệnh tận dụng bộ nhớ như phép toán số học và logic chỉ cần 8 bit.

	Ngầm định		Thanh ghi							Gián tiếp thanh ghi			Chỉ số		Đ/c mở rộng	Tức thì
	I	R	A	B	C	D	E	H	L	(HL)	(BC)	(DE)	(IX+d)	(IY+d)	(nn)	
A	ED 57	ED 5F	7F	78	79	7A	7B	7C	7D	7E	0A	1A	FD 7E d	DD 7E d	FD 3A nn	DD 3E n
B			47	40	41	42	43	44	45	46			DD 46 d	FD 46 d		DD 06 n
C			4F	48	49	4A	4B	4C	4D	4E			DD 4E d	FD 4E d		DD 0E n
D			57	50	51	52	53	54	55	56			DD 56 d	FD 56 d		DD 16 n
E			5F	58	59	5A	5B	5C	5D	5E			DD 5E d	FD 5E d		DD 1E n
H			67	60	61	62	63	64	65	66			DD 66 d	FD 66 d		DD 26 n
L			6F	68	69	6A	6B	6C	6D	6E			DD 6E d	FD 6E d		DD 36 n
(HL)			77	70	71	72	73	74	75							DD 7E d
(BC)			02													
(DE)			12													
(IX+d)			DD 77 d	DD 70 d	DD 71 d	DD 72 d	DD 73 d	DD 74 d	DD 75 d							DD 36 d n
(IY+d)			FD 77 d	FD 70 d	FD 71 d	FD 72 d	FD 73 d	FD 74 d	FD 75 d							FD 36 d n
(nn)			32 nn													
I			ED 47													
R			ED 4F													

Tất cả lệnh LOAD dùng định địa chỉ chỉ số để xác định nguồn và đích cùng với 3 byte bộ nhớ với byte thứ 3 là độ dời d.

Ví dụ:

Với lệnh gợi nhớ LD E, (IX +8) thì trình tự của nó trong bộ nhớ là

Địa chỉ A	DD
A+1	5E
A+2	08

Lệnh định địa chỉ mở rộng cũng là lệnh 3 byte.

Ví dụ: Lệnh Load vào thanh ghi A nội dung của địa chỉ 6F32H trong bộ nhớ được viết: LD A, (6F32H) và trình tự của nó trong bộ nhớ là:

Địa chỉ A	3A	Mã lệnh
A+1	32	Địa chỉ thấp
A+2	6F	Địa chỉ cao

Chú ý rằng byte thấp của địa chỉ luôn là toán hạng đầu.

Lệnh LOAD tức thời cho thanh ghi 8 bit đa dụng là lệnh 2 byte. Lệnh LOAD giá trị 36H vào thanh ghi H: LD H, 36H và trình tự của nó trong bộ nhớ là:

Địa chỉ A	26	Mã lệnh
A+1	36	Toán hạng

Load địa chỉ bộ nhớ dùng định địa chỉ chỉ số cho đích và định địa chỉ tức thời cho nguồn cần 4 byte.

Ví dụ: LD (IX-15), 21H và trình tự của nó trong bộ nhớ là:

Địa chỉ A	DD	Mã lệnh
A+1	36	Mã lệnh
A+2	F1	Độ dời
A+3	21	Load toán hạng

Chú ý rằng với cách định địa chỉ chỉ số thì độ dời luôn theo sau mã lệnh.

Bảng 4.2 liệt kê lệnh LOAD 16 bit. Chú ý rằng định địa chỉ mở rộng có thể dùng cho tất cả các cặp thanh ghi. Hoạt động gián tiếp thanh ghi cho con trỏ ngăn xếp là lệnh PUSH và POP. Điều khác biệt trong lệnh LOAD 16 bit là con trỏ ngăn xếp được tăng giảm tự động khi mỗi byte được đẩy vào hay lấy ra từ ngăn xếp.

Ví dụ: PUSH AF là lệnh 1 byte với mã lệnh F5H. Khi lệnh này được thực thi thì thứ tự các thao tác diễn ra là:

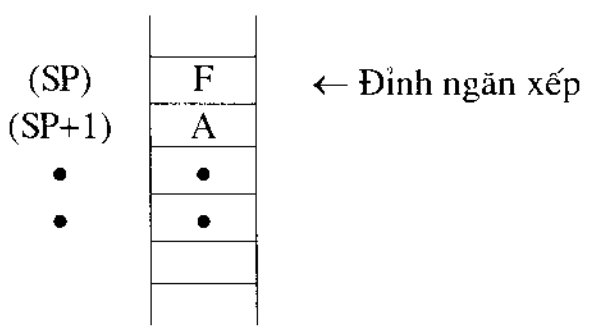
Giảm SP

LD (SP), A

Giảm SP

LD (SP), F

Do đó ngăn xếp ở bộ nhớ xuất hiện như sau:



Lệnh POP thì ngược với PUSH. Lệnh PUSH và lệnh POP dùng toán hạng 16 bit thì byte cao luôn được đẩy trước và lấy sau:

PUSH BC là đẩy B và sau đó là đẩy C

PUSH DE là đẩy D và sau đó là đẩy E

PUSH HL là đẩy H và sau đó là đẩy L

POP HL là lấy L và sau đó là lấy H

Lệnh dùng địa chỉ tức thời mở rộng cần 2 byte dữ liệu theo sau mã lệnh.

Ví dụ:

LD DE, 0659H

sẽ là:

Địa chỉ A	E6	Mã lệnh
A+1	07	Toán hạng

Trong kiểu định địa chỉ mở rộng tức thời, byte thấp luôn xuất hiện trước tiên ngay sau mã lệnh.

	Nguồn										
		Thanh ghi							Tức thì mở rộng	Đ/c mở rộng	Gián tiếp thanh ghi
		AF	BC	DE	HL	SP	IX	IY	nn	(nn)	(SP) (Lệnh POP)
Thanh ghi	AF										F1
	BC								01 n n	ED 4B n n	C1
	DE								11 n n	ED 5B n n	D1

	HL								21 n n	2A n n	E1
	SP				F9		D D F9	FD F9	31 n n	ED 7B n n	
	IX								DD 21 n n	DD 2A n n	DD E1
	IY								FD 21 n n	FD 2A n n	FD E1
Đ/c mở rộng	(nn)		ED 43 n n	ED 53 n n	22 n n	E D 73 n n	ED 22 n n	FD 22 n n			
Gián tiếp thanh ghi	(SP) (Lệnh PUSH)	F6	C6	D6	E6		D D E6	FD E6			

Bảng 4.2. Nhóm LD 16 bit, PUSH và POP

Bảng 4.3 liệt kê lệnh EXCHANGE trong Z80. Mã lệnh 08H cho phép người lập trình hoán đổi nội dung giữa 2 cặp thanh ghi tích luỹ, cờ trong khi D9H cho phép người lập trình hoán đổi nội dung giữa 6 thanh ghi đa dụng. Những mã lệnh này chỉ dài 1 byte nên nó giảm thiểu thời gian cần thiết để thi hành lệnh.

		Địa chỉ ngầm định				
		AF'	BC',DE',HL'	HL	IX	IY
Địa chỉ ngầm định	AF	08				
	BC DE HL		09			
	DE			EB		
Gián tiếp thanh ghi	(SP)			E8	DD E3	FD E3

Bảng 4.3. EX và EXX

3.2. Lệnh tìm và truyền khối

Bảng 4.4 liệt kê sự tiện lợi của lệnh truyền khối. Tất cả các lệnh đó khởi động với 3 thanh ghi :

HL: Chỉ tới vị trí nguồn

DE: Chỉ tới vị trí đích

BC: Đếm byte

Sau khi người lập trình đã khởi động 3 thanh ghi, bất cứ lệnh nào trong 4 lệnh đều có thể được dùng. Lệnh LDI (Load và tăng) đẩy 1 byte từ vị trí được trỏ bởi HL tới vị trí được trỏ bởi DE. Cặp thanh ghi HL và DE được tự động tăng và sẵn sàng chỉ đến vị trí sau. Byte đếm (cặp thanh ghi BC) cũng giảm tại thời điểm này. Lệnh LDIR (Load, tăng và lặp) là lệnh LDI mở rộng. Tác vụ tăng và Load được lặp lại cho đến khi byte đếm giảm xuống bằng 0. Do đó, lệnh đơn này có thể truyền bất cứ khối dữ liệu nào từ vị trí này đến vị trí khác.

Chú ý rằng khi dùng thanh ghi 16 bit, kích thước của khối được chuyển từ vị trí này đến vị trí khác trong bộ nhớ có thể lên đến 64 Kb (1K = 1024).

Lệnh LDD và LDDR tương tự như LDI và LDIR, chỉ khác là cặp thanh ghi HL và DE bị giảm sau khi mỗi lần chuyển do đó khối truyền khởi động từ địa chỉ cao nhất thay vì địa chỉ thấp nhất của khối.

	Nguồn (HL)	Giải thích
Đích (DE)	(ED) A0	"LDI": Load (DE) → (HL) Tăng HL và DE, giảm BC
	(ED) B0	"LDIR": Load (DE) → (HL) Tăng HL và DE, giảm BC lặp cho đến khi BC = 0
	(ED) A8	"LDD": Load (DE) → (HL) Tăng HL và DE, giảm BC
	(ED) B8	"LDDR": Load (DE) → (HL) Tăng HL và DE, giảm BC lặp cho đến khi BC = 0

Bảng 4.4. Nhóm lệnh truyền khối

Bảng 4.4 chỉ rõ mã lệnh của 4 lệnh tìm kiếm khối. Đầu tiên, CPI (So sánh và tăng) so sánh dữ liệu với giá trị mà thanh ghi HL chỉ đến. Kết quả của việc so sánh được chứa vào 1 trong các bit cờ và cặp thanh ghi HL tăng lên sau đó và byte đếm (cặp thanh ghi BC) thì giảm xuống.

Lệnh CPIR chỉ là lệnh CPI mở rộng trong đó sự so sánh được lặp cho đến khi 1 sự trùng nhau được tìm thấy hay byte đếm bằng 0. Do đó, lệnh đơn này có thể tìm trong bộ nhớ một ký tự 8 bit bất kỳ.

CPD (so sánh và giảm) và CPDR (so sánh, giảm và lặp) là lệnh giống nhau. Chỉ có 1 sự khác biệt là giảm HL sau mỗi lần so sánh do vậy chúng dò bộ nhớ theo chiều ngược lại. (Tìm kiếm được bắt đầu tại vị trí cao nhất trong khối bộ nhớ).

Vị trí tâm (HL)	
(ED) A1	"CPI" Tăng HL , giảm BC
(ED) B1	"CPR" Tăng HL giảm BC lặp cho đến khi BC=0 hoặc tìm thấy.
(ED) A9	"CPD" Load (DE) →(HL) Giảm HL và BC
(ED) B9	"CPDR" Giảm HL và BC lặp cho đến khi BC=0 hoặc tìm thấy.

Bảng 4.5. Nhóm lệnh tìm khối

3.3. Nhóm lệnh logic và số học

Bảng 4.6 liệt kê tất cả các lệnh số học 8 bit có thể được dùng với thanh ghi tích lũy. Trong tất cả các lệnh, trừ INC và DEC, phép toán 8 bit được thực thi giữa dữ liệu trong thanh ghi tích lũy và dữ liệu nguồn. Kết quả của phép toán được đặt trong thanh ghi tích lũy trừ lệnh CP không ảnh hưởng tới thanh ghi tích lũy. Các phép toán ảnh hưởng lên thanh ghi cờ được liệt kê trong bảng. Lệnh INC và DEC chỉ rõ thanh ghi hay vị trí bộ nhớ như là nguồn và đích. Khi thanh ghi chỉ số được dùng như toán hạng nguồn, độ dời phải được thêm vào. Với cách định địa chỉ tức thời, trong lệnh phải có một toán hạng cụ thể. Ví dụ:

AND 07H

sẽ xuất hiện trong bộ nhớ như sau:

Địa chỉ A	E6	Mã lệnh
A+1	07	Toán hạng

Giả sử thanh ghi tích lũy chứa giá trị F3H, kết quả của lệnh AND giữa F3 và 07 là 03H sẽ được đặt trong thanh ghi tích lũy:

Thanh ghi tích lũy trước khi thực hiện phép toán	1111 F3H	0011	=
Phép toán	0000 07H	0111	=
Kết quả trong thanh ghi tích lũy	0000 03H	0011	=

Lệnh ADD tiến hành phép cộng nhị phân giữa dữ liệu nguồn với dữ liệu trong thanh ghi tích lũy. Lệnh trừ làm phép trừ nhị phân. Khi cộng có nhớ (ADC) hay trừ có nhớ (SBC) cờ Carry cũng sẽ được cộng hay trừ. Cờ và lệnh hiệu chỉnh thập phân (DAA) trong Z80 cho các phép toán số học được thực hiện với:

- Độ chính xác cao cho số BCD.
- Độ chính xác cao cho số nhị phân có dấu và không dấu.
- Độ chính xác cao cho số bù hai có dấu.

Các lệnh khác trong nhóm này là lệnh Logical AND, OR, XOR và CP

Có 5 lệnh số học đa dụng thi hành trên thanh tích lũy hay cờ Carry được liệt kê trong bảng 4.7. Lệnh hiệu chỉnh thập phân có thể hiệu chỉnh cho phép trừ và phép cộng, như trong các phép toán số học trên số BCD đơn giản. Cờ N được dùng đến để cho phép thực hiện phép toán này. Cờ N được set khi phép toán số học cuối là phép trừ. Lệnh NEG bù hai toán hạng trong thanh ghi tích lũy. Chú ý rằng trong Z80 không có lệnh reset cờ, phép toán này có thể thực hiện dễ dàng bằng các lệnh khác như là lệnh logic AND của thanh ghi tích lũy với chính nó.

Hình 4-8 liệt kê các phép toán số học 16 bit giữa các thanh ghi 16 bit. Có 5 nhóm lệnh bao gồm cộng và trừ có Carry. ADC và SBC ảnh hưởng đến tất cả các cờ. Có hai nhóm: phép toán tính địa chỉ hay phép toán số học 16 bit.

	Địa chỉ thanh ghi							Gián tiếp thanh ghi	Chỉ số		Ngầm định
	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)	n
ADD	87	80	81	82	83	84	85	86	DD 86 d	FD 86 d	C8 n
ADD Carry ADC	8F	88	89	8A	8B	8C	8D	8E	DD 8E d	FD 8E d	CE n
SUB	97	90	91	92	93	94	95	96	DD 96 d	FD 96 d	D8 n
SBC	9F	98	99	9A	9B	9C	9D	9E	DD 9E d	FD 9E d	DE n
AND	A7	A0	A1	A2	A3	A4	A5	A6	DD A6 d	FD A6 d	E8 n
XOR	AF	A8	A9	AA	AB	AC	AD	AE	DD AE d	FD AE d	EE n
OR	B7	B0	B1	B2	B3	B4	B5	B6	DD B6 d	FD B6 d	F8 n
CP	BF	B8	B9	BA	BB	BC	BD	BE	DD BE d	FD BE d	FE n
INC	3C	04	0C	14	1C	24	2C	34	DD 34 d	FD 34 d	
DEC	30	05	00	15	10	25	20	36	DD 35 d	FD 35 d	

Bảng 4.6. Các lệnh số học và logic 8 bit

DAA	27
CPL	2F
NEG	ED 44
CCP	3F
SCP	37

Bảng 4.7. Các lệnh trên thanh ghi đa dụng AF

		BC	DE	HL	SP	IX	IY
ADD	HL	09	19	29	39		
	IX	DD 09	DD 19		DD 39	DD 29	
	IY	FD 09	FD 19		FD 39		FD 29
ADC	HL	ED 4A	ED 5A	ED 6A	ED 7A		
SBC	HL	ED 42	ED 52	ED 62	ED 72		
INC		03	13	23	33	DD 23	FD 23
DEC		0B	1B	2B	3B	DD 2B	FD 2B

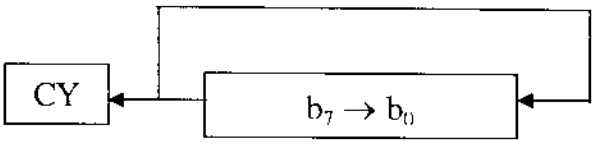
Bảng 4.8. Các lệnh số học 16 bit

3.4. Nhóm lệnh quay và dịch

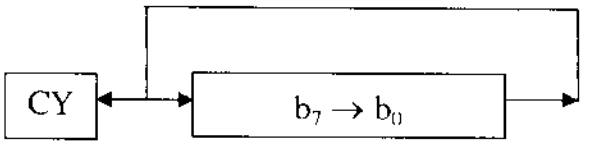
Khả năng quan trọng của Z80 là khả năng quay và dịch dữ liệu trong thanh ghi tích lũy, thanh ghi đa dụng bất kỳ, hay bất kỳ vị trí bộ nhớ nào. Tất cả các mã lệnh quay và dịch được chỉ ra trong bảng 4.9. Trong Z80 có nhiều lệnh dịch logic và số học. Các thao tác này rất có lợi trong khoảng ứng dụng rộng bao gồm nhân và chia số nguyên. Hai lệnh quay BCD là RRD và RLD cho phép 1 số trong thanh ghi tích lũy quay với 2 số trong bộ nhớ được chỉ bởi cặp thanh ghi HL, hai lệnh này rất có hiệu quả với số BCD.

	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)
RCL	CB 07	CB 00	CB 01	CB 02	CB 03	CB 04	CB 05	CB 06	DD CB d 06	FD CB d 06
RRC	CB 0F	CB 08	CB 09	CB 0A	CB 0B	CB 0C	CB 0D	CB 0E	DD CB d 0E	FD CB d 0E
RL	CB 17	CB 10	CB 11	CB 12	CB 13	CB 14	CB 15	CB 16	DD CB d 16	FD CB d 16
RR	CB 1F	CB 18	CB 19	CB 1A	CB 1B	CB 1C	CB 1D	CB 1E	DD CB d 1E	FD CB d 1E
SLA	CB 27	CB 20	CB 21	CB 22	CB 23	CB 24	CB 25	CB 26	DD CB d 26	FD CB d 26
SRA	CB 2F	CB 28	CB 29	CB 2A	CB 2B	CB 2C	CB 2D	CB 2E	DD CB d 2E	FD CB d 2E
SRL	CB 3F	CB 38	CB 39	CB 3A	CB 3B	CB 3C	CB 3D	CB 3E	DD CB d 3E	FD CB d 3E
SRL								ED 6F		
SRL								ED 67		

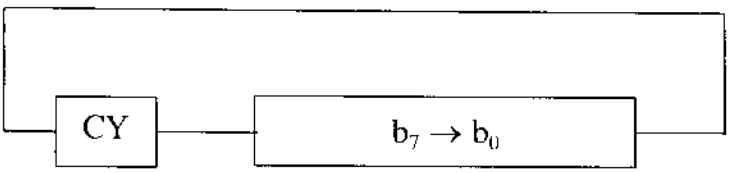
	A
RLCA	07
RRCA	0P
RLA	17
RRA	1F



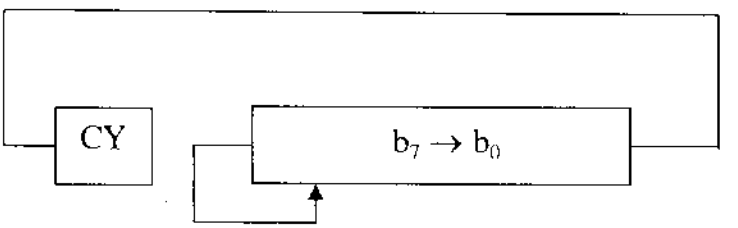
Vòng quay trái



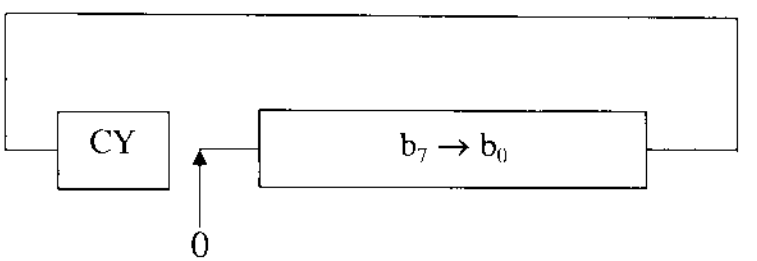
Vòng quay phải



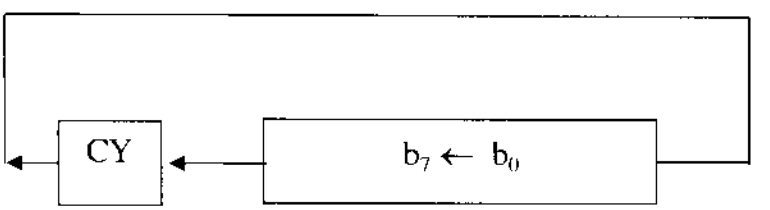
Quay phải



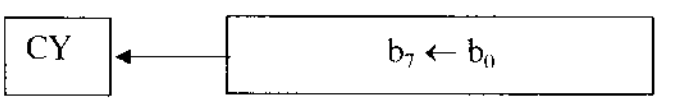
Dịch phải số học



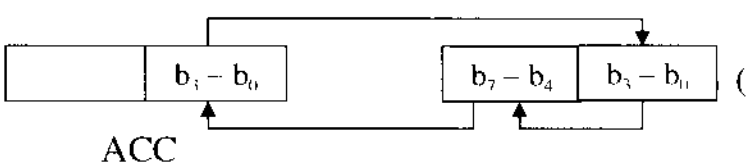
Dịch phải logic



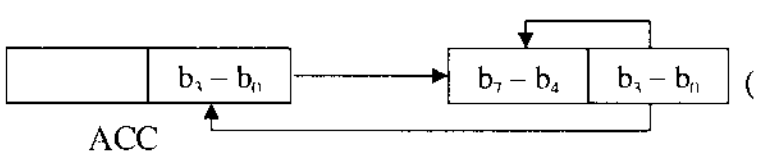
Quay trái



Dịch trái số học



Quay trái số BCD



Quay phải số BCD

Bảng 4.9. Các lệnh dịch và quay

3.5. Thao tác trên bit

Khả năng Set, Reset, và Test các bit riêng biệt trong thanh ghi hay bộ nhớ là cần thiết trong hầu hết các chương trình. Những bit này có thể là các cờ trong các chương trình con, báo các điều kiện điều khiển ...

Z80 có các lệnh có khả năng Set, Reset, và Test bất kỳ bit nào trong thanh ghi tích lũy, trong các thanh ghi đa dụng hay trong bộ nhớ. Bảng 4.10 liệt kê 240 lệnh cho mục đích này. Định địa chỉ thanh ghi có thể chỉ rõ các phép toán thực thi trên thanh ghi tích lũy hay thanh ghi đa dụng. Định địa chỉ gián tiếp thanh ghi và định địa chỉ chỉ số có thể thực hiện các phép toán trên bộ nhớ ngoài. Phép toán Test bit thiết lập cờ Zero (Z) lên 1 nếu kết quả test là zero.

		Địa chỉ thanh ghi							Gián tiếp thanh ghi	Chỉ số	
	Bit	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)
TEST BIT	0	CB 47	CB 40	CB 41	CB 42	CB 43	CB 44	CB 45	CB 46	DD CB d 46	FD CB d 46
	1	CB 4F	CB 48	CB 49	CB 4A	CB 4B	CB 4C	CB 4D	CB 4E	DD CB d 4E	FD CB d 4E
	2	CB 57	CB 50	CB 51	CB 52	CB 53	CB 54	CB 55	CB 56	DD CB d 56	FD CB d 56
	3	CB 5F	CB 58	CB 59	CB 5A	CB 5B	CB 5C	CB 5D	CB 5E	DD CB d 5E	FD CB d 5E

	4	CB 67	CB 60	CB 61	CB 62	CB 63	CB 64	CB 65	CB 66	DD CB d 66	FD CB d 66
	5	CB 6F	CB 68	CB 69	CB 6A	CB 6B	CB 6C	CB 6D	CB 6E	DD CB d 6E	FD CB d 6E
	6	CB 77	CB 70	CB 71	CB 72	CB 73	CB 74	CB 75	CB 76	DD CB d 76	FD CB d 76
	7	CB 7F	CB 78	CB 79	CB 7A	CB 7B	CB 7C	CB 7D	CB 7E	DD CB d 7E	FD CB d 7E
RESET BIT	0	CB 87	CB 80	CB 81	CB 82	CB 83	CB 84	CB 85	CB 86	DD CB d 86	FD CB d 86
	1	CB 8F	CB 88	CB 89	CB 8A	CB 8B	CB 8C	CB 8D	CB 8E	DD CB d 8E	FD CB d 8E
	2	CB 97	CB 90	CB 91	CB 92	CB 93	CB 94	CB 95	CB 96	DD CB d 96	FD CB d 96
	3	CB 9F	CB 98	CB 99	CB 9A	CB 9B	CB 9C	CB 9D	CB 9E	DD CB d 9E	FD CB d 9E
	4	CB A7	CB A0	CB A1	CB A2	CB A3	CB A4	CB A5	CB A6	DD CB d A6	FD CB d A6

	5	CB AF	CB A8	CB A9	C B A A	C B A B	C B A C	C B A D	CB AE	DD CB d AE	FD CB d AE
	6	CB B7	CB B0	CB B1	C B B2	C B B3	C B B4	C B B5	CB B6	DD CB d B6	FD CB d B6
	7	CB BF	CB B8	CB B9	C B B A	C B BB	C B B C	C B B D	CB BE	DD CB d BE	FD CB d BE
SET BIT	0	CB C7	CB C0	CB C1	C B C2	C B C3	C B C4	C B C5	CB C6	DD CB d C6	FD CB d C6
	1	CB CF	CB C8	CB C9	C B C A	C B C B	C B C C	C B C D	CB CE	DD CB d CE	FD CB d CE
	2	CB D7	CB D0	CB D1	C B D2	C B D 3	C B D 4	C B D 5	CB D6	DD CB d D6	FD CB d D6
	3	CB DF	CB D8	CB D9	C B D A	C B D B	C B D C	C B D D	CB DE	DD CB d DE	FD CB d DE
	4	CB E7	CB E0	CB E1	C B E2	C B E3	C B E4	C B E5	CB E6	DD CB d E6	FD CB d E6
	5	CB EF	CB E8	CB E9	C B E A	C B E B	C B E C	C B E D	CB EE	DD CB d EE	FD CB d EE

6	CB F7	CB F0	CB F1	C B F2	C B F3	C B F4	C B F5	CB F6	DD CB d F6	FD CB d F6
7	CB FF	CB F8	CB F9	C B F A	C B FB	C B FC	C B F D	CB FE	DD CB d FE	FD CB d FE

Bảng 4.10. Nhóm các lệnh thao tác bit

3.6. JUMP, CALL và RETURN

Bảng 4.11. liệt kê tất cả lệnh JUMP, call và return của Z80 CPU. Lệnh JUMP là lệnh rẽ nhánh trong chương trình, đến nơi mà PC sẽ được nạp với giá trị 16 bit được chỉ rõ bởi 1 trong 3 chế độ định địa chỉ (Định địa chỉ tức thời mở rộng, tương đối hoặc gián tiếp thanh ghi). Chú ý rằng với lệnh JUMP có điều kiện, điều kiện này được xác định trước khi lệnh JUMP thực thi. Nếu điều kiện này không xảy ra, chương trình sẽ tiếp tục thực hiện các lệnh tiếp theo. Các điều kiện phụ thuộc vào nội dung trong thanh ghi cờ. Lệnh JUMP sử dụng kiểu địa chỉ mở rộng tức thời được dùng để nhảy tới vị trí bất kỳ trong bộ nhớ, lệnh này cần 3 byte trong đó 2 byte để xác định địa chỉ 16 bit với vị trí byte thấp của địa chỉ nằm trước, theo sau là byte địa chỉ cao.

Ví dụ, lệnh nhảy không điều kiện tới địa chỉ 3E32H:

Địa chỉ A	C3	Mã lệnh
A+1	32	Địa chỉ thấp
A+2	3E	Địa chỉ cao

Lệnh nhảy sử dụng kiểu địa chỉ tương đối chỉ dùng 2 byte. Byte thứ 2 là số bù hai có dấu chính là độ dời tính từ PC hiện tại. Độ dời này có giá trị trong khoảng từ +129 đến -126 và được tính từ địa chỉ của mã lệnh hiện tại.

Có ba loại nhảy sử dụng kiểu địa chỉ gián tiếp thanh ghi, là những lệnh được thi hành bằng cách nạp cặp thanh ghi HL hay một trong hai thanh ghi chỉ số IX, IY lên PC. Khả năng này cho phép tính toán trước vị trí mà chương trình sẽ nhảy đến.

Lệnh CALL là một dạng đặc biệt của lệnh Jump, byte địa chỉ theo sau lệnh CALL được đưa vào ngăn xếp trước khi thực hiện lệnh nhảy. Lệnh RETURN ngược lại với lệnh CALL vì dữ liệu tại đỉnh ngăn xếp được lấy ra và đặt trực tiếp vào PC để tạo địa chỉ nhảy. Lệnh CALL và RETURN cho phép các chương trình chính gọi các chương trình con và chương trình ngắt. Trong họ Z80 có hai lệnh quay về đặc biệt là quay về từ chương trình con phục vụ ngắt (RETI) và quay về từ chương trình con phục vụ ngắt không che (RETN), chúng được CPU xử lý như là một lệnh nhảy không điều kiện với mã lệnh là C9H.

			Un-cond	Carry	Non Carry	Zero	Non Zero	Parity Even	Parity Odd	Sign Neg	Sign Pos	Reg B=0
Jump 'JP'	Trực tiếp mở rộng	(nn)	C3 n n	DA n n	D2 n n	CA n n	C2 n n	EA n n	E2 n n	FA n n	F2 n n	
Jump 'JR'	Tương đối	PC >e	18 e-2	38 e-2	30 e-2	28 e-2	20 e-2					
Jump 'JP'	Gián tiếp thanh ghi	(HL)	EB									
		(IX)	DD E9									
		(IY)	FD E9									
'Call'	Trực tiếp mở rộng	nn	CD n n	0C n n	D4 n n	CC n n	C4 n n	EC n n	E4 N N	FC n n	F4 n n	
Giảm B. nhảy nếu không zero DJNZ	Tương đối	PC+ e	18 e-2	38 e-2	30 e-2	28 e-2	20 e-2					
RE												10 e-2
Return 'RET'		(SP) (SP+ 1)	C9	D8	D0	C8	C0	E8	E0	F8	F0	
Return từ Int 'RETI'		(SP) (SP+ 1)	ED 4D									
Return từ ngắt không che 'RETN'	Reg Indr	(SP) (SP+ 1)	ED 45									

Bảng 4.11. Nhóm các lệnh JUMP, CALL và RETURN

Lệnh DJNZ làm đơn giản chương trình điều khiển vòng lặp, lệnh này có 2 byte, giảm thanh ghi B và nhảy nếu thanh ghi B chưa bằng '0'.

Bảng 4.12 liệt kê 8 mã lệnh cho lệnh khởi động lại RESTART. Lệnh này gồm 1 byte gọi đến 1 trong 8 địa chỉ bất kỳ trong danh sách. Bảng sau là từ lệnh gợi nhớ cho các lệnh gọi. Các lệnh này được dùng thường xuyên để tối thiểu hóa nhu cầu bộ nhớ.

		MÃ LỆNH	
ĐỊA CHỈ CALL	0000H	C7	'RST 0 '
	0008H	CF	'RST 8 '
	0010H	D7	'RST 16 '
	0018H	DF	'RST 24 '
	0020H	E7	'RST 32'
	0028H	EF	'RST 40 '
	0030H	F7	'RST 48 '
	0038H	FF	'RST 56 '

Bảng 4.12. Nhóm lệnh khởi động lại

3.7. INPUT/OUTPUT

Z80 có tập lệnh mở rộng INPUT và OUTPUT được chỉ ra trong bảng 4.13 và 4.14. Việc định địa chỉ của thiết bị INPUT và OUTPUT có thể dùng một trong các cách sau: định địa chỉ tuyệt đối, gián tiếp thanh ghi hay sử dụng thanh ghi C. Trong chế độ định địa chỉ gián tiếp thanh ghi, dữ liệu có thể được truyền giữa thiết bị I/O với bất kỳ thanh ghi nào.

Ngoài ra còn có 8 lệnh truyền khối sử dụng được giống như truyền khối bộ nhớ, cặp thanh ghi HL để chỉ tới khối nguồn (lệnh OUTPUT) hay đích (lệnh INPUT) trong khi thanh ghi B được dùng như là byte đếm. Thanh ghi C giữ địa chỉ cổng nơi mà lệnh INPUT và OUTPUT cần. Do độ dài của thanh ghi B là 8 bit nên khối được truyền có thể lên đến 256 byte.

Trong lệnh IN A,n và OUT n, A, địa chỉ n của thiết bị I/O nằm ở nửa thấp của bus địa chỉ (A7÷A0) trong khi nội dung thanh ghi tích lũy được truyền trên nửa cao của bus địa chỉ.

Trong tất cả các lệnh INPUT và OUTPUT với cách định địa chỉ gián tiếp thanh ghi, nội dung thanh ghi C được đưa lên byte thấp của bus địa chỉ (địa chỉ thiết bị), nội dung thanh ghi B được đưa lên byte cao của bus địa chỉ.

			Trực tiếp	Gián tiếp thanh ghi
			(n)	(c)
INPUT "IN"	Trực tiếp thanh ghi	A	D8 N	ED 78
		B		ED 40
		C		ED 48
		D		ED 50
		E		ED 58
		H		ED 60
		L		ED 68
"INT" – INPUT A tăng HL, giảm B	Gián tiếp thanh ghi	(HL)		ED A2
"INIR" - Input tăng HL, giảm B lặp lại nếu B≠0				ED B2
"IND" - Input giảm HL, giảm B				ED AA
"INDR" - Input giảm HL, giảm B lặp lại nếu B≠0				ED BA

Bảng 4.13. Nhóm lệnh nhập INPUT

			THANH GHI							Gián tiếp thanh ghi
			A	B	C	D	E	H	L	(HL)
OUT	Trực tiếp	(n)	D3 n							
	Gián tiếp thanh ghi	(c)	ED 79	ED 41	ED 49	ED 51	ED 59	ED 61	ED 69	
OUT Output tăng HL, giảm B	Gián tiếp thanh ghi	(c)								ED A3
OUT – Output giảm B lặp lại nếu B≠0		(c)								ED B3
OUT Output giảm HL & B		(c)								ED AB
OUT Output giảm HL & B lặp lại nếu B≠0		(c)								ED BB

Bảng 4.14. Nhóm lệnh xuất OUTPUT

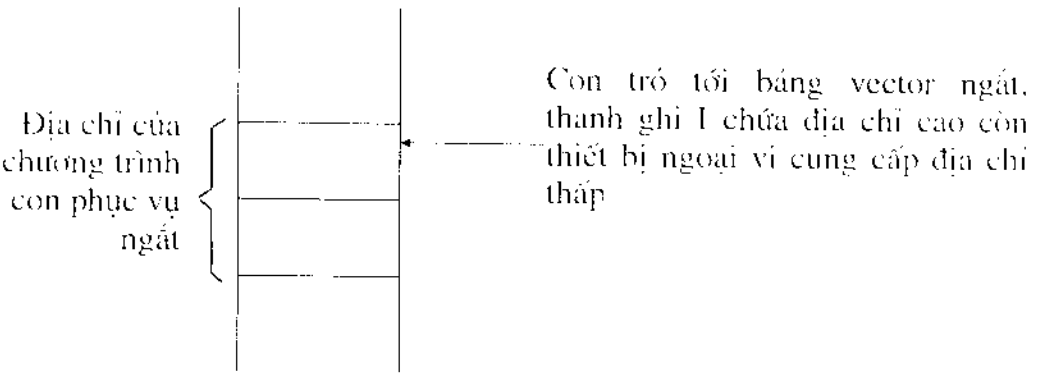
3.8. Nhóm lệnh điều khiển CPU

Bảng 4.15 miêu tả sáu lệnh điều khiển CPU. Lệnh NOP là lệnh không làm gì cả. Lệnh HALT hoãn hoạt động CPU cho đến khi thu được một lệnh ngắt, trong khi DI và EI được dùng để khóa và cho phép ngắt. Có ba chế độ ngắt:

- Chế độ 0: Thiết bị ngắt có thể đưa bất kỳ lệnh nào lên bus dữ liệu và cho phép CPU thi hành nó.

- Chế độ 1: Là chế độ đơn giản, CPU tự động thi hành một lệnh RST (khởi động lại) tới vị trí 0038H do đó phần cứng ngoài là không cần thiết (nội dung cũ của PC được đẩy lên ngăn xếp).

- Chế độ 2: Rất hữu dụng, nó cho phép gọi gián tiếp tới vị trí bộ nhớ bất kỳ. Với chế độ này, CPU thiết lập địa chỉ bộ nhớ 16 bit có 8 bit cao là nội dung của thanh ghi I và 8 bit thấp được cung cấp bởi thiết bị ngắt. Địa chỉ này chỉ đến byte thứ nhất trong chuỗi 2 byte liên tiếp trong bảng vector ngắt chứa địa chỉ của chương trình con phục vụ ngắt. CPU tự động lấy địa chỉ bắt đầu và thi hành một lệnh CALL tới địa chỉ này.



NOP	00	
HALT	76	
DISABLE INT '(DI)'	F3	
DISABLE INT '(EI)'	FB	
SET INT MODE 0 'IM0'	ED 46	8080A MODE
SET INT MODE 1 'IM1'	ED 56	CALL tới 0038H
SET INT MODE 2 'IM2'	ED 5E	CALL gián tiếp dùng thanh ghi I và 8 bit từ thiết bị

Bảng 4.15. Các lệnh điều khiển CPU

4. Một số ví dụ về lập trình Assembly cho Z80

Trước khi xem xét các ví dụ cụ thể chúng ta cần phải nắm được cách viết một chương trình cho Z80. Sau đây là ví dụ về một chương trình mẫu cho Z80:

Trường nhãn	Trường mã lệnh	Trường toán hạng	Trường chú thích
	ORG	8000H	
	LD	A,01H	
LOOP:	INC	A	;đặt giá trị đầu
	OUT	(0FDH), A	cho thanh ghi A
	JP	LOOP	
	END		

Ta thấy rằng một lệnh gồm có 4 trường:

- *Trường nhãn* (Label) để chỉ các địa chỉ rẽ nhánh. Nhãn được đặt theo qui tắc sau:

+ Không quá 6 kí tự bao gồm cả chữ cái và chữ số, trong đó kí tự đầu tiên phải là chữ cái.

+ Không trùng với các lệnh gợi nhớ như LD, JP ...

+ Kết thúc tên nhãn là dấu hai chấm ':'.

+ Phải có dấu cách giữa nhãn và mã lệnh.

- *Trường mã lệnh* để xác định hành động của lệnh.

- *Trường toán hạng* xác định đối tượng mà lệnh tác động tới. Toán hạng có thể là thanh ghi, ô nhớ, hằng số, địa chỉ biểu diễn bởi một nhãn.

- *Trường chú thích* thường được sử dụng để giải thích ý nghĩa của lệnh, nó được bắt đầu bởi dấu ';'.

*** Các lệnh mã giả**

- ORG: Được sử dụng để xác định địa chỉ bắt đầu trong bộ nhớ, nếu không xác định địa chỉ đầu thì mặc định là 0.

Ví dụ:

ORG 8000H; địa chỉ bắt đầu của chương trình là 8000h.

- END: Báo kết thúc chương trình.

- EQU: Dùng để định nghĩa một giá trị hằng số cho một nhãn ở bên trái.

Ví dụ:

MOTOR EQU 2

.....

OUT (MOTOR),A

Lệnh OUT (MOTOR), A có nghĩa là xuất dữ liệu trong thanh ghi A ra cổng ra có địa chỉ là 2 để điều khiển hoạt động của một môtơ.

- BD (Define Byte): dùng để định nghĩa byte dữ liệu trong bộ nhớ.
- DW (Befine Word): dùng để định nghĩa từ dữ liệu trong bộ nhớ.
- DS (Define Storage): dùng để tạo một vùng nhớ dự trữ.

Ví dụ:

DS 10; để dành 10 bytes tiếp theo

DS 10H; để dành 16 bytes tiếp theo

Sau đây là các ví dụ minh họa:

Ví dụ 1

Chương trình nạp một hằng số 25H vào thanh ghi B sau đó chuyển sang thanh ghi C.

ORG 8000H; chương trình bắt đầu ở địa chỉ 8000H

LD B, 25H; nạp giá trị 25H vào thanh ghi B

LD C, B; nạp nội dung của thanh ghi B vào thanh ghi C nên nội dung C sẽ là 25H

HALT

END

Sau khi chạy chương trình thì cả B và C đều chứa giá trị là 25H.

Ví dụ 2

Chương trình lưu giá trị 10H vào ô nhớ có địa chỉ là 8200H

ORG 8000H

LD A, 10H; nạp giá trị 10H vào thanh ghi chứa A

LD (8200H), A; nạp nội dung của A (là 10H) vào ô nhớ 8200H

HALT

END

Ví dụ 3

Chương trình lưu địa chỉ 8200H vào cặp thanh ghi HL sau đó cất giá trị 15H vào địa chỉ đó.

ORG 8500H

LD HL, 8200H; nạp giá trị 8200H vào cặp thanh ghi HL

LD (HL), 15H; lưu giá trị 15H vào ô nhớ có địa chỉ lưu ở HL (là 8200H)

HALT

END

Ví dụ 4

Tính tổng 8 + 9 rồi cất kết quả vào ô nhớ có địa chỉ là 8200H

ORG 8000H

LD A, 8; nạp 8 vào thanh ghi A

ADD A, 9; cộng nội dung của thanh ghi A với 9, kết quả cất tại A

LD (8200H), A; lưu nội dung trong A vào ô nhớ có địa chỉ là 8200H

HALT

END

Ví dụ 5

Lưu hằng 0102H vào cặp thanh ghi HL sau đó nhân giá trị này với 3 rồi lưu kết quả ra địa chỉ 8200H và 8201H

```
ORG 8000H
LD HL, 0102H
LD B, L
LD C, H
ADD HL, HL; cộng nội dung  HL với chính nó, kết quả đặt trong HL
ADD HL, BC; cộng nội dung HL với nội dung BC (0102H)
LD (8200H), HL; lưu giá trị trong HL vào ô nhớ có địa chỉ 8200H
HALT
END
```

Chương trình này thực hiện phép nhân 3 bằng cách lưu nội dung của HL vào BC, sau đó cộng nội dung HL với chính nó (tương đương nhân 2) và cộng nội dung HL với nội dung BC.

Ví dụ 6

Tính hiệu 9 - 3 rồi lưu kết quả vào địa chỉ 8200H

```
ORG 8000H
LD A, 9; nạp 9 vào thanh ghi A
SUB 3; trừ nội dung của A đi 3
LD (8200H), A; lưu nội dung A vào ô nhớ địa chỉ 8200H
HALT
END
```

Ví dụ 7

Nạp hằng số 04H vào thanh ghi A, sau đó tính số bù 2 của nó rồi lưu kết quả vào ô nhớ có địa chỉ là 8200

```
ORG 8000H
LD A,04H; nạp 04H vào A
CPL; lấy bù 2 của A rồi lưu ở A
LD (8200H), A
HALT
END
```

Vì số bù 2 của một số là số bù 1 (lấy phủ định) của nó cộng với 1 nên ta có thể có cách tính khác như sau:

```
ORG 8000H
LD A, 04H; nạp 04H vào A
NEG; lấy bù 1 của A rồi lưu ở A
INC A; tăng nội dung của A thêm 1
LD (8200H), A
HALT
END
```

Ví dụ 8

Nạp hằng số EBH vào địa chỉ 8200H, sửa cho nó thành 0BH rồi lưu vào địa chỉ 8300H

Số EBH viết ở hệ 2 là: 1110 1011B

Số 0BH viết ở hệ 2 là: 0000 1011B

Do đó muốn biến EBH thành 0Bh ta chỉ việc nhân logic EBH với 0FH.

Hệ 16	Hệ 2
EB	1110 1011
AND 0F	AND 0000 1111
= 0B	= 0000 1011

Chương trình thực hiện:

```
ORG 8000H
LD A, EBH
LD (8200H), A
LD A, EBH
AND 0FH
LD (8300H), A
HALT
END
```

Ví dụ 9

Tính tổng logic (OR) của 66H và 56H, cất kết quả vào ô nhớ có địa chỉ 8200H

```
ORG 8000H
LD B, 66H
LD C, 56H
LD A, B
OR C; thực hiện phép OR giữa A với C
LD (8200H), A
HALT
END
```

Ví dụ 10

Tính XOR của 66H và 56H, cất kết quả vào ô nhớ có địa chỉ 8200H

```
ORG 8000H
LD B, 66H
LD C, 56H
LD A, B
XOR C; thực hiện phép XOR giữa A với C
LD (8200H), A
HALT
END
```

Ví dụ 11

Nạp giá trị 06H vào thanh ghi A, nhân giá trị này với 5 rồi lưu kết quả ra địa chỉ 8200H, sử dụng lệnh quay RLA hoặc quay RLCA.

Với lệnh quay trái RLA, nếu nội dung cờ nhớ CY là 0 thì khi thực hiện RCA một lần nội dung trong A tăng gấp đôi. Để thực hiện nhân 06H với 5 ta thực hiện lệnh quay trái này 2 lần (tương đương nhân nội dung A với 4) rồi cộng nội dung A với 06H.

```
ORG 8000H
LD A,06H
LD B, A
SCF; thiết lập cờ nhớ CY
CCF; xóa cờ nhớ CY
RLA; quay trái A
RLA; quay trái A
ADD A, B; cộng nội dung A với nội dung B, cất kết quả ở A
```

LD (8200H), A; lưu nội dung của A vào ô nhớ 8200H

HALT

END

Với lệnh quay trái RLCA, cờ nhớ CY không ảnh hưởng đến vòng quay nên không cần xóa CY. Để thực hiện nhân 06H với 5 ta thực hiện lệnh quay trái RLCA 2 lần (tương đương nhân nội dung A với 4) rồi cộng nội dung A với 06H.

ORG 8000H

LD A, 06H

LD A, B

RLCA; quay trái A

RLCA; quay trái A

ADD A, B; cộng nội dung A với nội dung B, cất kết quả ở A

LD (8200H), A; lưu nội dung của A vào ô nhớ 8200H

HALT

END

Ví dụ 12

Đưa giá trị 0FH ra cổng FDH

ORG 8000H

LD A, 0FH;

OUT (0FD), A; xuất nội dung thanh ghi A ra cổng FDH

HALT

END

Ví dụ 13

Đọc dữ liệu từ cổng vào có địa chỉ là FCH, sau đó đưa giá trị đọc được ra cổng FDH

```
ORG 8000H
IN A, (0FCH); đọc cổng FCH, lưu giá trị đọc được vào thanh ghi A
OUT (0FD), A; xuất nội dung thanh ghi A ra cổng FDH
HALT
END
```

Ví dụ 14

Giả sử có 8 LED nối vào cổng ra có địa chỉ là FDH. Lập trình điều khiển các LED sáng theo trình tự sau:

Trạng thái của 8 LED	Tương đương giá trị đưa ra ở cổng FD
T T T T T T T S	01H
T T T T T T S T	02H
T T T T T S T T	04H
T T T T S T T T	08H
T T T S T T T T	10H
T T S T T T T T	20H
T S T T T T T T	40H
S T T T T T T T	80H
T T T T T T T S	01H
.....	...

Trong đó S trạng thái LED sáng, ứng với trường hợp đầu ra điều khiển là 1, T trạng thái LED tắt, ứng với trường hợp đầu ra điều khiển là 0.

```
ORG 8000H
LD A, 01H; nạp giá trị đầu
LOOP: OUT (0FDH), A
```

```
        CALL DELAY; gọi chương trình con tạo trễ DELAY
        RLCA; nhân nội dung A với 2
        JP LOOP; lặp lại từ vị trí 'LOOP'
DELAY: PUSH A; cất nội dung của A vào ngăn xếp
LD BC, 0FFFH
LOOP1:  DEC BC; giảm BC
        LD A, B
        OR C
        JP NZ, LOOP1; nếu BC > 0 thì nhảy về 'LOOP1'
        POP A; lấy lại nội dung của A từ ngăn xếp
        RET; trở về chương trình chính
        END
```

Chương 5

BOARD I/O AB-10MKII

1. Kết nối với máy tính

1.1. Cáp nối

Các tín hiệu được nối như bảng sau:

	Mã tín hiệu của I/O board		MP-80
	Màu dây	Mã tín hiệu	
1	Đỏ	+5V	+5V
2		+5V	+5V
3		GND	GND
4		GND	GND
5	Xanh		(AB15)
6		A7	AB7
7			(AB14)
8		A6	AB6
9			(AB13)
10	Xanh	A5	AB5
11			(AB12)
12		A4	AB4
13			(AB11)
14		A3	AB3
15	Xanh		(AB10)
16		A2	AB2
17			(AB9)
18		A1	AB1
19			(AB8)
20	Xanh	A0	AB0
21		D7	DB7
22			
23		D6	DB6
24			

25	Xanh	D5	DB5
26		IO/$\overline{M}$	
27		D4	DB4
28			$\overline{MEMR}$
29		D3	DB3
30	Xanh		$\overline{MEMW}$
31		D2	DB2
32			
33		D1	DB1
34			
35	Xanh	D0	DB0
36			
37			
38			
39		$\overline{INT}$	
40	Xanh		
41			
42			
43		$\overline{I/OR}$($\overline{RD}$)	$\overline{I/OR}$
44		RESET OUT	RESET OUT
45	Xanh		
46		$\overline{I/OW}$($\overline{WR}$)	$\overline{I/OW}$
47		GND	GND
48		GND	GND
49		GND	GND
50	Xanh	GND	GND

1.2. Cầu nối

Khi cầu nối được nối: ON

Khi cầu nối không được nối: OFF

1.2.1. Cầu nối A(JA)

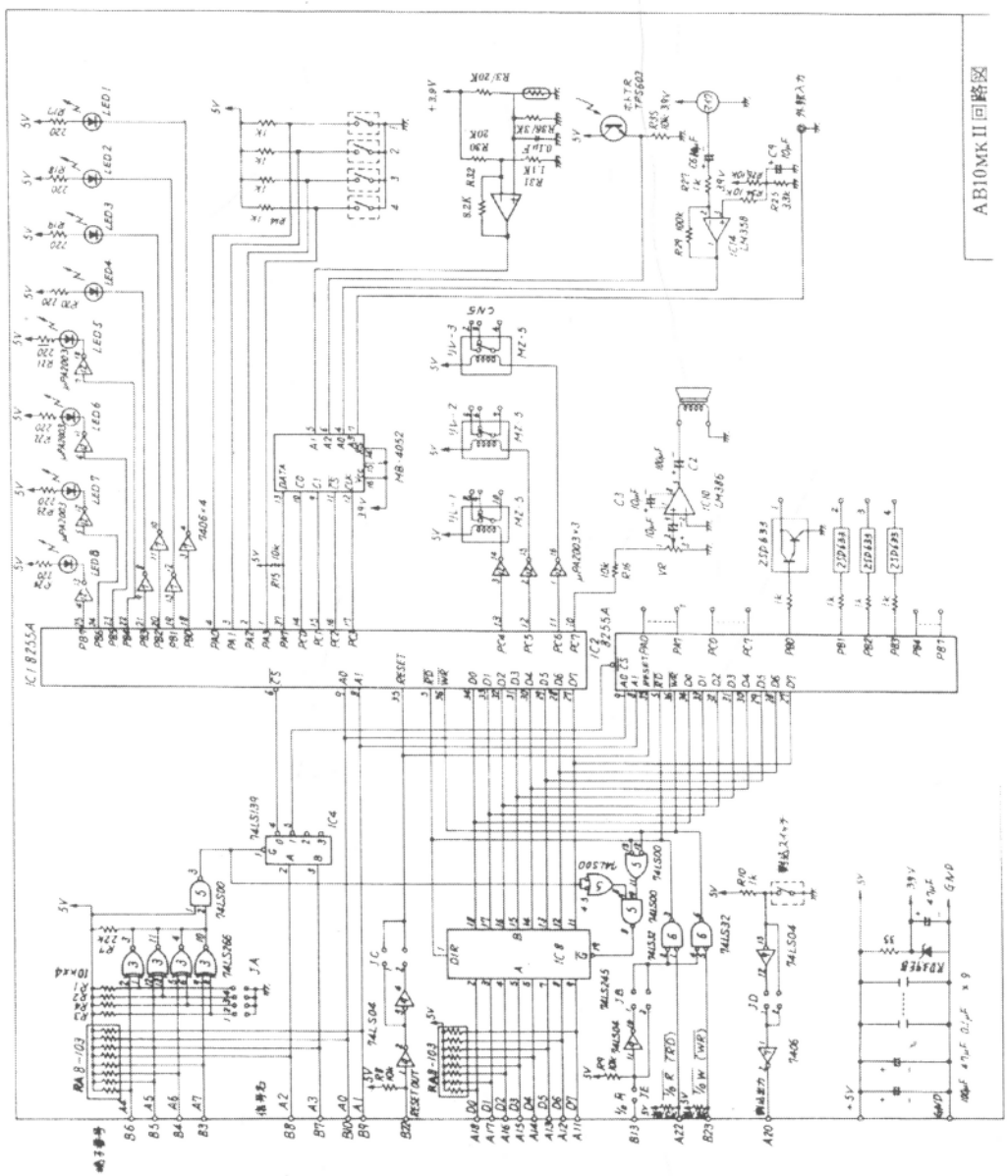
Địa chỉ I/O của 8255 (IC1 và IC2) trong board được thiết lập bởi cầu nối A.

Cầu nối A				Địa chỉ IC1	Địa chỉ IC2
1	2	3	4		
ON	ON	OFF	ON	20H-23H	24H-27H

1.2.2. Cầu nối B, C, D, E (JB, JC, JD, JE)

Các cầu nối được thiết lập như bảng dưới đây:

Vi xử lí	JB		JC		JD		JE
	1	2	1	2	1	2	
Z- 80	OFF	ON	ON	OFF	ON	OFF	ON



Hình 5.1. Board I/O AB – 10MKII

2. Giải thích phần cứng

2.1. Giao diện I/O 8255

8255 là một giao diện ngoại vi lập trình được (PPI), trong đó có 3 cổng vào ra 8 bit: PORTA, PORTB, PORTC. Có thể lựa chọn bất kì một trong 3 chế độ (Mode 0, Mode 1, Mode 2), bằng cách ghi vào thanh ghi điều khiển CWR.

* Mode 0:

- Có 2 cổng 8 bit và 2 cổng 4 bit.
- Có thể thiết lập các cổng này là vào hoặc ra.
- Cổng ra được chốt.
- Cổng vào có bộ đệm.

* Modc1:

- Được chia làm 2 nhóm A và B.
- Mỗi nhóm có thể thiết lập thành cổng dữ liệu 8 bit và cổng điều khiển 4 bit.
- Cổng dữ liệu 8 bit là cổng chốt mà nó được sử dụng làm cổng vào hoặc cổng ra.
- Cổng 4 bit được sử dụng như thiết bị điều khiển và trạng thái cho cổng dữ liệu.

* Mode 2:

- Chỉ sử dụng cho nhóm A.
- 8 bít của cổng A sử dụng cho vào ra dữ liệu và 5 bit của cổng C được sử dụng cho điều khiển.
- Cả hai chiều vào ra đều được chốt.

- Cổng 5 bit được sử dụng như thiết bị điều khiểnvà trạng thái cho cổng vào ra dữ liệu.

Chú ý: Cũng có thể sử dụng tổng hợp các chế độ trên.

IC 8255A của Board I/O được sử dụng như sau:

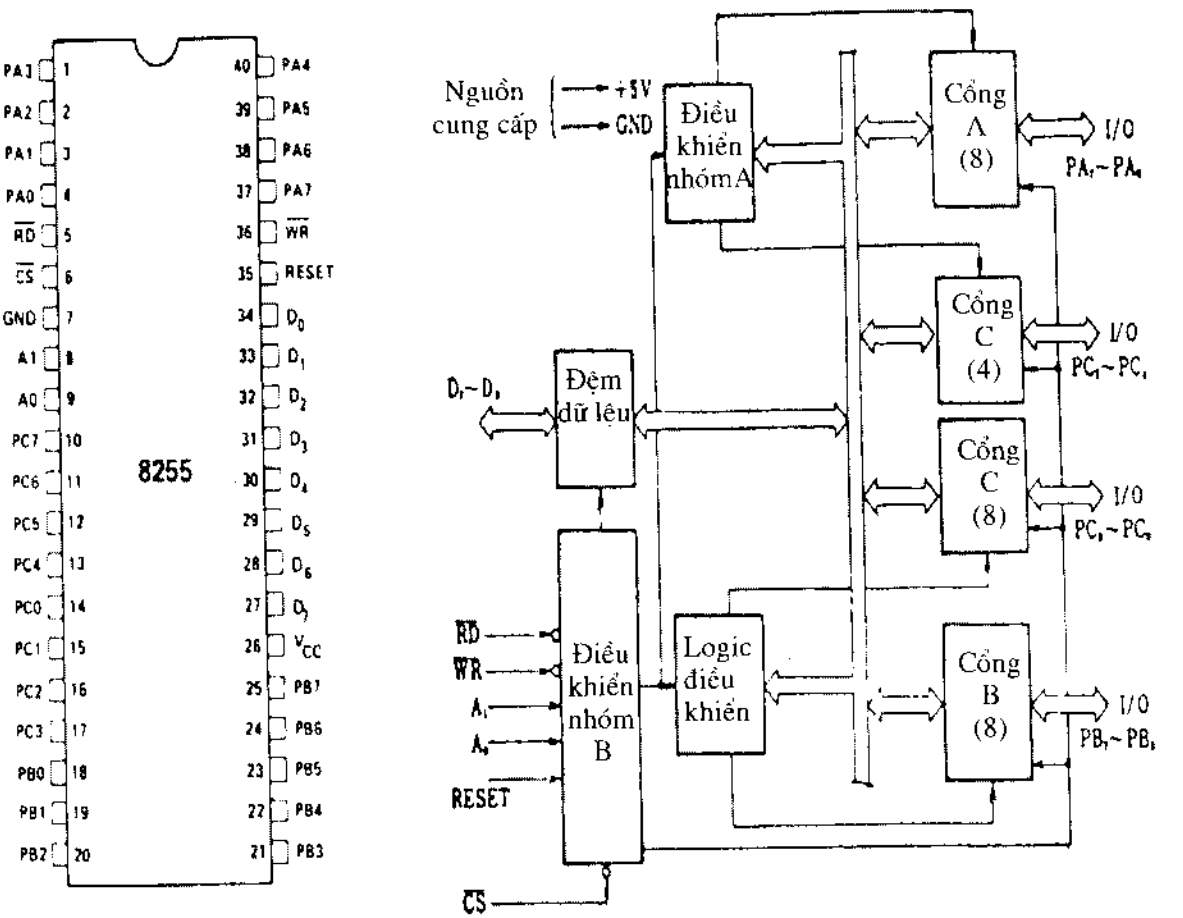
Mode = 0 PortB = OUTPUT

PortA = INPUT PortC = OUTPUT

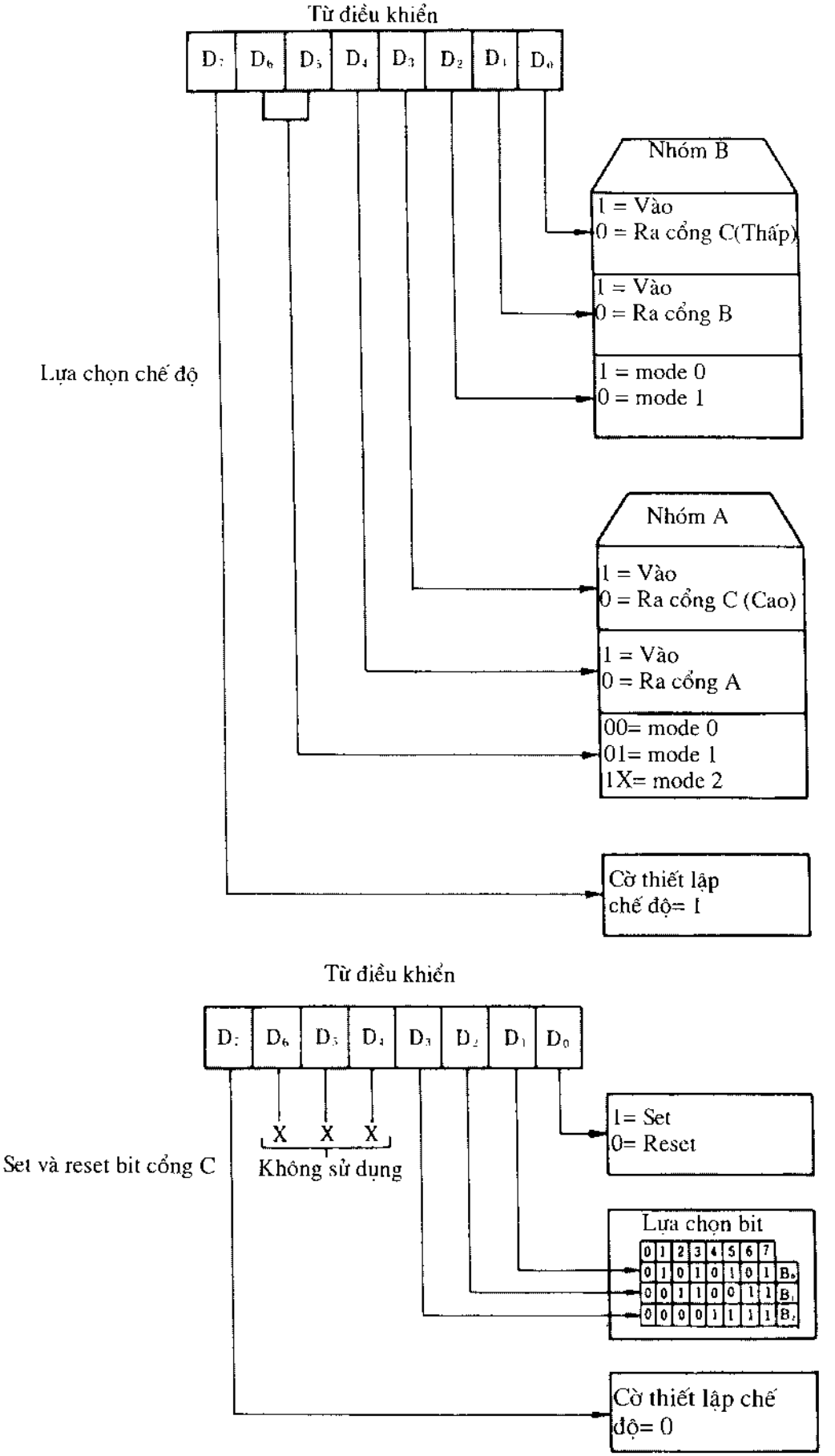
Từ điều khiển CWR sẽ được đọc như sau:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	0	0	1	0	0	0	0	= 90

Khi bít D_7 là "1", thì các bít khác của CWR dùng để xác định các chế độ và trạng thái của 8255A. Khi bit D_7 bằng "0", thì 4 bit của thanh ghi trạng thái CWR dùng để thiết lập bít và reset bit của cổng C.



Hình 5.2. Cấu trúc bên trong của IC8255



Hình 5.3. Chức năng của từ điều khiển cho 8255

2.2. Mạch giải mã địa chỉ

Địa chỉ vào ra của Board I/O được thiết lập tương ứng với JA (cầu nối A). Hình vẽ dưới đây trình bày sơ đồ mạch của 74LS 266, đầu ra (điểm A) trở thành mức thấp khi điều kiện của JA tương ứng với điều kiện của A_4 - A_7. Khi đầu vào A, B được nối tới A_2, A_3, chỉ có một khối đầu ra $\overline{Y0}$, $\overline{Y3}$ trở thành mức thấp "L" như trình bày trong bảng chức năng của IC 74LS139:

Vào			Ra			
G	A	B	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$
H	H hoặc L	H hoặc L	H	H	H	H
L	L	L	L	H	H	H
L	H	L	H	L	H	H
L	L	H	H	H	L	H
L	H	H	H	H	H	L

Bảng chức năng của IC74LS139

H: Mức điện áp cao

L: Mức điện áp thấp

2.3. Mạch điều khiển LED (Hình 5.4 và hình 5.5)

LED sẽ được sáng khi có dòng điện khoảng 5 – 20 mA chảy trong mạch. Như vậy nó không thể được điều khiển trực tiếp thông qua cổng 8255A, mà phải được khuếch đại dòng điện bởi IC μPA2003 hoặc 7406.

2.4. Mạch điều khiển Rơle (Hình 5.6)

Khi điều khiển dòng điện trên 100 mA thì cần phải có mạch điều khiển rơle, nó được khuếch đại bởi Transistor.

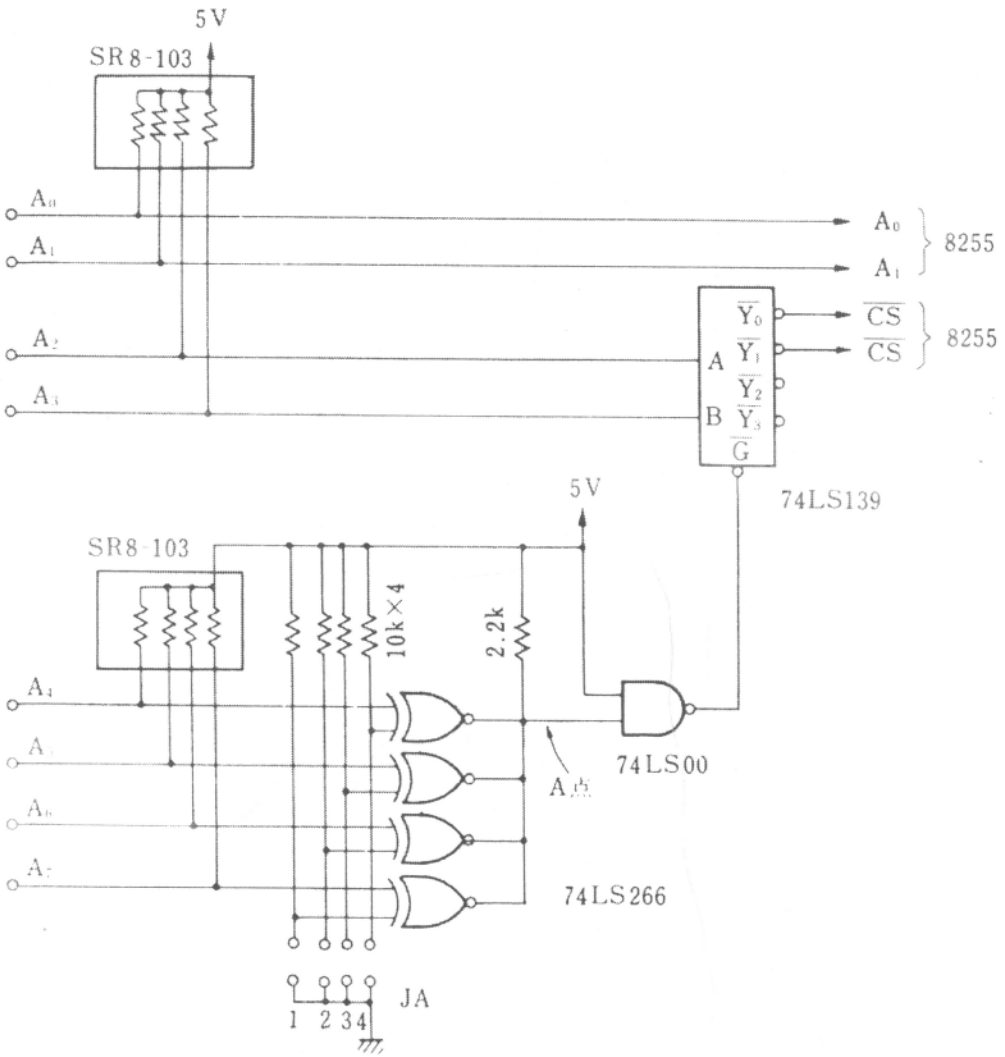
Để loại bỏ nguy hiểm của điện áp cao trong cuộn dây khi Transistor tắt, thì điôt cần được mắc song song với tải đầu ra như sơ đồ sau (Hình 5.6). Đối với IC μPA2003 thì không cần mắc thêm điôt bên ngoài bởi vì nó đã được mắc bên trong IC đó .

Rơle đặc biệt:

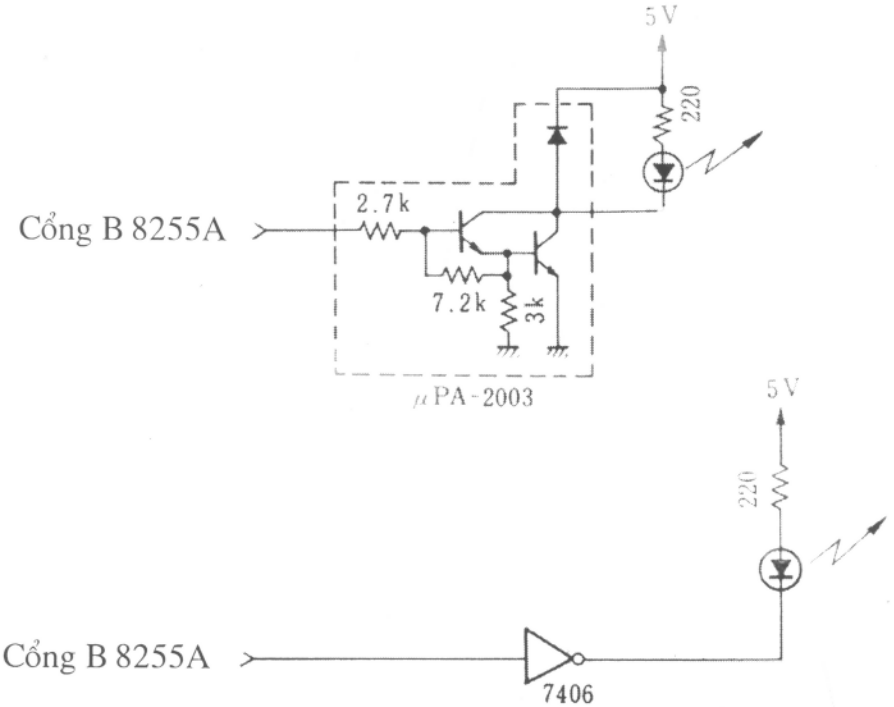
Điện áp, dòng điện ON/OFF....DC24V, 1A hoặc AC100V, 5A.

Trở kháng..... Dưới 100mΩ.

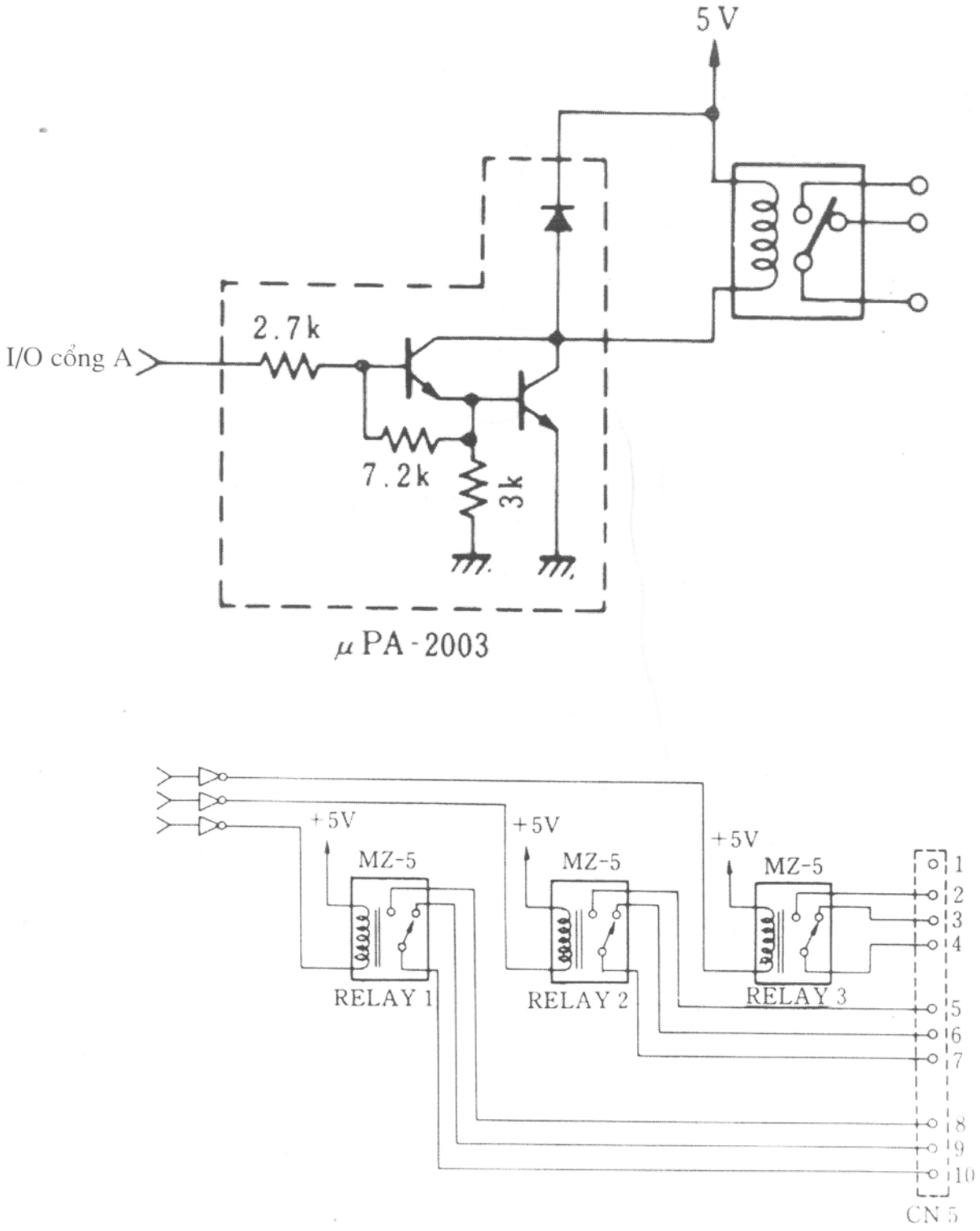
Giá trị dòng điện cực đại.....2A.



Hình 5.4. Mạch giải mã địa chỉ



Hình5.5. Mạch điều khiển LED



Hình 5.6. Mạch điều khiển rơle

2.5. Mạch điện chuyển mạch

Trạng thái của mỗi chuyển mạch được đọc bởi máy tính thông qua cổng vào IC8255A. Khi chuyển mạch ON thì điện áp trên I/O board ở mức cao "H", điện áp chuyển xuống mức thấp "L" khi chuyển mạch OFF.

Dạng tín hiệu khi chuyển mạch ON-OFF hoặc OFF-ON được trình bày như hình vẽ sau. Đó là khoảng thời gian không ổn định do vậy cần thiết phải đợi sau khoảng 10ms trước khi đọc trạng thái của chuyển mạch.

2.6. Mạch loa

Loa được điều khiển bởi mạch khuếch đại tín hiệu của PC7 với IC .LM386. Có thể điều chỉnh âm lượng của loa bằng biến trở 1kΩ.

2.7. Mạch điều khiển A/D

MB-4052 là IC biến đổi A/D 8 bit, với 4 đầu vào tương tự, 2 đầu vào điều khiển. Đầu vào RS được dùng để chọn mạch. Điện áp đầu vào tương tự nên để trong khoảng (0-1.95V) và (0-0,49V) khi RS thiết lập ở mức "H" và mức "L".

CS là chân lựa chọn mạch, bộ biến đổi A/D không được lựa chọn khi CS thiết lập ở mức cao "H" và bộ biến đổi A/D được lựa chọn khi CS được thiết lập ở mức thấp "L".

Dữ liệu sẽ được đọc đồng bộ với tín hiệu ADC.CLK. Tốc độ biến đổi là 1 bit/10 chu kì của ADC.CLK.

2.8. Mạch cảm biến

Các bộ cảm biến âm thanh (Microphon), nhiệt độ (Thermistor), và ánh sáng (Phototranssistor) được gắn trực tiếp trên I/O board. Đầu ra của Microphon được khuếch đại lên 100 lần bởi mạch khuếch đại đảo và được nối tới kênh 0 (A_0) của bộ biến đổi A/D

Thermistor được sử dụng trong bộ cảm biến nhiệt và đầu ra được nối tới kênh 1 (A_1) của bộ biến đổi A/D sau khi được khuếch đại bởi bộ khuếch đại vi sai. Khoảng nhiệt độ đo được là 0^0-30^0C.

Chú ý: Điện áp đầu vào của của bộ biến đổi A/D không được lớn hơn 4,4V (0-1,95V là trong khoảng tuyến tính).

2.9. Mạch điều khiển đa năng

Đây là chuyển mạch có dòng điện lớn sử dụng các Transistor được mắc theo kiểu mạch Darlington. Như hình vẽ dưới đây có 4 mạch được điều khiển bởi cổng B của IC 8255A. Mỗi mạch có thể điều khiển được dòng điện trên 1A.

3. Giải thích phần mềm

Tất cả chương trình ví dụ trong chương này đều bắt đầu ở địa chỉ 8000H với giả thiết Board vào/ra được nối tới Z80. Nếu nối tới Board vào/ra là các loại vi xử lí khác thì phải thay đổi địa chỉ tương ứng trong các tài liệu hướng dẫn sử dụng. Mạch điều khiển trên Board vào/ra tất cả được điều khiển bởi IC 8255A. Hình dưới đây trình bày các cổng vào/ra tương ứng với các mạch điều khiển.

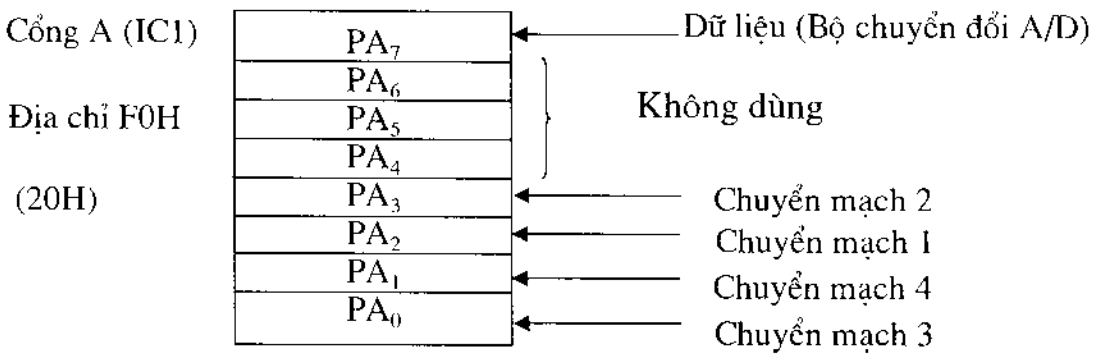
Địa chỉ của từng cổng được thiết lập như sau:

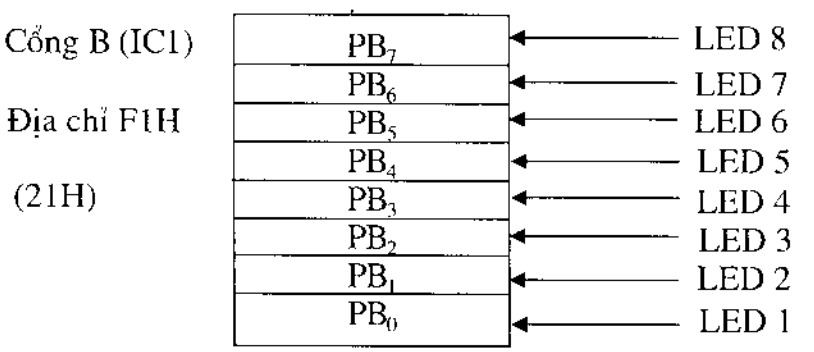
	Địa chỉ	MP- Z80
IC1 8255A Cổng A	F0H	20H
IC1 8255A Cổng B	F1H	21H
IC1 8255A Cổng C	F2H	22H
IC1 8255A Từ điều khiển	F3H	23H
IC1 8255A Cổng A	F4H	24H
IC1 8255A Cổng B	F5H	25H
IC1 8255A Cổng C	F6H	26H
IC1 8255A Từ điều khiển	F7H	27H

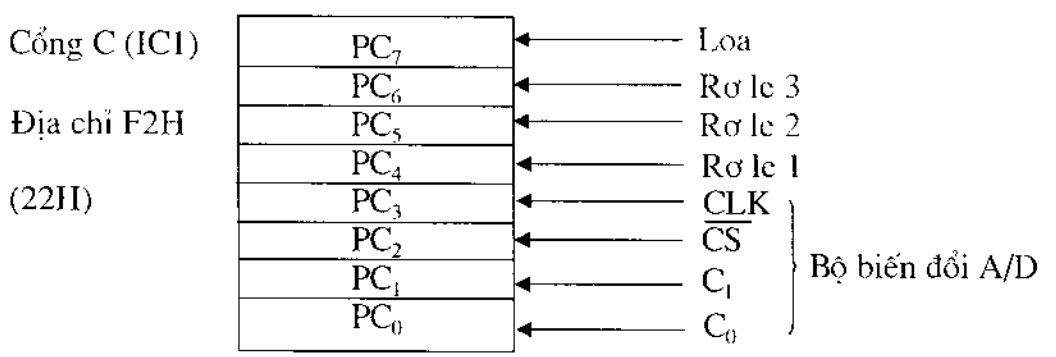
3.1. Khởi tạo cho 8255A

Từ điều khiển được thiết lập 10010000 = 90H, bởi vì cổng A là cổng vào, cổng B là cổng ra, cổng C là cổng ra, chế độ 0.

Các câu lệnh 'ORG', 'EQU' được sử dụng trong chương trình không có trong tập lệnh của vi xử lí, nhưng chúng được sử dụng giống như các lệnh cho ngôn ngữ Assembly. "ORG" chỉ ra địa chỉ bắt đầu nạp chương trình của ngôn ngữ máy, và EQU là lệnh gán giá trị số cho các nhãn. Ví dụ trong chương trình 1, giá trị số F2H được gán cho nhãn "Cổng C"







Chương trình1: Chương trình khởi tạo

```
0000                        ORG     8000H
8000
8000    3E 90               LD      A, 90       | Ghi (90H)
8002    D3 23               OUT     (CWR1), A   | cho từ điều
8004    D3 27               OUT     (CWR2), A   | khiển
8006    3E 04               LD      A, 04H      | Thiết lập CS
8008    D3 22               OUT     (PORTC1), A | của bộ A/D
800A    76                  HALT
800B
0022            (PORTC1)    EQU     22H
0023            CWR1        EQU     23H
0027            CWR2        EQU     27H
800B
800B                        END
```

Chương trình 2: Điều khiển LED

```
0000                        ORG     8000H
8000
8000    3E 90               LD      A ,90
8002    D3 23               OUT     (CWR1), A
8004    D3 27               OUT     (CWR2), A
8006    3E 04               LD      A, 04H
8008    D3 22               OUT     (PORTC1), A
800A
800A    3E55                LD      A, 55H      Đưa tới cổng B
800C    D3 21               OUT     (PORTB1), A 55H(01010101B)
800E    76                  HALT
800F
0021            PORTB1      EQU     21H
0022            PORTC1      EQU     22H
0023            CWR1        EQU     23H
0027            CWR2        EQU     27H
800F
800F                        END
```

Giống chương trình 1

Chương trình3: Điều khiển Rơle

```
0000                        ORG     8001H
8000
8000    3E 90               LD      A ,90
8002    D3 23               OUT     (CWR1), A
8004    D3 27               OUT     (CWR2), A
8006    3E 74               LD      A, 74H
8008    D3 22               OUT     (PORTC1), A
800A    76                  HALT
800B
0022            PORTC1      EQU     22H
0023            CWR1        EQU     23H
0027            CWR2        EQU     27H
800B
800B                        END
```

3.2. Đọc trạng thái chuyển mạch

Các chuyển mạch được nối tới PA0 – PA3 của cổng A, vị trí nào là "1" có nghĩa là chuyển mạch không được ấn và "0" có nghĩa là chuyển mạch được ấn. Bởi vì 3 bit cao (PA4 – PA6) của cổng A không được nối, do đó dữ liệu không thể đưa vào 3 bit cao. Vì vậy câu lệnh "AND 0FH" để xoá 4 bit cao về "0". Khi chuyển mạch được nhấn thì LED tương ứng sẽ tắt.

Chương trình 4:

```
0000                        ORG     8000H
8000
8000    3E 90               LD      A ,90H
8002    D3 23               OUT     (CWR1), A
8004    D3 27               OUT     (CWR2), A
8006    3E  04              LD      A, 04H
8008    D3 22               OUT     (PORTC1), A
800A
800A    DB 20       KEYIN   IN      A, (PORTA1)     ;Đặt trạng thái
                                                    chuyển mạch
```

```
800C  E6 0F          AND   0FH          ;Xoá 4bit cao
800E  D3 21          OUT   (PORTB1), A  ;Đưa ra LED
8010  C3 0A 80       JP    KEYIN
8013
0020          PORTA1 EQU   20H
0021          PORTB1 EQU   21H
0022          PORTC1 EQU   22H
0023          CWR1   EQU   23H
0027          CWR2   EQU   27H
8013
8013                 END
```

3.3. Điều khiển loa

Âm thanh có thể được tạo ra bởi việc gửi các tín hiệu "1" và "0" tới chân PC7 của cổng C.

Chương trình 5:

```
0000                 ORG   8000H
8000
8000  3E 90          LD    A, 90H
8002  D3 23          OUT   (CWR1), A
8004  D3 27          OUT   (CWR2), A
8006  3E 04          LD    A, 04H
8008  D3 22          OUT   (PORTC1), A
800A
800A  3E 0F    SOUND: LD    A, 0FH      Thiết lập bit 7
                                        của cổng C lên
                                        mức cao
800C  CD 17 80       CALL  OUTPC7       Thiết lập bit 7 của
                                        cổng C xuống
                                        mức thấp
800F  3E 0E          LD    A, 0EH
8011  CD 17 80       CALL  OUTPC7
8014  C3 0A 80       JP    SOUND
8017
```

```
8017   D3 23       OUTPC7:   OUT    (CWR1), A
8019   3E FA                 LD     A, 250
801B   00          DELAY:    NOP
801C   3D                    DEC    A
801D   C2 1B 80               JP     NZ, DELAY
8020   C9                     RET
8021
0022                PORTC1    EQU    22H
0023                CWR1      EQU    23H
0027                CWR2      EQU    27H
8021
8021                          END
```

3.4. Điều khiển bộ biến đổi A/D

Dây tín hiệu của bộ biến đổi A/D (MB4052) được nối như hình vẽ 4.14 và hình 5.1. Trạng thái của C_0 và C_1 sẽ xác định bốn cổng vào tương tự:

C_0	C_1	Tín hiệu điều khiển
0	0	Microphon
0	1	Cảm biến nhiệt
1	0	Đèn
1	1	Đầu vào mở rộng

Dữ liệu được biến đổi đồng bộ với xung nhịp của bộ biến đổi (ADC.CLK).

Chương trình 6:

Dòng lệnh điều khiển được trình bày từ dòng thứ 14 tới dòng thứ 58 của chương trình. Trong dòng lệnh điều khiển, lựa chọn kênh vào tương tự được thực hiện bởi việc qui định dữ liệu (0-3) của kênh biến đổi ở hai bit thấp của thanh ghi C.

Kênh vào tương tự sẽ được biến đổi thành thông tin số và được lưu trữ trong địa chỉ 8210H trước khi chương trình dừng

Chương trình 6:

```
0000                    ORG     8000H
8000
8000  3E 90             LD      A ,90H
8002  D3 23             OUT     (CWR1), A
8004  D3 27             OUT     (CWR2), A
8006  3E 04             LD      A, 04H
8008  D3 22             OUT     (PORTC1), A
800A
800A  3A 00 82           LD      A, (8200H)
800D  4F                 LD      C, A
800E  CD 00 81           CALL    ADCNV
8011  32 10 82           LD      (8210H), A
8014  76                 HALT
8015
8015                    ORG     8100H
8100
8100  C5        ADCNV:   PUSH    BC
8101  D5                 PUSH    DE
8102
8102  79                 LD      A, C
8103  E6 03              AND     03H
8105  4F                 LD      C, A
8106  DB 22              IN      A, (PORTC1)
8108  E6 F0              AND     F0H
810A  Bi                 OR      C
810B  EE 04              XOR     04H
810D  D3 22              OUT     (PORTC1), A
810F  EE 04              XOR     04H
8111  D3 22              OUT     (PORTC1), A
8113
8113  0E 09              LD      C, 9H
8115  11 06 07           LD      DE, 0706H
8118  7A        ADC1:    LD      A, D
8119  D3 23              OUT     (CWR1), A
811B  7B                 LD      A, E
```

Xoá 6 bit cao của thanh ghi C về "0"

Thiết lập 2 bít thấp của cổng C thành kênh dữ liệu

$\overline{CS}$ = "H"

$\overline{CS}$ = "L"

Đếm vòng

CLK= "H"

CLK= "l"

```
811C  D3 23          OUT   (CWR1), A
811E  DB 20          IN    A, (PORTA1)
8120  17             RLA                 Dịch trái dữ liệu
8121  78             LD    A, B          trong thanh ghi B 1
8122  17             RLA                 bit và thiết lập dữ liệu
8123  47             LD    A, B          của bộ chuyển đổi
                                         A/D tới bít thấp nhất
                                         của thanh ghi B
8124  0D             DEC   C
8125  C2 18 81       JP    NZ, ADC1
8128
8128  3E 05          LD    A, 05H        CS = "H"
812A  D3 23          OUT   (CWR1), A
812C  78             LD    A, B
812D
812D  D1             POP   DE
812E  C1             POP   BC
812F  C9             RET
8130
0020        PORTA1   EQU   20H
0022        PORTC1   EQU   22H
0023        CWR1     EQU   23H
0027        CWR2     EQU   27H
8130
8130                 END
```

Các thông số vào/ra của chương trình 6:

Thông số vào: Kênh vào tương tự (2 bit thấp của thanh ghi C)

Thông số ra: Dữ liệu ra (thanh ghi A)

Các thanh ghi khác: Thanh ghi A, cờ.

3.5. Điều khiển ngắt

Trong chương trình 7, hoạt động của chương trình điều khiển loa được thực hiện bởi vòng lặp. Khi ngắt được thực hiện, thì chương trình sẽ nhảy tới thủ tục ngắt. Trong thủ tục ngắt, rơle 1 sẽ ON và OFF hai lần và chương trình sẽ quay trở lại chương trình chính.

Chương trình 7:

```
0000                        ORG     8000H
8000
8000  3E 90                 LD      A, 90H
8002  D3 23                 OUT     (CWR1), A
8004  D3 27                 OUT     (CWR2), A
8006  3E 04                 LD      A, 04H
8008  D3 22                 OUT     (PORTC1), A
800A
800A  3E ED                 LD      A, 0EDH
800C  D3 13                 OUT     (CTC_CH3), A      Khởi động cho bộ
800E  3E 01                 LD      A, 1              đếm Z 80
8010  D3 13                 OUT     (CTC_CH3), A      chế độ đếm
8012  AF                    XOR     A                 Số bộ đếm 1
8013  D3 10                 OUT     (CTC_CH0), A
8015
8015  3E 82                 LD      A, 82H
8017  ED 47                 LD      I, A
8019  ED 5E                 IM      2                 Ngắt chế độ 2
801B  FB                    EI                        Cho phép ngắt
801C
801C  3E 0F       SOUND:    LD      A, 0FH
801E  CD 29 80              CALL    PUTPC7
8021  3E 0E                 LD      A, 0EH
8023  CD 29 80              CALL    OUTPC7
8026  C3 1C 80              JP      SOUND
8029
8029  D3 23       OUTPC7:   OUT     (CWR1), A
802B  3E FA                 LD      A, 250            Giống chương
802D  00          DELAY:    NOP                       trình điều khiển
802E  3D                    DEC     A                 loa
802F  C2 2D 80              JP      NZ, DELAY
8032  C9                    RET
8033
8033              INT:
8033  F5                    PUSH    AF                Thủ tục ngắt
```

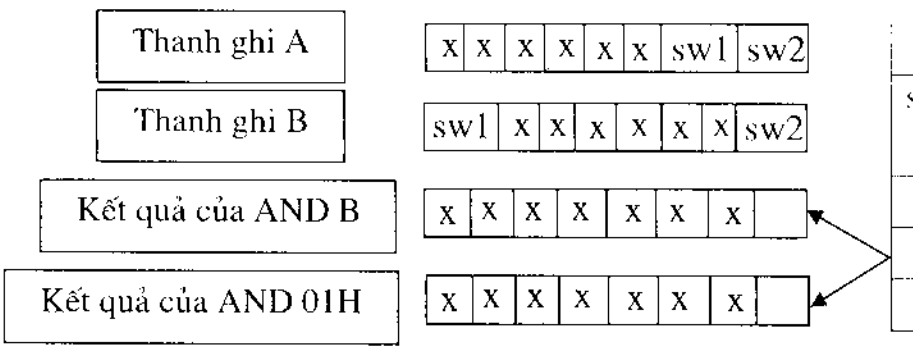
```
8034  E5              PUSH  HL
8035
8035  3E 09           LD    A, 09H        Rơle_ON
8037  CD 4E 80        CALL  INT5
803A  3E 08           LD    A, 08H        Rơle_OFF
803C  CD 4E 80        CALL  INT5
803F  3E 09           LD    A, 09H        Rơle_ ON
8041  CD 4E 80        CALL  INT5
8044  3E 08           LD    A, 08H        Rơle_OFF
8046  CD 4E 80        CALL  INT5
8049
8049  E1              POP   HL
804A  F1              POP   AF
804B  FB              EI
804C  ED 4D           RETI
804E
804E  D3 23     INT5: OUT   (CWR1), A
8050  21 00 00        LD    HL, 0000H
8053            INT6: NOP
8053  2B              DEC   HL            Trễ
8054  7C              LD    A, H
8055  B5              OR    L
8056  C2 53 80        JP    NZ, INT6
8059  C9              RET
805A
805A                  ORG   8200H
8200
8200  00 00           DW    0
8202  00 00           DW    0
8204  00 00           DW    0
8206  33 80           DW    INT
8208
8208
0022  PORTC1          EQU   22H
0023  CWR1            EQU   23H
0027  CWR2            EQU   27H
```

```
0010    CTC_CH0          EQU     0010H
0013    CTC_CH3          EQU     0013
8208
8208                     END
```

3.6. Một số chương trình ứng dụng

3.6.1. Lệnh "AND"

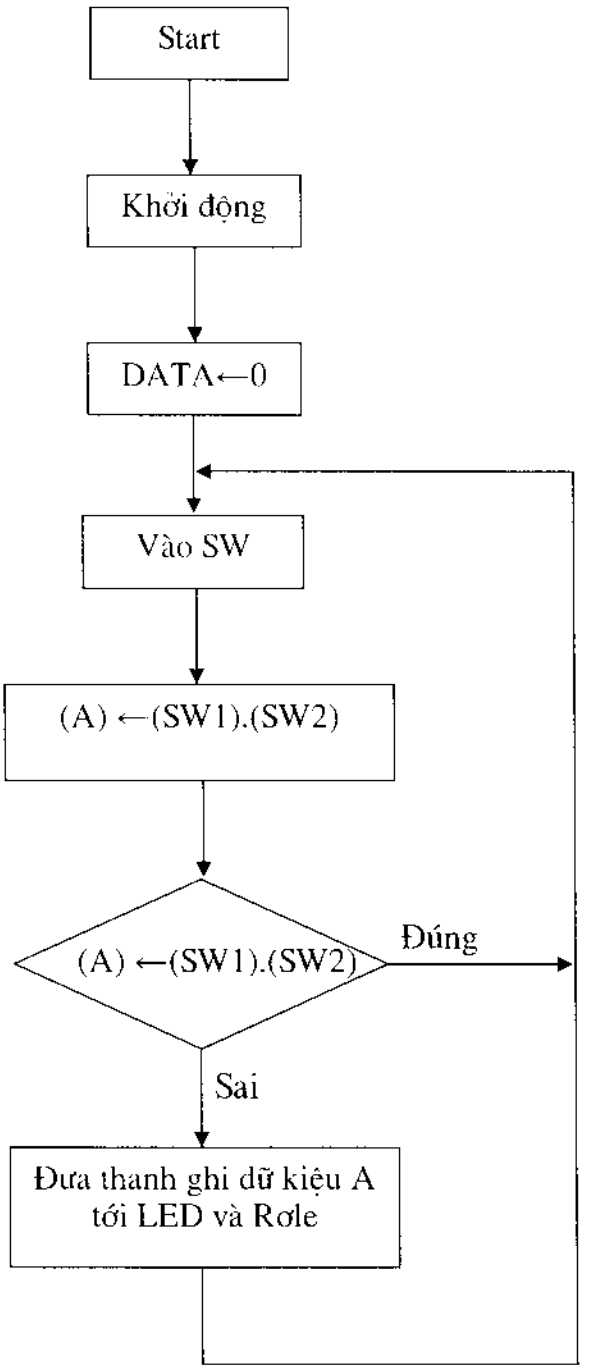
Đây là chương trình trình bày ví dụ về hoạt động của lệnh "AND". Trong chương trình này, khi nhấn cả hai chuyển mạch Switch1 và Switch2, thì LED_1 và Rơle_1 sẽ ON.



Vào		Ra
sw2	sw1	
0	0	0
0	1	0
1	0	0
1	1	1

Bảng chức năng của lệnh "AND"

Lưu đồ thuật toán:



Chương trình 8:

```
0000                    ORG     8000H
8000
8000    3E 90           LD      A, 90H
8002    D3 23           OUT     (CWR1), A
```

```
8004    D3 27           OUT     (CWR2), A
8006    3E 04           LD      A, 04H
8008    D3 22           OUT     (PORTC1), A
800A
800A    21 00 8D        LD      HL, DATA
800D    36 00           LD      (HL), 0
800F
800F    DB 20   _AND:   IN      A, (PORTA1)
8011    2F              CPL
8012    47              LD      B, A
8013    0F              RRCA
8014    A0              AND     B
8015    E6 01           AND     01H
8017
8017    BE              CP      (HL)
8018    CA 0F 80        JP      Z, _AND
801B    77              LD      (HL), A
801C
801C    D3  21          OUT     (PORTB1), A
801E    F6 08           OR      08H
8020    D3 23           OUT     (CWR1), A
8022    C3 0F 80        JP      _AND
8025
8D00            DATA    EQU     8D00H
8025
0020            PORTA1  EQU     20H
0021            PORTB1  EQU     21H
0022            PORTC1  EQU     22H
0023            CWR1    EQU     23H
0027            CWR2    EQU     27H
8025
8025                    END
```

3.6.2. Bảng chức năng và chương trình của lệnh "OR"

Hoạt động của lệnh "OR" được thực hiện trong 16 dòng lệnh từ dòng 12 tới dòng 28 của chương trình.

Vào		Ra
SW2	SW1	
0	0	0
0	1	1
1	0	1
1	1	1

Bảng chức năng của lệnh "OR"

Chương trình 9:

```
0000                            ORG     8000H
8000
8000    3E 90                   LD      A, 90H
8002    D3 23                   OUT     (CWR1), A
8004    D3 27                   OUT     (CWR2), A
8006    3E 04                   LD      A, 04H
8008    D3 22                   OUT     (PORTC1), A
800A
800A    21 00 8D                LD      HL, DATA
800D    36 00                   LD      (HL), 0
800F
800F    DB 20           _OR:    IN      A, (PORTA1)
8011    2F                      CPL
8012    47                      LD      B, A
8013    0F                      RRCA
8014    A0                      OR      B
8015    E6 01                   AND     01H
8017
8017    BE                      CP      (HL)
8018    CA 0F 80                JP      Z, _OR
801B    77                      LD      (HL), A
```

```
801C
801C    D3  21                  OUT     (PORTB1), A
801E    F6 08                   OR      08H
8020    D3 23                   OUT     (CWR1), A
8022    C3 0F 80                JP      _OR
8025
8D00            DATA    EQU     8D00H
8025
0020            PORTA1  EQU     20H
0021            PORTB1  EQU     21H
0022            PORTC1  EQU     22H
0023            CWR1    EQU     23H
0027            CWR2    EQU     27H
8025
8025                    END
```

3.6.3. Lệnh "XOR" (exclusive "OR")

Bảng chức năng và chương trình của lệnh "XOR" được trình bày như sau:

Vào		Ra
SW2	SW1	
0	0	0
0	1	1
1	0	1
1	1	0

Bảng chức năng của lệnh "XOR"

Chương trình 10:

```
0000                    ORG     8000H
8000
8000    3E 90           LD      A ,90H
8002    D3 23           OUT     (CWR1), A
8004    D3 27           OUT     (CWR2), A
```

```
8006    3E 04           LD      A, 04H
8008    D3 22           OUT     (PORTC1), A
800A
800A    21 00 8D        LD      HL, DATA
800D    36 00           LD      (HL), 0
800F
800F    DB 20   _OR:    IN      A, (PORTA1)
8011    2F              CPL
8012    47              LD      B, A
8013    0F              RRCA
8014    A0              XOR     B
8015    E6 01           AND     01H
8017
8017    BE              CP      (HL)
8018    CA 0F 80        JP      Z, _OR
801B    77              LD      (HL), A
801C
801C    D3  21          OUT     (PORTB1), A
801E    F6 08           OR      08H
8020    D3 23           OUT     (CWR1), A
8022    C3 0F 80        JP      _OR
8025
8D00            DATA    EQU     8D00H
8025
0020            PORTA1  EQU     20H
0021            PORTB1  EQU     21H
0022            PORTC1  EQU     22H
0023            CWR1    EQU     23H
0027            CWR2    EQU     27H
8025
8025                    END
```

3.6.4. Bộ đếm (1)

Số lần nhấn chuyển mạch SW1 sẽ được đếm trong mã nhị phân và được hiển thị trên các đèn LED. Trong chương trình này, SW1 tác động được bằng tay sẽ có vấn đề về khả năng đáp ứng, vì vậy điều kiện ON – OFF của các công tắc sẽ được đọc lại sau khi đợi 20 ms chuyển mạch được tác động.

Chương trình 11:

0000			ORG	8000H	
8000					
8000	3E 90		LD	A, 90H	
8002	D3 23		OUT	(CWR1), A	
8004	D3 27		OUT	(CWR2), A	
8006	3E 04		LD	A, 04H	
8008	D3 22		OUT	(PORTC1), A	
800A					
800A	0E 00		LD	C, 0	Xoá thanh ghi C về 0
800C					
800C	DB 20	COUNT:	IN	A, (PORTA1)	Đợi chuyển mạch SW1 ON
800E	E6 01		AND	01H	
8010	C2 0C 80		JP	NZ, COUNT	
8013					
8013	0C		INC	C	Tăng thanh ghi C lên 1 đơn vị
8014	79		LD	A, C	Đưa giá trị đếm ra LED
8015	D3 21		OUT	(PORTB1), A	
8017					
8017	CD 27 80		CALL	DELAY	
801A					
801A	DB 20	COUNT1:	IN	A, (PORTA1)	Đợi chuyển mạch SW1 OFF
801C	E6 01		AND	01H	
801E	CA 1A 80		JP	Z, CUNT1	

```
8021    CD 27 80            CALL    DELAY
8024    C3 0C 80            JP      COUNT
8027
8027    11 00 10    DELAY:  LD      DE, 1000H
802A    1B          DELAY1: DEC     DE
802B    7A                  LD      A, D            Trễ 20ms
802C    B3                  OR      E
802D    C2 2A 80            JP      NZ, DELAY1
8030    C9                  RET
8031
0020                PORTA1  EQU     20H
0021                PORTB1  EQU     21H
0022                PORTC1  EQU     22H
0023                CWR1    EQU     23H
0027                CWR2    EQU     27H
8031
8031                        END
```

3.6.5 Bộ đếm (2)

Đây là chương trình được sửa lại từ chương trình trước. Trong chương trình này bộ đếm làm việc với mã BCD (Binary Coded Decimal) và được hiển thị trên các LED.

Chương trình 12:

```
0000                        ORG     8000H
8000
8000    3E 90               LD      A, 90H
8002    D3 23               OUT     (CWR1), A
8004    D3 27               OUT     (CWR2), A
8006    3E 04               LD      A, 04H
8008    D3 22               OUT     (PORTC1), A
800A
800A    0E 00               LD      C, 0
```

```
800C
800C  DB 20      COUNT:  IN    A, (PORTA1)
800E  E6 01              AND   01H
8010  C2 0C 80          JP    NZ, COUNT
8013
8013  79                 LD    A, C
8014  3C                 INC   A
8015  27                 DAA
8016  4F                 LD    C, A
8017  D3 21              OUT   (PORTB1), A
8019
8019  CD 29 80          CALL  DELAY
801C
801C  DB 20      COUNT1: IN    A, (PORTA1)
801E  E6 01              AND   01H
8020  CA 1A 80           JP    Z, CUNT1
8023  CD 27 80           CALL  DELAY
8026  C3 0C 80           JP    COUNT
8029
8029  11 00 10   DELAY:  LD    DE, 1000H
802C  1B         DELAY1: DEC   DE
802D  7A                 LD    A, D
802E  B3                 OR    E
802F  C2 2A 80           JP    NZ, DELAY1
8032  C9                 RET
8033
0020             PORTA1  EQU   20H
0021             PORTB1  EQU   21H
0022             PORTC1  EQU   22H
0023             CWR1    EQU   23H
0027             CWR2    EQU   27H
8033
8033                     END
```

3.6.6. Đầu vào chuyển mạch

Đây là chương trình, mà LED tương ứng với một trong bốn chuyển mạch (SW1- SW4) sẽ được bật ON.

Chương trình 13:

```
0000                        ORG   8000H
8000
8000  3E 90                 LD    A, 90H
8002  D3 23                 OUT   (CWR1), A
8004  D3 27                 OUT   (CWR2), A
8006  3E 04                 LD    A, 04H
8008  D3 22                 OUT   (PORTC1), A
800A
800A  DB 20       SW:       IN    A, (PORTA1)
800C  EE FF                 XOR   0FFH          Đợi chuyển
800E  E6 0F                 AND   0F            mạch bật ON
8010  CA 0A 80              JP    Z, SW
8013
8013  D3 21                 OUT   (PORTB1), A
8015

8015  CD 24 80    SW1:      CALL  DELAY
8018  DB 20                 IN    A, (PORTA1)
801A  EE FF                 XOR   0FFH          Đợi chuyển
801C  E6 0F                 AND   0F            mạch OFF
801E  C2 15 80              JP    NZ, SW
8021  C3 0A 80              JP    SW
8024
8024  11 00 10    DELAY:    LD    DE, 1000H
8027  1B          DELAY1:   DEC   DE
8028  7A                    LD    A, D
8029  B3                    OR    E
802A  C2 27 80              JP    NZ, DELAY1
802D  C9                    RET
```

```
802E
0020            PORTA1  EQU   20H
0021            PORTB1  EQU   21H
0022            PORTC1  EQU   22H
0023            CWR1    EQU   23H
0027            CWR2    EQU   27H
802E
802E                    END
```

3.6.7. Tạo nhạc tự động

Trong yêu cầu để tạo ra âm thanh với vi xử lí, chúng ta cung cấp dữ liệu của giai điệu và thời gian thông qua chương trình, sau đó máy tính sẽ tạo ra các rung động điện, được khuếch đại và đưa ra LED thông qua loa.

+ Bảng thông số giai điệu:

Cấp độ	Thông số giai điệu	Cấp độ	Thông số giai điệu
DO	AC (HEXA)		2F (HEXA)
	A2	TI	2C
RE	99	DO	2A
	90		27
ME	88	RE	25
FAH	80		23
	79	ME	21
SO	72	FAH	1F
	6C		1D
LA	65	SO	1B
	60		1A
TI	5A	LA	18
DO	55		17
	50	TI	15
RE	4C	DO	14
	47		13
ME	43	RE	12
FAH	3F		11

	3C	ME	10
SO	38	FAH	0F
	35		0D
LA	32		

Chú ý: Thiết lập thông số của giai điệu là FF khi tạo khoảng ngưng giữa hai nốt

+ Bảng thông số thời gian

Nốt	Thời gian	Khoảng ngưng	Thời gian
	10		10
	0C		0C
	08		08
	06		06
	04		04
	03		03
	02		02
	01		01

Chương trình 14:

Phương pháp vào dữ liệu nốt nhạc: Đầu tiên thiết lập thông số giai điệu nhạc theo sau là thời gian và chúng được ghi sau địa chỉ 8D00H. Dữ liệu "0" ở cuối nốt thì sẽ lập lại bản nhạc.

```
0000                        ORG     8000H
8000
8000    3E 90               LD      A, 90H
8002    D3 23               OUT     (CWR1) , A
8004    D3 27               OUT     (CWR2), A
8006    3E 04               LD      A, 04H
8008    D3 22               OUT     (PORTC1), A
800A
800A    21 00 8D    MAIN:   LD      HL, MUSIC
800D    7E          MAIN1:  LD      A, (HL)     Thiết lập địa chỉ nốt
800E    46                  LD      B, (HL)     nhạc
                                                Gán giá trị nốt nhạc
800F    23                  INC     HL
```

```
8010  4E           LD      C, (HL)
8011  A7           AND     A           Gán thời gian
8012  CA 0A 80     JP      Z, MAIN     Nếu nhạc kết thúc
                                       thì nhảy tới MAIN
8015
8015  E5     MAIN2: PUSH    HL
8016  CD 22 80       CALL    SOUND
8019  E1             POP     HL
801A  0D             DEC     C
801B  C2 15 80       JP      NZ, MAIN2
801E  23             INC     HL
801F  C3 0D 80       JP      MAIN1
8022
8022  21 FF 33 SOUND:  LD     HL, 33FFH
8025  50       SOUND1: LD     D, B
8026  78               LD     A, B
8027  3C               INC    A
8028  CA 31 80         JP     Z, SOUND2   Dữ kiệu cấp độ
                                         hoặc
                                         ngưng(FFH)
802B
802B  3E 0F            LD     A, 0FH
802D  CD 39 80         CALL   COUNT
8030  50               LD     D, B
8031
8031  3E 0E  SOUND2:  LD     A, 0EH
8033  CD 39 80         CALL   COUNT
8036  C3 25 80         JP     SOUND1
8039
8039  D3 23  COUNT:   OUT    (CWR1), A
803B  2B       COUNT1: DEC    HL
803C  7C               LD     A, H
803D  A7               AND    A
803E  CA 46 80         JP     Z, COUNT2
8041  15               DEC    D
8042  C2 3B 80         JP     NZ, COUNT2
```

```
8045    C9              RET
8046
8046    D1      COUNT2: POP     DE
8047    C9              RET
8048
0022            PORTC1  EQU     22H
0023            CWR1    EQU     23H
0027            CWR2    EQU     27
8048
8048                    ORG     8D00H
8D00
8D00    21 04 21 02     DB      21H, 4, 21H, 2, 25H, 2      Dữ liệu giai điệu
8D04    25 02
8D06    2A 02 25 02     DB      2AH, 2, 25H, 2, 2AH, 2
8D0A    2A 02
8D0C    2C 02 32 04     DB      2CH, 2, 32H, 4, 32H, 4
8D10    32 04
8D12    32 08 1F 04     DB      32H, 8, 1FH, 4, 1FH, 2
8D16    1F 02
8D18    21 02 25 02     DB      21H, 2, 25H, 2, 2AH, 2
8D1C    2A 02
8D1E    25 02 1F 02     DB      25H, 2, 1FH, 2, 21H, 16
8D22    21 10
8D24    1F 04 1F 02     DB      1FH, 4, 1FH, 2, 21H, 2
8D28    21 02
8D2A    25 04 25 02     DB      25H, 4, 25H, 2, 1FH, 2
8D2E    1F 02
8D30    21 04 21 02     DB      21H, 4, 21H, 2, 2AH, 2
8D34    2A 02
8D36    32 08 2C 04     DB      32H, 8, 2CH, 4, 21H, 4
8D3A    21 04
8D3C    25 02 2A 02     DB      25H, 2, 2AH, 2, 2CH, 2
8D40    2C 02
```

```
8D42    2A 02 32 10          DB      2AH, 2, 32H, 16, FFH, 4
8D46    FF 04
8D48    00                   DB      00
8D49                         END
```

3.6.8. Vonmet số

Chương trình 15:

```
0000                                 ORG     8000H
8000
8000    3E 90                        LD      A, 90H
8002    D3 23                        OUT     (CWR1), A
8004    D3 27                        OUT     (CWR2), A
8006    3E 04                        LD      A, 04H
8008    D3 22                        OUT     (PORTC1), A
800A
800A    CD 69 00                     CALL    CLS
800D    0E 00                        LD      C, 0
800F    ;
800F    ;MAIN
800F                    ;
800F    CD 8D 80        MAIN1:       CALL    SWIN
8012    D2 16 80                     JP      NC,MAIN2
8015    4F                           LD      C, A
8016    CD 34 80        MAIN2:       CALL    ADCONV
8019    D3 21                        OUT     (PORTB1), A
801B    C5                           PUSH    BC
801C    1E 76                        LD      E,76H
801E    CD 60 80                     CALL    MPY
8021    CD 71 80                     CALL    BCD
8024    62                           LD      H,D
8025    68                           LD      L,B
8026    CD 57 00                     CALL    DHEX4
8029    CD A5 80                     CALL    DELAY
802C    C1                           POP     BC
```

```
802D    AF                      XOR     A
802E    CD 5A 00                CALL    LOC
8031    C3 0F 80                JP      MAIN1
8034
8034    ;A/D CONVERT
8034    ;
8034    C5          ADCONV:     PUSH    BC
8035    D5                      PUSH    DE
8036                            IN      A, (PORTC1)
8038    DB 22                   AND     0F0H
803A    B1                      OR      C
803B    EE 04                   XOR     04H
803D    D3 22                   OUT     (PORTC1), A
803F    EE 04                   XOR     04H
8041    D3 22                   OUT     (PORTC1), A
8043    0E 09                   LD      C, 9
8045    11 06 07                LD      DE, 0706H
8048
8048    7A          ADC1:       LD      A,D
8049    D3 23                   OUT     (CWR1), A
804B    7B                      LD      A, E
804C    D3 23                   OUT     (CWR1), A
804E    DB 20                   IN      A, (PORTA1)
8050    17                      RLA
8051    78                      LD      A, B
8052    17                      RLA
8053    47                      LD      B, A
8054    0D                      DEC     C
8055    C2 48 80                JP      NZ, ADC1
8058    3E 05                   LD      A, 5
805A    D3 23                   OUT     (CWR1), A
805C    78                      LD      A, B
805D    D1                      POP     DE
805E    C1                      POP     BC
805F    C9                      RET
8060
```

```
8060    ; HL = E*A
8060    ;
8060    16 00       MPY:    LD      D, 0
8062    62                  LD      H, D
8063    6A                  LD      L, D
8064    A7          MPY1:   AND     A
8065    C8                  RET     Z
8066    1F                  RRA
8067    D2 6B 80            JP      NC, MPY2
806A    19                  ADD     HL, DE
806B    EB          MPY2:   EX      DE, HL
806C    29                  ADD     HL, HL
806D    EB                  EX      DE, HL
806E    C3 64 80            JP      MPY1
8071
8071    ;HEX(HL)→BCD(D, C, B)
8071    ;
8071    F5          BCD:    PUSH    AF
8072    E5                  PUSH    HL
8073    1E 10               LD      E, 16
8075    16 00               LD      D, 0
8077    42                  LD      B, D
8078    4A                  LD      C, D
8079
8079    29          BCD1:   ADD     HL, HL
807A    79                  LD      A, C
807B    8F                  ADC     A, A
807C    27                  DAA
807D    4F                  LD      C, A
807E    78                  LD      A, B
807F    8F                  ADC     A, A
8080    27                  DAA
8081    47                  LD      B, A
8082    7A                  LD      A, D
8083    8F                  ADC     A, A
8084    27                  DAA
```

```
8085    57              LD      D, A
8086    1D              DEC     E
8087    C2 79 80        JP      NZ, BCD1
808A    E1              POP     HL
808B    F1              POP     AF
808C    C9              RET
808D
808D    ;IPUT
808D    ;
808D    C5      SWIN:   PUSH    BC
808E    01 00 04        LD      BC, 0400H
8091    DB 20           IN      A (PORTA1)
8093    17              RLA
8094    17              RLA
8095    17              RLA
8096    17              RLA
8097    17      SWIN1:  RLA
8098    D2 A1 80        JP      NC, SWIN2
809B    0C              INC     C
809C    10 F9           DJNZ    SWIN1
809E    0E FF           LD      C, 0FFH
80A0    37              SCF
80A1    3F      SWIN2:  CCF
80A2    79              LD      A, C
80A3    C1              POP     BC
80A4    C9              RET
80A5
80A5    21 00 50        DELAY:  LD      HL, 5000H
80A8    2B      DELAY1: DEC     HL
80A9    7C              LD      A, H
80AA    B5              OR      L
80AB    C2 A8 80        JP      NZ, DELAY1
80AE    C9              RET
80AF
0057            DHEXA4  EQU     57H
005A            LOC     EQU     5AH
```

```
0069    CLS     EQU     69H
80AF
0020    PORTA1  EQU     20H
0021    PORTB1  EQU     21H
0022    PORTC1  EQU     22H
0023    CWR1    EQU     23H
0027    CWR2    EQU     27H
80AF
80AF            END
```

Chịu trách nhiệm xuất bản

Giám đốc : NGUYỄN ĐÌNH THIÊM

Tổng biên tập : NGUYỄN BÁ NGỌC

Biên tập, sửa bài : KIỀU XUÂN THỰC

NGUYỄN THANH HÀ

Trình bày bìa : BÀNH HOÀNG ANH

GIÁO TRÌNH KỸ THUẬT VI XỬ LÝ
In 500 cuốn, In tại Công ty in Ka Long
Số xuất bản: 699/XB - QLXB.
In xong và nộp lưu chiểu quý III năm 2004