LỜI NÓI ĐẦU

Cùng với các mô đun của ngành Điện tử công nghiệp, mô đun Vi điều khiển là mô đun kỹ thuật chuyên ngành quan trọng của ngành điện tử, hiện nay mô đun được ứng dụng rộng rãi trong ngành kỹ thuật và các lĩnh vực điều khiển khác.

Mô đun được ứng dụng cho tất cả sinh viên ngành Sửa chữa thiết bị tự động hóa của trường ta. Bởi vậy để tạo điều kiện cho việc học tập và nghiên cứu mô đun của học viên được thuận lợi trong quá trình học tập. Bộ môn Điện tử thuộc Khoa Điện – Điện tử - Ti học Trường cao đẳng nghề số 20/QK3 tổ chức biên soạn tài liệu: " ***Vi điều khiển*** " làm bài giảng lưu hành hội bộ. Trong quá trình biên soạn chắc chắn sẽ không tránh khỏi những thiếu sót, bởi vậy tôi mong nhận được sự thông cảm và góp ý chân thành của các bạn đồng nghiệp để cho giáo trình ngày càng hoàn thiện hơn.

Xin chân thành cảm ơn!

MỤC LỤC

BÀI 1: SƠ LƯỢC VỀ LỊCH SỬ VÀ HƯỚNG PHÁT TRIỂN CỦA VI ĐIỀU KHIỂN

1. Lịch sử phát triển

Bộ Vi xử lý có khả năng vượt bậc so với các hệ thống khác về khả năng tính toán, xử lý, và thay đổi chương trình linh hoạt theo mục đích người dùng, đặc biệt hiệu quả đối với các bài toán và hệ thống lớn.Tuy nhiên đối với các ứng dụng nhỏ, tầm tính toán không đòi hỏi khả năng tính toán lớn thì việc ứng dụng vi xử lý cần cân nhắc. Bởi vì hệ thống dù lớn hay nhỏ, nếu dùng vi xử lý thì cũng đòi hỏi các khối mạch điện giao tiếp phức tạp như nhau. Các khối này bao gồm bộ nhớ để chứa dữ liệu và chương trình thực hiện, các mạch điện giao tiếp ngoại vi để xuất nhập và điều khiển trở lại, các khối này cùng liên kết với vi xử lý thì mới thực hiện được công việc. Để kết nối các khối này đòi hỏi người thiết kế phải hiểu biết tinh tường về các thành phần vi xử lý, bộ nhớ, các thiết bị ngoại vi. Hệ thống được tạo ra khá phức tạp, chiếm nhiều không gian, mạch in phức tạp và vấn đề chính là trình độ người thiết kế. Kết quả là giá thành sản phẩm cuối cùng rất cao, không phù hợp để áp dụng cho các hệ thống nhỏ.

Vì một số nhược điểm trên nên các nhà chế tạo tích hợp một ít bộ nhớ và một số mạch giao tiếp ngoại vi cùng với vi xử lý vào một IC duy nhất được gọi là Microcontroller-Vi điều khiển. Vi điều khiển có khả năng tương tự như khả năng của vi xử lý, nhưng cấu trúc phần cứng dành cho người dùng đơn giản hơn nhiều. Vi điều khiển ra đời mang lại sự tiện lợi đối với người dùng, họ không cần nắm vững một khối lượng kiến thức quá lớn như người dùng vi xử lý, kết cấu mạch điện dành cho người dùng cũng trở nên đơn giản hơn nhiều và có khả năng giao tiếp trực tiếp với các thiết bị bên ngoài. Vi điều khiển tuy được xây dựng với phần cứng dành cho người sử dụng đơn giản hơn, nhưng thay vào lợi điểm này là khả năng xử lý bị giới hạn (tốc độ xử lý chậm hơn và khả năng tính toán ít hơn, dung lượng chương trình bị giới hạn). Thay vào đó, Vi điều khiển có giá thành rẻ hơn nhiều so với vi xử lý, việc sử dụng đơn giản, do đó nó được ứng dụng rộng rãi vào nhiều ứng dụng có chức năng đơn giản, không đòi hỏi tính toán phức tạp.

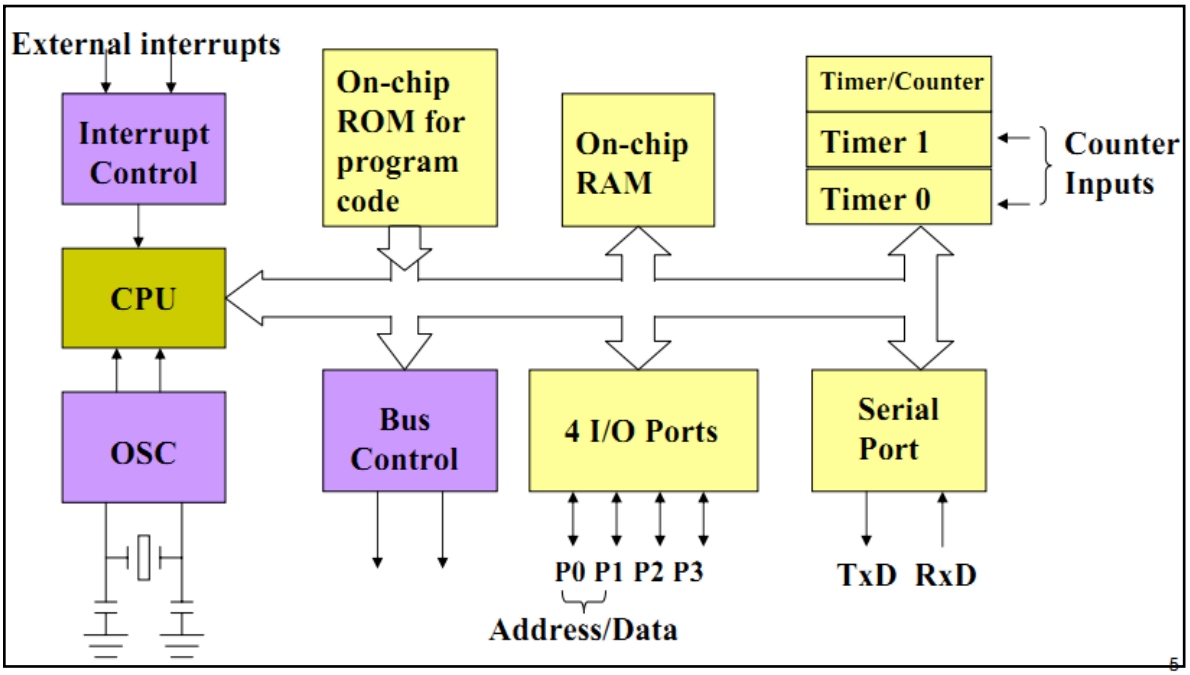
Vi điều khiển được ứng dụng trong các dây chuyền tự động loại nhỏ, các robot có chức năng đơn giản, trong máy giặt, ôtô v.v...

Năm 1976 Intel giới thiệu bộ vi điều khiển (microcontroller) 8748, một chip tương tự như các bộ vi xử lý và là chip đầu tiên trong họ MCS-48. Độ phức tạp, kích thước và khả năng của Vi điều khiển tăng thêm một bậc quan trọng vào năm 1980 khi intel tung ra chip 8051, bộ Vi điều khiển đầu tiên của họ MCS-51

và là chuẩn công nghệ cho nhiều họ Vi điều khiển được sản xuất sau này. Sau đó rất nhiều họ Vi điều khiển của nhiều nhà chế tạo khác nhau lần lượt được đưa ra thị trường với tính năng được cải tiến ngày càng mạnh.

2. Vi điều khiển

2.1. Nguyên lý cấu tạo



Điểm cần lưu ý về vi điều khiển là sơ đồ khối cấu tạo của nó. Cấu tạo một họ microcontroller chủ yếu dựa trên một kiểu tiêu chuẩn bao gồm các tính năng quan trọng nhất, nhiều chủng loại phù hợp với các lĩnh vực ứng dụng đặc biệt khác nhau, có thể kết hợp thêm thiết bị ngoại vi để tăng khả năng hoặc giảm nhỏ kích thước đến mức tối thiểu trong các ứng dụng chuyên biệt như: Kết nối bus, kết nối video hoặc điều khiển trực tiếp các cơ cấu hiển thị LCD...Với kiểu tiêu chuẩn cũng đủ dùng cho hầu hết các ứng dụng

2.2. Các kiểu cấu trúc bộ nhớ

Các bộ vi xử lý và vi điều khiển hiện nay chủ yếu được chế tạo theo 2 kiểu cấu trúc khác nhau: Cấu trúc Von Neumann và Harvard.

2.2.1. Cấu trúc Von Neumann

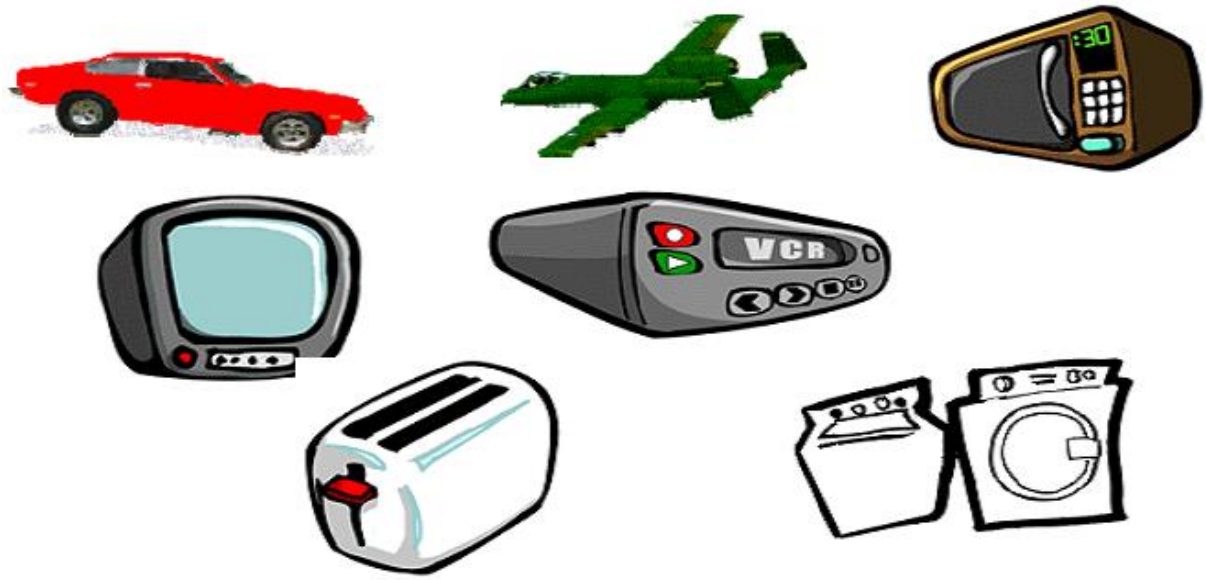
Trong cấu trúc Von Neumann chỉ có một vùng địa chỉ tuyến tính bao gồm tất cả dữ liệu và lệnh điều khiển, độ lớn của vùng địa chỉ phụ thuộc vào chiều dài của bộ đếm chương trình, nếu không trang bị thêm linh kiện phụ thì việc định địa chỉ bộ nhớ chương trình và bộ nhớ dữ liệu không độc lập với nhau. Trong cấu trúc này chỉ tồn tại một bus dữ liệu và một bus địa chỉ để đọc-ghi dữ liệu và đọc lệnh điều khiển chương trình và không có khả năng thực hiện song song (truy xuất đồng thời bộ nhớ dữ liệu và bộ nhớ chương trình)

2.2.2. Cấu trúc Harvard

Gồm hai vùng địa chỉ riêng biệt cho bộ nhớ dữ liệu và bộ nhớ chương trình nên có thể truy xuất song song dữ liệu và lệnh điều khiển, cấu trúc này đặc biệt thích hợp với các vi điều khiển 16 và 32 bít vì làm tăng tốc độ làm việc. Nếu chỉ có một hệ thống bus như thường thấy ở vi điều khiển 8 bít thì việc truy xuất bộ nhớ dữ liệu hoặc bộ nhớ chương trình sẽ được thực hiện thông qua các tín hiệu điều khiển, nếu không có yêu cầu ghi vào bộ nhớ chương trình thì cấu trúc này còn cho phép tăng tính an toàn của chương trình

3. Lĩnh vực và ứng dụng

- Hầu hết các thiết bị điện tử hiện nay đều sử dụng các chip vi điều khiển
- Ứng dụng trong nhiều lĩnh vực khác nhau:
  + Điện thoại di động
  + Các thiết bị đo lường điện tử
  + Tivi, CD players, radio,…
  + Bàn phím PC
  + Các hệ thống bảo vệ, báo cháy….
  + Các máy in



4. Hướng phát triển

Yêu cầu đặt ra cho vi điều khiển hiện nay là tăng lĩnh vực ứng dụng với tốc độ xử lý ngày càng nhanh và kích thước nhỏ gọn, công suất tiêu thụ thấp. Vấn đề đặt ra là liệu với vi điều khiển 8 bít có còn phù hợp hay không ? hoặc trong tương lai phải thay bằng các vi điều khiển 16/32 bít.

Khác với vi xử lý việc phát triển luôn kèm theo việc nâng cao khả năng tính toán bằng cách mở rộng hệ thống bus. Đối với vi điều khiển không nhất thiết phải như thế, một vi điều khiển 8 bít cũng đủ cho rất nhiều ứng dụng và vi

điều khiển 16 bít là hoàn toàn quá dư thừa, trong trường hợp cần giảm giá thành, kích thước và công suất tiêu thụ thì vi điều khiển 4 bít là giải pháp tối ưu. Một vài ứng dụng cần vi điều khiển có nhiều khối ngoại vi, có ứng dụng lại cần ngoại vi tốc độ cao, hướng phát triển tương lai là tăng khả năng của CPU và khối ngoại vi

Một hướng đơn giản là tăng tần số xung đồng hồ để rút ngắn thời gian thực hiện chương trình, giảm thời gian biến đổi A/D và tăng tần số giới hạn của timer. Tuy nhiên các linh kiện bên ngoài cũng phải có khả năng làm việc ở tần số cao, khi tăng tần số đồng cũng làm tăng công suất tiêu thụ của vi điều khiển

Việc tối ưu hóa cấu trúc chương trình và bộ nhớ cũng góp phần nâng cao khả năng hệ thống. Trong các ứng dụng đa nhiệm, phương pháp phân đoạn và phân dãy hóa có một ý nghĩa rất lớn

BÀI 2: CẤU TRÚC HỌ VI ĐIỀU KHIỂN 8051

1. Tổng quan

Các thành viên của họ MCS-51 (Atmel) có các đặc điểm chung như sau:

+ Có 4/8/12/20 Kbyte bộ nhớ FLASH ROM bên trong để lưu chương trình. Nhờ vậy Vi điều khiển có khả năng nạp xoá chương trình bằng điện đến 10000 lần.

+ 128 Byte RAM nội

+ 4 Port xuất/nhập 8 bit

+ Từ 2 đến 3 bộ định thời 16-bit

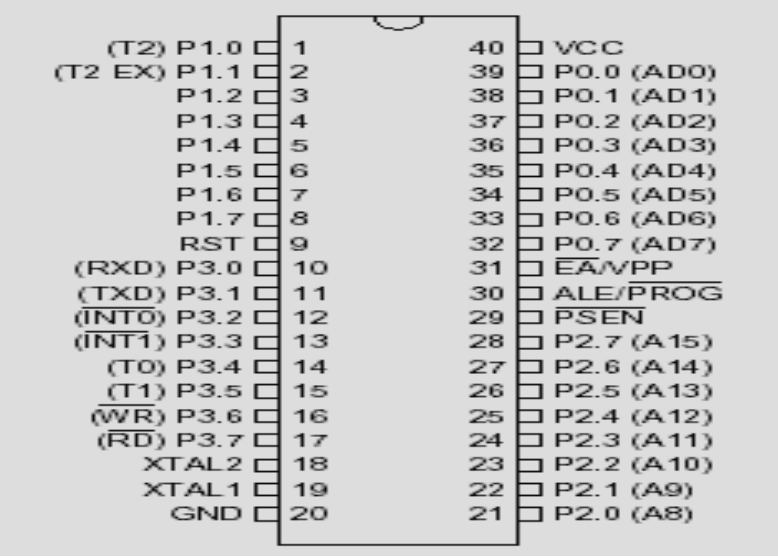
+ Có khả năng giao tiếp truyền dữ liệu nối tiếp

+ Có thể mở rộng không gian nhớ chương trình ngoài 64KByte (bộ nhớ ROM ngoại): *khi chương trình do người lập trình viết ra có dung lượng lớn hơn dung lượng bộ nhớ ROM nội, để lưu được chương trình này cần bộ nhớ ROM lớn hơn, cách giải quyết là kết nối Vi điều khiển với bộ nhớ ROM từ bên ngoài (hay còn gọi là ROM ngoại). Dung lượng bộ nhớ ROM ngoại lớn nhất mà Vi điều khiển có thể kết nối là 64Kbyte*

+ Có thể mở rộng không gian nhớ dữ liệu ngoài 64KByte (bộ nhớ RAM ngoại)

+ Bộ xử lí bit (thao tác trên các bit riêng rẽ) 210 bit có thể truy xuất đến từng bit

2. Sơ đồ chân



Hình 2.1. Sơ đồ chân vi điều khiển 8051

* Port 0 (P0)

Port 0 gồm 8 chân (từ chân 32 đến 39) có hai chức năng:

- Chức năng xuất/nhập :*các chân này được dùng để nhận tín hiệu từ bên ngoài vào để xử lí, hoặc dùng để xuất tín hiệu ra bên ngoài, chẳng hạn xuất tín hiệu để điều khiển led đơn sáng tắt.*

- Chức năng là bus dữ liệu và bus địa chỉ (AD7-AD0) : *8 chân này (hoặc Port 0) còn làm nhiệm vụ lấy dữ liệu từ ROM hoặc RAM ngoại (nếu có kết nối với bộ nhớ ngoài), đồng thời Port 0 còn được dùng để định địa chỉ của bộ nhớ ngoài.*

* Port 1 (P1)

- Port P1 gồm 8 chân (từ chân 1 đến chân 8), chỉ có chức năng làm các đường xuất/nhập, không có chức năng khác.

* Port 2 (P2)

Port 2 gồm 8 chân (từ chân 21 đến chân 28) có hai chức năng:

- Chức năng xuất/nhập

- Chức năng là bus địa chỉ cao (A8-A15): khi kết nối với bộ nhớ ngoài có dung lượng lớn,cần 2 byte để định địa chỉ của bộ nhớ, byte thấp do P0 đảm nhận, byte cao do P2 này đảm nhận.

* Port 3 (P3)

Port 3 gồm 8 chân (từ chân 10 đến 17):

- Chức năng xuất/nhập

- Với mỗi chân có một chức năng riêng thứ hai như trong bảng sau

Bit	Tên	Chức năng
P3.0	RxD	Ngõ vào nhận dữ liệu nối tiếp
P3.1	TxD	Ngõ xuất dữ liệu nối tiếp
P3.2	INT0	Ngõ vào ngắt cứng thứ 0
P3.3	INT1	Ngõ vào ngắt cứng thứ 1
P3.4	T0	Ngõ vào của Timer/Counter thứ 0
P3.5	T1	Ngõ vào của Timer/Counter thứ 1
P3.6	WR	Ngõ điều khiển ghi dữ liệu lên bộ nhớ ngoài
P3.7	RD	Ngõ điều khiển đọc dữ liệu từ bộ nhớ bên ngoài
P1.0	T2	Ngõ vào của Timer/Counter thứ 2
P1.1	T2X	Ngõ Nạp lại/thu nhận của Timer/Counter thứ 2

* Chân cho phép bộ nhớ chương trình

- PSEN (program store enable) tín hiệu được xuất ra ở chân 29 dùng để truy xuất bộ nhớ chương trình ngoài. Chân này thường được nối với chân OE (output enable) của ROM ngoài.

- Khi vi điều khiển làm việc với bộ nhớ chương trình ngoài, chân này phát ra tín hiệu kích hoạt ở mức thấp và được kích hoạt 2 lần trong một chu kì máy

- Khi thực thi một chương trình ở ROM nội, chân này được duy trì ở mức logic không tích cực (logic 1)

(Không cần kết nối chân này khi không sử dụng đến)

* Chân cho phép chốt địa chỉ ALE

- Khi Vi điều khiển truy xuất bộ nhớ từ bên ngoài, port 0 vừa có chức năng là bus địa chỉ, vừa có chức năng là bus dữ liệu do đó phải tách các đường dữ liệu và địa chỉ. Tín hiệu ở chân ALE dùng làm tín hiệu điều khiển để giải đa hợp các đường địa chỉ và các đường dữ liệu khi kết nối chúng với IC chốt.

- Các xung tín hiệu ALE có tốc độ bằng 1/6 lần tần số dao động đưa vào Vi điều khiển, như vậy có thể dùng tín hiệu ở ngõ ra ALE làm xung clock cung cấp cho các phần khác của hệ thống.

- Ghi chú: khi không sử dụng có thể bỏ trống chân này

* Chân truy suất ROM ngoài (EA)

- Chân EA dùng để xác định chương trình thực hiện được lấy từ ROM nội hay ROM ngoại.

- Khi EA nối với logic 1(+5V) thì Vi điều khiển thực hiện chương trình lấy từ bộ nhớ nội

- Khi EA nối với logic 0(0V) thì Vi điều khiển thực hiện chương trình lấy từ bộ nhớ ngoại

* Chân Reset

- Ngõ vào RST ở chân 9 là ngõ vào Reset dùng để thiết lập trạng thái ban đầu cho vi điều khiển. Hệ thống sẽ được thiết lập lại các giá trị ban đầu nếu ngõ này ở mức 1 tối thiểu 2 chu kì máy.

* Các chân Xtal 1 và Xtal 2

- Hai chân này có vị trí chân là 18 và 19 được sử dụng để nhận nguồn xung clock từ bên ngoài để hoạt động, thường được ghép nối với thạch anh và các tụ để tạo nguồn xung clock ổn định.

* Chân cấp nguồn

- Chân số 40 là VCC cấp điện áp nguồn cho Vi điều khiển

- Nguồn điện cấp là +5V±0.5.

- Chân số 20 nối GND(hay nối Mass)

3. Cấu trúc Port I/O

- Port 0 (Chân 32 đến 39): P0 (P0.0 đến P0.7)
- Port 1 (Chân 1 đến 8): P1 (P1.0 đến P1.7)
- Port 2 (Chân 21 đến 28): P2 (P2.0 đến P2.7)
- Port 3 (Chân 10 đến 17): P3 (P3.0 đến P3.7)

Mỗi cổng có 8 chân

- Đánh tên từ P0.X (X = 0, 1...7), P1.X, P2.X, P3.X
- Với P0.0 là bít 0 (LSB) của P0
  P0.7 là bít 7 (MSB) của P0
- 8 bít này cấu thành một byte
- Mỗi cổng có thể được dùng như input hay output

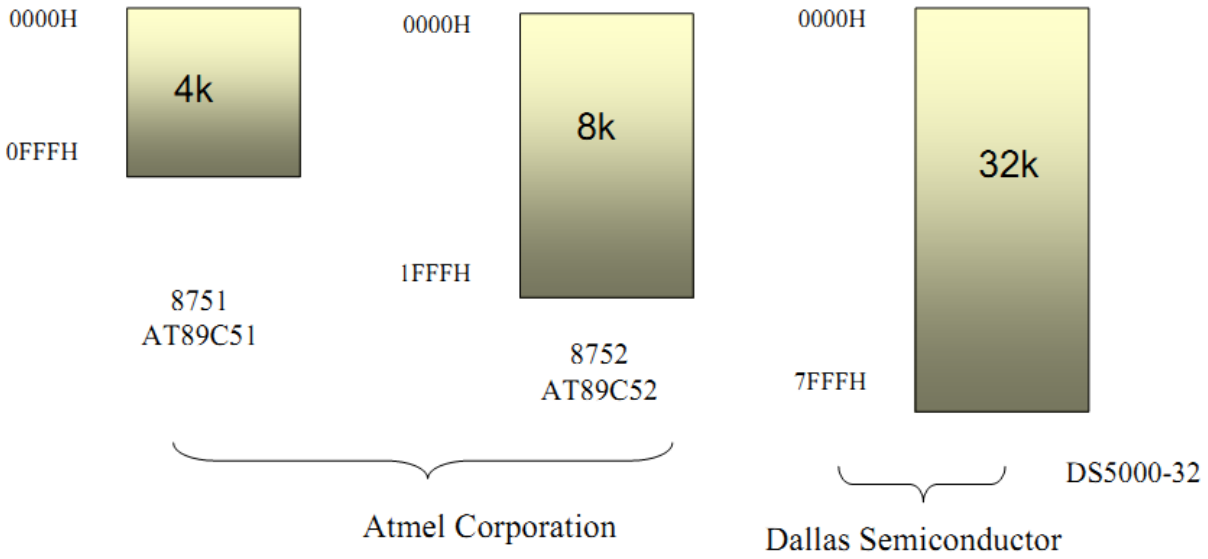
4. Tổ chức bộ nhớ

4.1. Bộ nhớ chương trình (ROM)

Bộ nhớ ROM dùng để lưu chương trình do người viết chương trình viết ra. Chương trình là tập hợp các câu lệnh thể hiện các thuật toán để giải quyết các công việc cụ thể, chương trình do người thiết kế viết trên máy vi tính, sau đó được đưa vào lưu trong ROM của vi điều khiển, khi hoạt động, vi điều khiển truy xuất từng câu lệnh trong ROM để thực hiện chương trình. ROM còn dùng để chứa số liệu các bảng, các tham số hệ thống, các số liệu cố định của hệ thống. Trong quá trình hoạt động nội dung ROM là cố định, không thể thay đổi, nội dung ROM chỉ thay đổi khi ROM ở chế độ xóa hoặc nạp chương trình (do các mạch điện riêng biệt thực hiện).

- Bộ nhớ ROM được tích hợp trong chip Vi điều khiển với dung lượng tùy vào chủng loại cần dùng, chẳng hạn đối với 89S52 là 8KByte, với 89S53 là 12KByte.

- Bộ nhớ bên trong Vi điều khiển 89Sxx là bộ nhớ Flash ROM cho phép xóa bộ nhớ ROM bằng điện và nạp vào chương trình mới cũng bằng điện và có thể nạp xóa nhiều lần



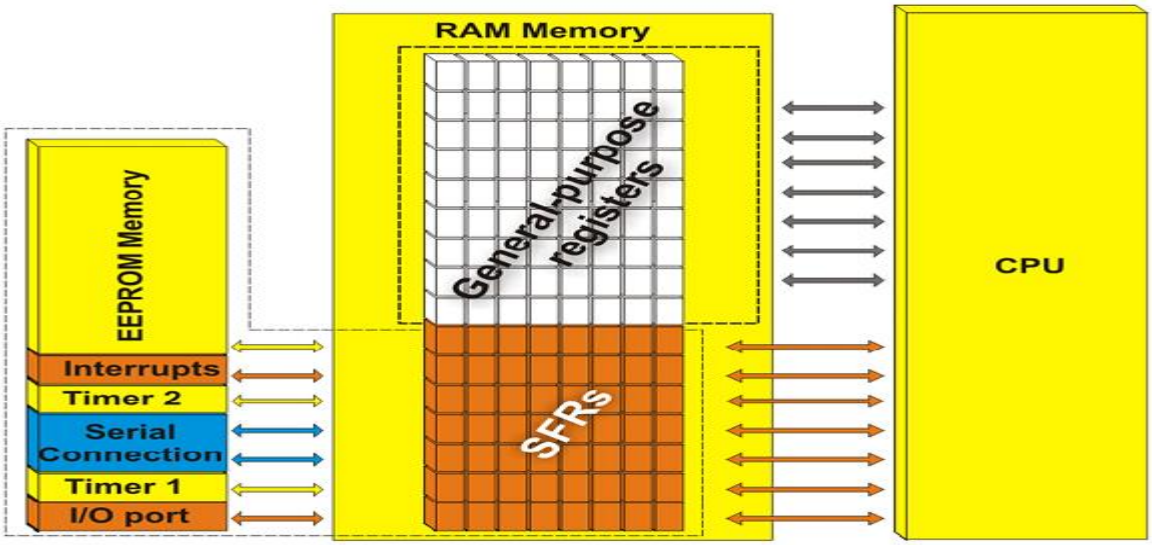
Hình 2.2. Tổ chức bộ nhớ ROM của 8051

Bộ nhớ ROM được định địa chỉ theo từng Byte, các byte được đánh địa chỉ theo số hex-số thập lục phân, bắt đầu từ địa chỉ 0000H, khi viết chương trình cần chú ý đến địa chỉ lớn nhất trên ROM, chương trình được lưu sẽ bị mất khi địa chỉ lưu vượt qua vùng này. Ví dụ: AT89S52 có 8KByte bộ nhớ ROM nội, địa chỉ lớn nhất là 1FFFH, nếu chương trình viết ra có dung lượng lớn hơn 8KByte các byte trong các địa chỉ lớn hơn 1FFFH sẽ bị mất.

- Ngoài ra Vi điều khiển còn có khả năng mở rộng bộ nhớ ROM với việc giao tiếp với bộ nhớ ROM bên ngoài lên đến 64KByte(địa chỉ từ 0000H đến FFFFH)

4.2. Bộ nhớ dữ liệu

Bộ nhớ RAM dùng làm môi trường xử lý thông tin, lưu trữ các kết quả trung gian và kết quả cuối cùng của các phép toán, xử lí thông tin. Nó cũng dùng để tổ chức các vùng đệm dữ liệu, trong các thao tác thu phát, chuyển đổi dữ liệu.



Hình 2.3. Giao tiếp bộ nhớ

* Vùng RAM đa năng

Mặc dù hình 2.4 trình bày vùng RAM đa năng có 80 byte đặt ở địa chỉ từ 30H đến 7FH, bên dưới vùng này từ địa chỉ 00H đến 2FH là vùng nhớ có thể được sử dụng tương tự (mặc dù vùng này có công dụng khác). Bất kỳ vị trí nhớ nào trong vùng RAM đa năng đều có thể được truy xuất tự do bằng cách sử dụng cách định địa chỉ trực tiếp hoặc gián tiếp. Ví dụ để đọc nội dung tại 5FH của RAM nội vào thanh ghi A có thể dùng lệnh sau

MOV A, 5FH

Lệnh trên di chuyển một byte dữ liệu bằng cách dùng kiểu định địa chỉ trực tiếp để xác định vị trí nguồn (nghĩa là địa chỉ 5FH). Đích của dữ liệu được xác định rõ ràng trong mã lệnh là thanh ghi A

Vùng RAM đa năng còn có thể được truy xuất bằng cách dùng kiểu định địa chỉ gián tiếp qua các thanh ghi R0, R1. Ví dụ hai lệnh sau thực hiện cùng công việc như ở ví dụ trên.

MOV R0,#5FH

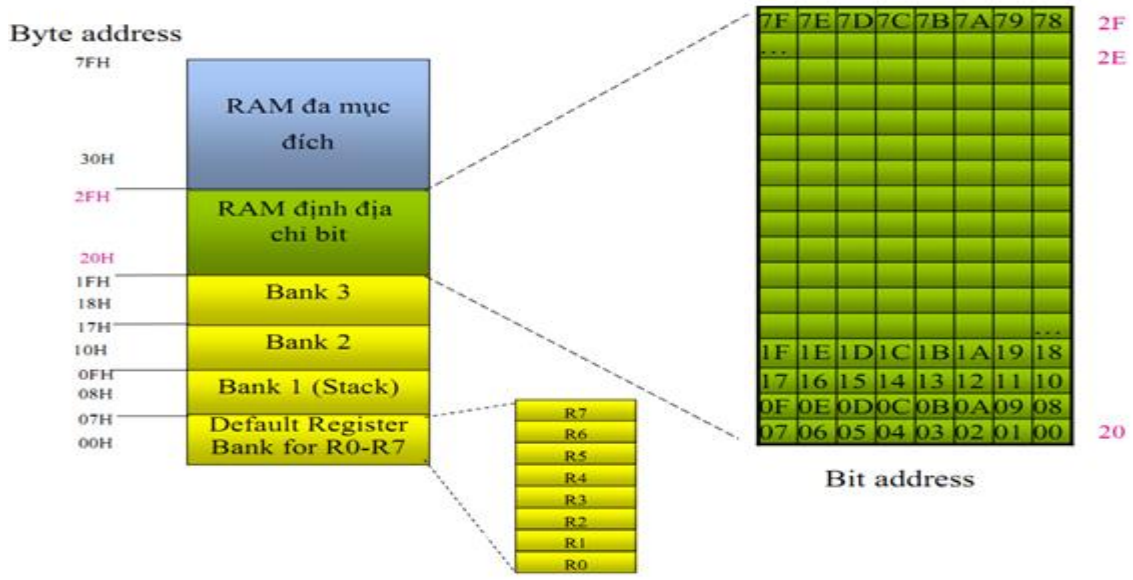
MOV A,@R0

Lệnh đầu tiên sử dụng kiểu định địa chỉ tức thời di chuyển giá trị 5FH vào thanh ghi R0, lệnh tiếp theo sử dụng cách định địa chỉ gián tiếp di chuyển dữ liệu trỏ bởi R0 vào thanh ghi A

7F	Vùng RAM đa năng								FF								
									F0	F7	F6	F5	F4	F3	F2	F1	F0	B
									E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
									D0	D7	D6	D5	D4	D3	D2	-	D0	PSW
30									B8	-	-	-	BC	BB	BA	B9	B8	IP
2F	7F	7E	7D	7C	7B	7A	79	78										
2E	77	76	75	74	73	72	71	70	B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
2D	6F	6E	6D	6C	6B	6A	69	68										
2C	67	66	65	64	63	62	61	60	A8	AF	-	-	AC	AB	AA	A9	A8	IE
2B	5F	5E	5D	5C	5B	5A	59	58										
2A	57	56	55	54	53	52	51	50	A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
29	4F	4E	4D	4C	4B	4A	49	48										
28	47	46	45	44	43	42	41	40	99	Địa chỉ byte								SBUF
27	3F	3E	3D	3C	3B	3A	39	38	98	9F	9E	9D	9C	9B	9A	99	98	SCON
26	37	36	35	34	33	32	31	30										
25	2F	2E	2D	2C	2B	2A	29	28	90	97	96	95	94	93	92	91	90	P1
24	27	26	25	24	23	22	21	20										
23	1F	1E	1D	1C	1B	1A	19	18	8D	Địa chỉ byte								TH1
22	17	16	15	14	13	12	11	10	8C	Địa chỉ byte								TH0
21	0F	0E	0D	0C	0B	0A	09	08	8B	Địa chỉ byte								TL1
20	07	06	05	04	03	02	01	00	8A	Địa chỉ byte								TL0
1F	Bank 3								89	Địa chỉ byte								TMOD
18									88	8F	8E	8D	8C	8B	8A	89	88	TCON
17	Bank 2								87	Địa chỉ byte								PCON
10																		
0F	Bank 1								83	Địa chỉ byte								DPH
08									82	Địa chỉ byte								DPL
07	Bank 0 (mặc định) R0-R7								81	Địa chỉ byte								SP
00									80	87	86	85	84	83	82	81	80	P0

Hình 2.4. Cấu trúc RAM nội của 8051

* Vùng RAM địa chỉ bit



Hình 2.5. Vùng RAM định địa chỉ bit

Trên RAM nội có 210 ô nhớ bit được định địa chỉ và có thể truy xuất đến từng bit, các bit nhớ này cũng được định địa chỉ bằng các số thập lục phân- số Hex. Trong đó có 128 bit nằm trong các ô nhớ có địa chỉ byte từ 20H đến 2FH, các bit nhớ còn lại chứa trong nhóm thanh ghi có chức năng đặc biệt.

Mặc dù các bit nhớ và ô nhớ (byte) cùng được định bằng số Hex, tuy nhiên chúng sẽ được nhận dạng là địa chỉ bit hay địa chỉ byte thông qua các câu lệnh tương ứng dành cho các bit nhớ hoặc các ô nhớ này.

Ví dụ:

Mov 05H,#10111111B; >>> lệnh này thiết lập giá trị cho ***ô nhớ*** có địa chỉ là 05H

JB 05H,nhan01; >>> lệnh này liên quan đến trạng thái của ***bit nhớ*** có địa chỉ 05H

* Các dãy thanh ghi

Các thanh ghi này được định địa chỉ byte, một số được định thêm địa chỉ bit, có địa chỉ của các thanh ghi này nằm trong khoảng 80H đến FFH. Các thanh ghi đặc biệt này này được dùng để xác lập trạng thái hoạt động cần thiết cho Vi điều khiển.

5. Các thanh ghi chức năng đặc biệt

* Từ trạng thái chương trình PSW

BIT	ĐỊA CHỈ BIT	KÍ HIỆU	CHỨC NĂNG
PSW.7	D7H	C hoặc Cy	Cờ nhớ

PSW.6	D6H	AC	Cờ nhớ phụ
PSW.5	D5H	F0	Cờ 0 hay cờ Zero
PSW.4	D4H	RS1	Bit lựa chọn dãy thanh ghi
PSW.3	D3H	RS0	Bit lựa chọn dãy thanh ghi
PSW.2	D2H	0V	Cờ tràn với phép tính liên quan đến số nhị phân có dấu
PSW.1	D1H	-	Chưa được thiết kế để sử dụng
PSW.0	D0H	P	Cờ chẵn lẻ

Chức năng từng bit trong thanh trạng thái PSW

- *Cờ nhớ C*: Cờ được sử dụng trong các lệnh toán học:

+ C=1 nếu phép toán cộng xảy ra tràn hoặc phép trừ có mượn

+ C=0 nếu phép toán cộng không tràn hoặc phép trừ không có mượn.

- *Cờ nhớ phụ AC:*

+ Cờ AC được dùng trong các phép toán cộng hai số BCD.

Khi cộng số BCD:

+ Nếu kết quả 4 bit lớn hơn 09H thì AC=1

+ Nếu kết quả 4 bit dưới 09H thì AC=0.

- *Cờ 0 hay cờ nhớ Z:*

+ Cờ Z = 0 khi thanh ghi A có giá trị khác 0

+ Cờ Z =1 khi A thanh ghi A có giá trị là 0

- *Các bit chọn bank thanh ghi:*

Hai bit RS1 và RS2 dùng để xác lập bank thanh ghi được sử dụng, mặc định RS1=0 và RS2=0

RS1	RS2	Bank thank ghi được sử dụng
0	0	Bank 0
0	1	Bank 1
1	0	Bank 2
1	1	Bank 3

- *Cờ tràn OV*

+ Được sử dụng trong các phép toán cộng có dấu, với các phép toán cộng không dấu cờ tràn OV được bỏ qua, không cần quan tâm đến OV. Nếu:

Phép cộng hai số có dấu lớn hơn +127 thì OV=1

Hoặc phép trừ hai số có dấu nhỏ hơn -127 thì OV=1

Các trường hợp còn lại OV=0

- *Cờ chẵn lẻ*

+ Cờ chẵn lẻ P tự động được đặt bằng 1 hoặc 0 sao cho tổng số bit mang giá trị 1 trên thanh ghi A với cờ P luôn là một số chẵn. Cờ chẵn lẻ được dùng để xử lí dữ liệu trước khi truyền đi theo kiểu nối tiếp hoặc xử lí dữ liệu trước khi nhận vào theo kiểu nối tiếp (hạn chế lỗi phát sinh trong quá trình truyền).

* Thanh ghi B

Thanh ghi B ở địa chỉ F0H, được dùng với thanh ghi A để thực hiện các phép toán số học. Khi thực hiện lệnh chia với thanh ghi A, số dư được lưu trữ ở thanh ghi B. Ngoài ra thanh ghi B còn được dùng như một thanh ghi đệm có nhiều chức năng

* Con trỏ Stack

Con trỏ stack SP nằm tại địa chỉ 81h và không cho phép định địa chỉ bit. SP dùng để chỉ đến đỉnh của stack. Stack là một dạng bộ nhớ lưu trữ dạng Lipo (Last in first out) thường dùng lưu trữ địa chỉ trả về khi gọi một chương trình con. Ngoài ra, stack còn dùng như bộ nhớ tạm để lưu lại và khôi phục các giá trị cần thiết.

Mặc định khi khởi động, giá trị của SP là 07h

* Con trỏ dữ liệu DPTR

Con trỏ dữ liệu DPTR là thanh ghi 16 bit duy nhất của Vi điều khiển được tạo thành từ hai thanh ghi DPL (byte thấp-địa chỉ byte 82H) và DPH (byte cao-địa chỉ byte 83H). Hai thanh ghi DPL và DPH có thể truy xuất độc lập bởi người sử dụng. Con trỏ dữ liệu DPTR thường được sử dụng khi truy xuất dữ liệu từ bộ nhớ ROM hoặc bộ nhớ từ bên ngoài.

* Các thanh ghi Port

Các thanh ghi P0 tại địa chỉ 80h, P1 tại địa chỉ 90h, P2 tại địa chỉ A0h, P3 tại địa chỉ B0h là các thanh ghi chốt cho 4 port xuất/nhập (Port 0, 1, 2, 3). Tất cả các thanh ghi này đều cho phép định địa chỉ bit trong đó địa chỉ bit của P0 từ 80h – 87h, P1 từ 90h – 97h, P2 từ A0h – A7h và P3 từ B0h – B7h. Các địa chỉ bit này có thể thay thế bằng toán tử địa chỉ.

* Các thanh ghi định thời

Các cặp thanh ghi (TH0, TL0), (TH1, TL1) và (TH2, TL2) là các thanh ghi cho các bộ định thời 0, 1 và 2 trong đó bộ định thời 2 chỉ có trong 8032/8052.

* Các thanh ghi của Port nối tiếp

Thanh ghi port nối tiếp tại địa chỉ 99h thực chất bao gồm 2 thanh ghi: thanh ghi nhận và thanh ghi truyền. Nếu dữ liệu đưa tới SBUF thì đó là thanh ghi

truyền, nếu dữ liệu được đọc từ SBUF thì đó là thanh ghi nhận. Các thanh ghi này không cho phép định địa chỉ bit.

* Các thanh ghi ngắt

- Thanh ghi IP tại địa chỉ B8h cho phép chọn mức ưu tiên ngắt khi có 2 ngắt xảy ra đồng thời. IP cho phép định địa chỉ bit từ B8h – BFh.

- Thanh ghi IE tại địa chỉ A8h cho phép hay cấm các ngắt. IE có địa chỉ bit từ A8h – Afh.

* Thanh ghi điều khiển nguồn

- Thanh ghi TMOD tại địa chỉ 89h dùng để chọn chế độ hoạt động cho các bộ định thời (0, 1) và không cho phép định địa chỉ bit.

- Thanh ghi TCON tại địa chỉ 88h điều khiển hoạt động của bộ định thời và ngắt. TCON có địa chỉ bit từ 88h – 8Fh.

- Thanh ghi T2CON tại địa chỉ C8h điều khiển hoạt động của bộ định thời 2. T2CON có địa chỉ bit từ C8h – CFh.

- Thanh ghi SCON tại địa chỉ 98h điều khiển hoạt động của port nối tiếp. SCON có địa chỉ bit từ 98h – 9Fh.

* Thanh ghi tích lũy (A hay Acc)

- Thanh ghi tích lũy là thanh ghi sử dụng nhiều nhất trong AT89C51, được ký hiệu trong câu lệnh là A. Ngoài ra, trong các lệnh xử lý bit, thanh ghi tích lũy được ký hiệu là Acc.

- Thanh ghi tích lũy có thể truy xuất trực tiếp thông qua địa chỉ E0h (byte) hay truy xuất từng bit thông qua địa chỉ bit từ E0h đến E7h.

Ví dụ

```
Mov A,#1
Mov 0E0h,#1
```

Hay

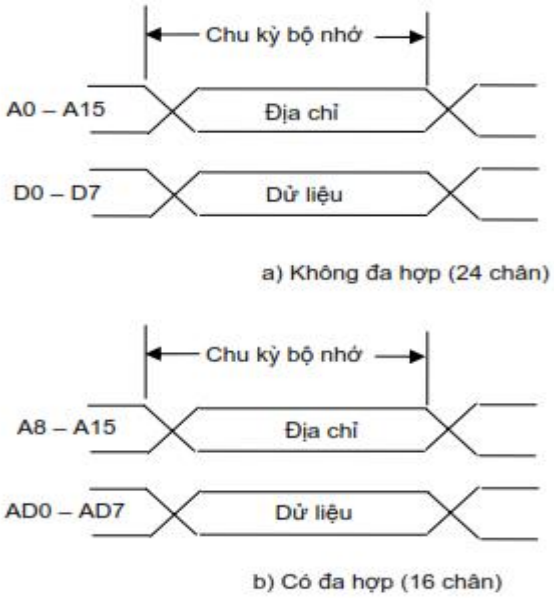
```
Setb Acc.4
Setb 0E4h
```

6. Bộ nhớ ngoài

Các bộ vi điều khiển cần có khả năng mở rộng các tài nguyên trên chip (bộ nhớ, I/O...) để tránh hiện tượng cổ chai trong thiết kế. Cấu trúc của MCS-51 cho phép khả năng mở rộng không gian bộ nhớ chương trình đến 64K và không gian bộ nhớ dữ liệu đến 64K ROM và RAM ngoài được thêm vào khi cần. Các IC giao tiếp ngoại vi cũng có thể được thêm vào để mở rộng khả năng xuất/nhập. Chúng trở thành một phần của không gian bộ nhớ dữ liệu ngoài bằng cách sử dụng cách định địa chỉ kiểu I/O ánh xạ bộ nhớ.

Khi bộ nhớ ngoài được sử dụng, port 0 không làm nhiệm vụ của port xuất/nhập, port này trở thành bus địa chỉ (A0..A7) và bus dữ liệu (D0..D7) đa

hợp. Ngõ ra ALE chốt một byte thấp của địa chỉ ở thời điểm bắt đầu một chu kỳ bộ nhớ ngoài. Port 2 thường (nhưng không phải luôn luôn) được dùng làm byte cao của bus địa chỉ

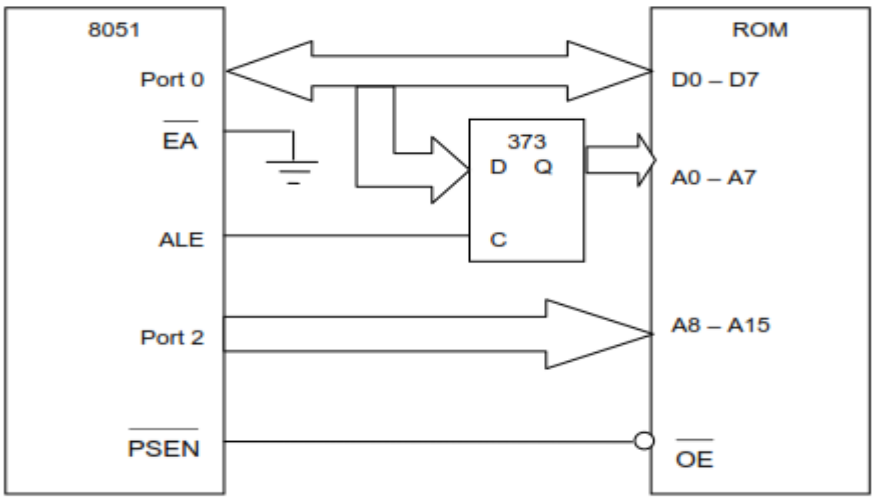


Hình 2.6. Bus đa hợp địa chỉ/ dữ liệu

Sắp xếp không đa hợp sử dụng 16 đường địa chỉ và 8 đường dữ liệu tổng cộng 24 đường. Sắp xếp đa hợp kết hợp 8 đường của bus dũ liệu và byte thấp của bus địa chỉ thì chỉ cần 16 đường. Việc tiết kiệm các chân cho phép đóng gói họ MCS-51 trong một vỏ 40 chân. Sắp xếp đa hợp hoạt động như sau: Trong ½ chu kỳ đầu của chu kỳ bộ nhớ, byte thấp của địa chỉ được cung cấp bởi port 0 và được chốt nhờ tín hiệu ALE. Mạch chốt 74373 giữ cho byte thấp của địa chỉ ổn định trong cả chu kỳ bộ nhớ. Trong ½ sau của chu kỳ bộ nhớ, port 0 được sử dụng làm bus dữ liệu và dữ liệu sẽ được đọc hay ghi.

6.1 Truy xuất bộ nhớ chương trình ngoài

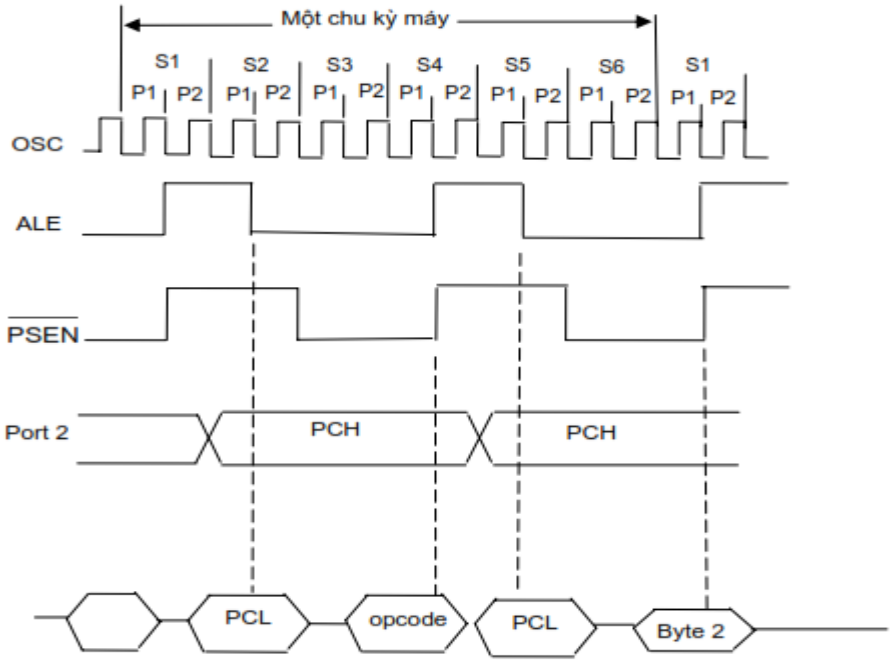
Bộ nhớ chương trình ngoài là bộ nhớ chỉ đọc, được cho phép bởi tín hiệu PSEN.



Hình 2.7. Truy xuất ROM ngoài

Khi có một ROM ngoài được sử dụng, cả hai port 0 và port 2 đều không còn là các port xuất/nhập. Kết nối phần cứng với bộ ngoài được trình bày ở hình 2.7

Một chu kỳ máy của 8051 có 12 chu kỳ dao động. Nếu bộ dao động trên chíp có tần số 12 MHz thì một chu kỳ máy dài 1 μs. Trong 1 chu kỳ máy điển hình, ALE có hai xung và 2 byte của lệnh được đọc từ bộ nhớ chương trình (nếu lệnh chỉ có 1 byte, byte thứ hai bị loại bỏ). Giản đồ thời gian của chu kỳ máy này được gọi là chu kỳ tìm - nạp lệnh được trình bày ở hình 2.8



Hình 2.8. Chu kỳ tìm nạp lệnh ROM ngoài

6.2 Truy xuất bộ nhớ dữ liệu ngoài

Bộ nhớ dữ liệu ngoài là bộ nhớ đọc-ghi được cho phép bởi các tín hiệu $\overline{RD}$ và $\overline{WR}$ ở các chân P3.7 và P3.6. Lệnh dùng để truy xuất bộ nhớ dữ liệu ngoài là MOVX, sử dụng hoặc con trỏ dữ liệu 16 bít DPTR hoặc R0, R1 làm thanh ghi chứa địa chỉ.

RAM có thể giao tiếp với 8051 theo cùng cách như ROM ngoại trừ đường $\overline{RD}$ nối với đường cho phép xuất $\overline{OE}$ của RAM và đường $\overline{WR}$ nối với đường $\overline{W}$ của RAM. Các kết nối với bus dữ liệu và bus địa chỉ giống như ROM bằng cách sử dụng port 0 và port 2 như ở phần trên dung lượng của RAM lên đến 64K

Giản đồ thời gian của thao tác đọc dữ liệu ở bộ nhớ dữ liệu ngoài được trình bày ở hình 2.10 cho lệnh MOVX A,@DPTR. Lưu ý là cả hai xung ALE và PSEN được bỏ qua ở nơi mà xung $\overline{RD}$ cho phép đọc RAM, nếu lệnh MOVX và RAM ngoài không bao giờ được dùng, các xung ALE luôn có tần số bằng 1/6 tần số của mạch dao động

Giản đồ thời gian của chu kỳ ghi (lệnh MOVX @DPTR, A) cũng tương tự ngoại trừ các xung $\overline{WR}$ ở mức thấp và dữ liệu được xuất ra ở port 0 ($\overline{RD}$ vẫn ở mức cao)