NGÔN NGỮ LẬP TRÌNH CHO 8051

2.1.1 GIỚI THIỆU

Vi điều khiển là một IC lập trình, vì vậy Vi điều khiển cần được lập trình trước khi sử dụng. Mỗi phần cứng nhất định phải có chương trình phù hợp kèm theo, do đó trước khi viết chương trình đòi hỏi người viết phải nắm bắt được cấu tạo phần cứng và các yêu cầu mà mạch điện cần thực hiện.

Chương trình là tập hợp các lệnh được tổ chức theo một trình tự hợp lí để giải quyết các yêu cầu của người lập trình.Tập hợp tất cả các lệnh gọi là tập lệnh. Họ Vi điều khiển MSC-51 đều có chung một tập lệnh, các Vi điều khiển được cải tiến sau này thường ít thay đổi hoặc mở rộng tập lệnh mà chú trọng phát triển phần cứng.

Lệnh của Vi điều khiển là các số nhị phân 8 bit hay còn gọi là mã máy. Các lệnh mang mã 00000000b đến 11111111b. Các mã lệnh này được đưa vào lưu trữ trong ROM, khi thực hiện chương trình Vi điều khiển đọc các mã lệnh này, giải mã, và thực hiện lệnh.

Vì các lệnh của Vi điều khiển có dạng số nhị phân quá dài và khó nhớ, hơn nữa việc gỡ lỗi khi chương trình phát sinh lỗi rất phức tạp và khó khăn. Khó khăn này được giải quyết với sự hỗ trợ của máy vi tính, người viết chương trình có thể viết chương trình cho vi điều khiển bằng các ngôn ngữ lập trình cấp cao, sau khi việc viết chương trình được hoàn tất, các trình biên dịch sẽ chuyển các câu lệnh cấp cao thành mã máy một cách tự động. Các mã máy này sau đó được đưa (nạp) vào bộ nhớ ROM của Vi điều khiển, Vi điều khiển sẽ tìm đến đọc các lệnh từ ROM để thực hiện chương trình . Bản thân máy tính không thể thực hiện các mã máy này vì chúng không phù hợp với phần cứng máy tính, muốn thực hiện phải có các chương trình mô phỏng dành riêng.

Chương trình cho Vi điều khiển có thể viết bằng C++,C,Visual Basic, hoặc bằng các ngôn ngữ cấp cao khác. Tuy nhiên hợp ngữ Assembler được đa số người dùng Vi điều khiển sử dụng để lập trình, vì lí do này chúng tôi chọn Assembly để hướng dẫn viết chương trình cho Vi điều khiển. Assembly là một ngôn ngữ cấp thấp, trong đó mỗi câu

lệnh chương trình tương ứng với một chỉ lệnh mà bộ xử lý có thể thực hiện được. Ưu điểm của hợp ngữ Assembly là: mã gọn,ít chiếm dung lượng bộ nhớ, hoạt động với tốc độ nhanh, và nó có hiệu suất tốt hơn so với các chương trình viết bằng ngôn ngữ bậc cao khác.

2.1.2 TỔNG QUAN VỀ NGÔN NGỮ ASSEMBLY

Assembly là một ngôn ngữ lập trình cấp thấp gần với ngôn ngữ máy, chương trình sau khi viết bằng assembly cần được chuyển đổi qua mã lệnh (hay còn gọi là mã máy) của vi điều khiển, quá trình chuyển đổi được thực hiện bằng chương trình dịch Assembler. Các mã lệnh sau đó được nạp vào Rom của vi điều khiển để thực hiện chương trình. Chương trình dịch Assembler được dùng phổ biến hiện nay là chương trình Macro Assembler sử dụng trên Dos.

Để soạn thảo chương trình có thể sử dụng Notepal hoặc bất cứ chương trình soạn thảo có sử dụng bộ kí tự chuẩn ASCII và lưu tên đuôi như sau: "tên.asm". Ngoài ra có thể sử dụng các phần mềm hỗ trợ soạn thảo dành riêng cho vi điều khiển đã tích hợp sẵn chương trình dịch Assembler.

2.1.3 MỘT SỐ QUI ƯỚC KHI LẬP TRÌNH VỚI HỢP NGỮ ASSEMBLER

a.Khi giới thiệu các câu lệnh viết bằng hợp ngữ, các câu lệnh cần được bao quát tất cả các trường hợp do đó có một số qui ước khi thiết lập cú pháp các lệnh như sau:

Tên qui ước	Tên qui ước đại diện cho	Ví dụ Lệnh sử dụng tên qui ước	Ví dụ khi sử dụng
Rn	Các thanh ghi ở các Bank thanh ghi Khi sử dụng thay n bằng các số từ 0 đến 7: R0, R1, R2, R3, R4, R5, R6, R7	Mov A,Rn	Mov A,R2
#data	Dữ liệu 8 bit, khi sử dụng data có thể viết dưới dạng : số nhị phân (Vd: #00110011b) số thập lục phân (Vd: #0A6H) số thập phân (Vd: #21)	Mov A,#data	Mov A,#20H

direct	***Ô nhớ có địa chỉ là direct***, direct được thay bằng địa chỉ từ 00H đến FFH khi viết chương trình.	Mov A,direct	Mov A,30H
@Ri	***Ô nhớ có địa chỉ gián tiếp***, đây là địa chỉ của một ô nhớ, địa chỉ này được xác định gián tiếp bằng giá trị của thanh ghi R0 hoặc R1 (chỉ được sử dụng hai thanh ghi R0 hoặc R1 để lưu giá trị này)	Mov A,@Ri	Mov A,@R1

#data: là giá trị cần thiết lập trong một ô nhớ, data được ghi trong chương trình assembly với qui định về cách viết số như ở bên dưới, các số này sau đó được trình biên dịch chuyển thành các số nhị phân tương ứng.

Ví dụ: khi ghi #95H đây là giá trị được thiết lập trong từng bit của ô nhớ.(các bit của ô nhớ có giá trị là 10010101).

Còn khi ghi 95H thì hiểu đây là ô nhớ có địa chỉ là 95H.

Đối với các ô nhớ được định tên bằng kí hiệu chẳng hạn P0,P1,A,B,TH0... thì được sử dụng tên đó thay cho địa chỉ cần sử dụng.

Ví dụ: hai lệnh sau đây là như nhau Mov TH0,#43H và Mov 8CH,#43H vì thanh ghi TH0 có địa chỉ là 8CH.

b. Qui định về cách viết số (data)

Trình biên dịch Assembler cho phép sử dụng các loại số sau trong chương trình:

Số Binary (số nhị phân): Số nhị phân khi viết cần thêm phía sau giá trị bằng kí tự "B". Các số này phải là số nhị phân 8 bit. Khi giá trị cần thiết lập là các giá trị cần cho từng bit trong byte thì dùng cách biểu diễn bằng số nhị phân

Ví dụ: khi cần thiết lập giá trị cho một byte mà các bit 0,1 xen kẽ nhau thì nên biểu diễn bằng số 01010101B cho dễ kiểm tra.

Hexadecimal (số thập lục phân-ghi tắt là hex): số hex khi viết cần thêm phía sau giá trị bằng kí tự "H" .Nếu sô hex bắt đầu là A,B,C,D,E,F thì cần thêm số "0" phía trước để trình biên dịch nhận biết được đó là số Hex, không lầm giá trị số với các kí tự chữ khác. Khi sử dụng các giá trị dành riêng cho một công việc nào đó, việc ghi nhớ bằng số nhị phân rất rắc rối và khó nhớ, khi đó số hex được sử dụng, vì số

hex là cách viết ngắn gọn của số nhị phân.

Ví dụ: 69H, 0A3H

Số Decimal (số thập phân): Số thập phân khi viết không cần cần thêm kí tự hoặc thêm sau giá trị bằng kí tự "D". Khi tính toán: cộng trừ nhân chia, nếu sử dụng số nhị phân hoặc số hex sẽ gây khó khăn cho người viết chương trình, trong trường hợp này số thập phân được sử dụng

Ví dụ: 45, 27, 68D

Chú ý: địa chỉ của các ô nhớ, của các bit nhớ, địa chỉ của ROM luôn được viết bằng số thập lục phân và cũng tuân theo qui tắc viết số như phía trên.

Để hiểu thêm về các loại số này và các cách chuyển đổi có thể xem thêm trong giáo trình kĩ thuật số tại địa chỉ http://www.codientu.info/ki_thuat_cdt/dien_tu/vi_mach_so/ hoặc các tài liệu về kĩ thuật số khác.

c.Kết thúc chương trình.

Sau khi chương trình hoàn tất phải kết thúc bằng câu lệnh ***END*** .Các câu lệnh này báo cho trình biên dịch biết phần kết thúc của chương trình, trình biên dịch bỏ qua tất cả các câu lệnh sau lệnh **END**

Tập lệnh trong Vi điều khiển được chia làm 5 nhóm:

- Nhóm lệnh di chuyển dữ liệu
- Nhóm lệnh số học
- Nhóm lệnh logic
- Nhóm lệnh rẽ nhánh
- Nhóm lệnh xử lí bit

Trước khi xem phần dưới, các bạn cần xem lại bài trước nắm rõ phần cứng, đặc biệt là vùng nhớ Ram của vi điều khiển. Chú ý các thuật ngữ sau:

Các byte RAM 8 bit của vi điều khiển được gọi là "*ô nhớ*", nếu các ô nhớ có chức năng đặc biệt thường được gọi là "*thanh ghi*", nếu là bit thì được gọi là "*bit nhớ*". dữ liệu của một ô nhớ là trạng thái (0 hoặc 1) cần thiết lập cho các bit của ô nhớ (8 bit)

2.2. NHÓM LỆNH DI CHUYỂN DỮ LIỆU

2.2.1. Lệnh chuyển dữ liệu từ một thanh ghi Rn vào thanh ghi A:

- Cú pháp: ***Mov A,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Chuyển dữ liệu của thanh ghi Rn vào thanh ghi A, dữ liệu trên thanh ghi Rn không đổi
- Ví dụ: Giả sử thanh ghi R5 mang dữ liệu với giá trị là 0A5H (10100101B)
  Lệnh ***Mov A,R5***
  Sau khi lệnh được thực hiện A mang dữ liệu giá trị A5H, Rn không đổi giá trị thanh ghi A trước khi thực hiện lệnh không cần quan tâm

2.2.2. Lệnh chuyển dữ liệu từ ô nhớ có địa chỉ direct vào thanh ghi A:

- Cú pháp: ***Mov A,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: chuyển dữ liệu của ô nhớ có địa chỉ bằng direct vào thanh ghi A.
- Ví dụ: Giả sử thanh ghi có địa chỉ 33H mang dữ liệu với giá trị là 09H (00001001B)
  Lệnh ***Mov A,33H***
  Sau khi lệnh được thực hiện A mang dữ liệu giá trị 09H

2.2.3. Lệnh chuyển dữ liệu từ ô nhớ có địa chỉ gián tiếp vào thanh ghi A:

- Cú pháp: ***Mov A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: chuyển dữ liệu của ô nhớ 'có địa chỉ bằng giá trị của thanh ghi Ri' vào thanh ghi A.
- Ví dụ: Giả sử trước khi thực hiện lệnh ô nhớ có địa chỉ 33H mang dữ liệu với giá trị là 09H (00001001B) và thanh ghi R1 được thiết lập giá trị là 33H
  Lệnh ***Mov A,@R1***
  Khi lệnh được thực hiện A nhận dữ liệu từ ô nhớ có vị trí bằng giá trị được

thiết lập trong thanh ghi R1, tức là A nhận dữ liệu từ ô nhớ có địa chỉ là 33H, chú ý: trước đó ô nhớ 33H mang dữ liệu là 09H.
Sau khi lệnh được thực hiện A mang giá trị là 09H (00001001B)

2.2.4. Lệnh đưa dữ liệu vào thanh ghi A

- Cú pháp: ***Mov A,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thiết lập dữ liệu cho thanh ghi A
- Ví dụ: Muốn thanh ghi A mang dữ liệu có giá trị là 56H ta thực hiện lệnh
  Mov A,#56H
  Sau khi lệnh được thực hiện A mang giá trị là 56H

2.2.5. Lệnh chuyển dữ liệu từ A vào thanh ghi Rn

- Cú pháp: ***Mov Rn,A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: chuyển dữ liệu từ thanh ghi A vào thanh ghi Rn (n=0-7)
- Ví dụ:
  Mov A,#56H
  Mov R1,A
  Sau khi các lệnh được thực hiện R1 mang giá trị là 56H

2.2.6. Lệnh chuyển dữ liệu từ một ô nhớ có địa chỉ direct vào thanh ghi Rn

- Cú pháp: ***Mov Rn,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: chuyển dữ liệu của ô nhớ có địa chỉ direct vào thanh ghi Rn (n=0-7)
- Ví dụ: giả sử ô nhớ 55H mang dữ liệu có giá trị là A3H
  Mov R4,55H
  Sau khi các lệnh được thực hiện R4 mang giá trị là A3H

2.2.7. Thiết đặt dữ liệu cho thanh ghi Rn

- Cú pháp: ***Mov Rn,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thiết đặt dữ liệu cho thanh ghi Rn
- Ví dụ: Muốn thanh ghi Rn mang dữ liệu có giá trị là 37H ta thực hiện lệnh

  Mov A,#37H

  Sau khi lệnh được thực hiện A mang giá trị là 37H

2.2.8. Lệnh chuyển dữ liệu từ thanh ghi A vào một ô nhớ có địa chỉ direct

- Cú pháp: ***Mov direct,A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: chuyển dữ liệu từ thanh ghi A vào một ô nhớ có địa chỉ direct.
- Ví dụ:

  Mov A,#77H

  Mov 69H,A

  Sau khi các lệnh được thực hiện ô nhớ 69H mang giá trị là 77H (giá trị của các bit được thiết lập trong ô nhớ 69H là 01110111B)

2.2.9. Lệnh chuyển dữ liệu từ thanh ghi Rn vào một ô nhớ có địa chỉ direct

- Cú pháp: ***Mov direct,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: chuyển dữ liệu từ thanh ghi A vào một ô nhớ có địa chỉ direct
- Ví dụ:

  Mov Rn,#78H

  Mov 7AH,Rn

  Sau khi các lệnh được thực hiện ô nhớ 7AH mang giá trị là 78H

2.2.10. Lệnh chuyển dữ liệu từ một ô nhớ có địa chỉ direct này vào một ô nhớ có địa chỉ direct khác

- Cú pháp: ***Mov direct,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte

- Thời gian thực hiện: 1 chu kì máy
- Công dụng: chuyển dữ liệu từ ô nhớ có địa chỉ direct này vào một ô nhớ có địa chỉ direct khác
- Ví dụ:giả sử thanh ghi 20H mang dữ liệu có giá trị là FFH

  Mov 22H,20H

  Sau khi lệnh được thực hiện thanh ghi 22H mang giá trị là FFH

2.2.11. Lệnh đưa dữ liệu vào ô nhớ có địa chỉ direct

- Cú pháp: ***Mov direct,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: thiết lập dữ liệu cho ô nhớ có địa chỉ direct
- Ví dụ:

  Mov 52H,#43H

  Sau khi các lệnh được thực hiện ô nhớ 52H mang giá trị là 43H

2.2.12. Lệnh chuyển dữ liệu từ một ô nhớ có địa chỉ gián tiếp vào ô nhớ có địa chỉ direct

- Cú pháp: ***Mov direct,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Chuyển dữ liệu của ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri vào ô nhớ có địa chỉ direct
- Ví dụ:

  Mov 30H,#46H

  Mov R0,#30H

  Mov 23H, @R0

  Sau khi các lệnh được thực hiện ô nhớ 23H mang giá trị là 46H

2.2.13. Lệnh chuyển dữ liệu từ thanh ghi A vào ô nhớ có địa chỉ gián tiếp

- Cú pháp: ***Mov @Ri,A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy

- Công dụng: Chuyển dữ liệu của thanh ghi A vào ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri
- Ví dụ:

  Mov A,#33H
  Mov R0,#22H
  Mov @R0,A
  Sau khi lệnh được thực hiện ô nhớ 22H mang giá trị là 33H

2.2.14. Lệnh chuyển dữ liệu từ một ô nhớ có địa chỉ direct vào ô nhớ có địa chỉ gián tiếp

- Cú pháp: ***Mov @Ri,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Chuyển dữ liệu của ô nhớ có địa chỉ direct vào ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri
- Ví dụ:

  Mov 4BH,#2AH
  Mov R0,#2AH
  Mov @R0,4BH
  Sau khi lệnh được thực hiện ô nhớ 2AH mang giá trị là 2AH

2.2.15. Lệnh đưa dữ liệu vào ô nhớ có địa chỉ gián tiếp

- Cú pháp: ***Mov @Ri,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Thiết đặt dữ liệu cho ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri
- Ví dụ:

  Mov R0,#3BH
  Mov @R0,#27H
  Sau khi lệnh được thực hiện ô nhớ 3BH mang giá trị là 27H

2.2.16. Lệnh đưa dữ liệu vào con trỏ dữ liệu DPTR

- Cú pháp: ***Mov DPTR,#data16***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Thiết đặt dữ liệu cho con trỏ dữ liệu với dữ liệu 16 bit, thực chất dữ liệu được lưu ở hai thanh ghi DPL (byte thấp-địa chỉ byte 82H) và DPH (byte cao-địa chỉ byte 83H).
- **Ví *dụ*: *Mov DPTR,#3A5FH***
  Sau khi lệnh được thực hiện DPTR mang giá trị là 3A5FH
  DPL mang giá trị 5FH và DPL mang giá trị 3AH

2.2.17. Lệnh trao đổi dữ liệu giữa ô nhớ có địa chỉ direct với thanh ghi A

- Cú pháp: ***XCH A,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Trao đổi dữ liệu của thanh ghi A với ô nhớ có địa chỉ direct, tức là sau khi thực hiện lệnh ô nhớ có địa chỉ direct mang dữ liệu của thanh ghi A trước đó và thanh ghi A mang dữ liệu của ô nhớ có địa chỉ direct.
- Ví dụ: *Mov A,#0FAH*
  Mov 50H,#60H
  XCH A,50H
  Kết quả :A mang giá trị là 60H
  50H mang giá trị là 0FAH

2.2.18. Lệnh trao đổi dữ liệu giữa thanh ghi Rn và thanh ghi A

- Cú pháp: ***XCH A,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Trao đổi dữ liệu của thanh ghi A với thanh ghi Rn.

2.2.19. Lệnh trao đổi dữ liệu giữa thanh ghi có địa chỉ gián tiếp và thanh ghi A

- Cú pháp: ***XCH A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy

- Công dụng: Trao đổi dữ liệu của thanh ghi A với ô nhớ có địa chỉ bằng giá trị lưu giữ trong thanh ghi Ri

2.2.20. Lệnh trao đổi dữ liệu 4 bit giữa thanh ghi có địa chỉ gián tiếp và thanh ghi A

- Cú pháp: ***XCHD A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Trao đổi dữ liệu của 4 bit thấp ở thanh ghi A với dữ liệu của 4 bit thấp ở ô nhớ có địa chỉ bằng giá trị lưu giữ trong thanh ghi Ri

2.2.21. Lệnh truy xuất dữ liệu từ ROM nội

- Cú pháp: ***MovC A,@A+DPTR***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Chuyển dữ liệu từ bộ nhớ ROM có địa chỉ bằng giá trị của A cộng với DPTR vào thanh ghi A

Các lệnh còn lại trong nhóm lệnh di chuyển

MovC A,@A+PC
MovC A,@i
MovX A,@DPTR
MovX A,@Ri
MovX @DPTR,A
PUSH direct
POP direct

sẽ được khảo sát ở các bài khác

2.3 Nhóm lệnh số học.

Để theo dõi các lệnh trong phần này, các bạn xem lại phần: *các ô nhớ có chức năng đặc biệt* và chú ý phần ***1.1.11*** *Thanh ghi trạng thái chương trình PSW*

Phần phụ chú: ảnh hưởng của phép cộng và trừ lên thanh trạng thái PSW.

Cờ nhớ C:

- 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.

Phép cộng xảy ra tràn là phép cộng mà kết quả lớn hơn 255 (hay FFH hay 11111111b), lúc này C=1

Ví dụ: phép cộng không tràn

Số cộng	38H	56	0 0 1 1 1 0 0 0 b
Số cộng	+3AH	58	0 0 1 1 1 0 1 0 b
Kết quả	72H	114	0 1 1 1 0 0 1 0 b
Cờ nhớ C	0		0

Phép cộng tràn

Số cộng	6CH	108	0 1 1 0 1 1 0 0 b
Số cộng	+9FH	159	1 0 0 1 1 1 1 1 b
Kết quả	10BH	267	1 0 0 0 0 1 0 1 1 b
Cờ nhớ C	1		1

Phần được tô màu xanh là 8 bit của thanh ghi A sau khi kết quả được thực hiện, phần màu đỏ trong kết quả là giá trị bị tràn, giá trị này không lưu ở thanh ghi A mà lưu ở thanh ghi PSW, tại cờ C

Số trừ	9FH	159	1 0 0 1 1 1 1 1 b
Số bị trừ	-6CH	108	0 1 1 0 1 1 0 0 b
Kết quả	33H	51	0 0 1 1 0 0 1 1 b
Cờ nhớ C	0		0

Số trừ	6CH	108	0 1 1 0 1 1 0 0 b
Số bị trừ	-9FH	159	1 0 0 1 1 1 1 1 b

Kết quả	CDH	-51	1 1 0 0 1 1 0 1 b
Cờ nhớ C	1		1 -phép trừ trên có số mượn

2.3.1. Lệnh cộng dữ liệu trên thanh ghi A với dữ liệu trên thanh ghi Rn:

- Cú pháp: ***Add A,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Cộng giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu trên thanh ghi Rn, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
Mov   A,#20H
Mov   R1,#08H
Add   A,R1
```

Kết quả : A có giá trị là 28H

R1 vẫn giữ nguyên giá trị là 08H

Cờ C = 0

Vidu2:

```
Mov   A,#0E9H
Mov   R6,#0BAH
Add   A,R6
```

Kết quả : A = #0A3h

R6 = #0BAh

Cờ C = 1

2.3.2. Lệnh cộng dữ liệu trên thanh ghi A với dữ liệu ở ô nhớ có địa chỉ direct:

- Cú pháp: ***Add A,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy

- Công dụng: Cộng giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu trên ô nhớ có địa chỉ direct, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
Mov     50h,#20H
Mov     A,#0E8H
Add     A,50H
```

Kết quả : A = #08H
50H = #20H
C = 1

2.3.3. Lệnh cộng dữ liệu trên thanh ghi A với dữ liệu của ô nhớ có địa chỉ gián tiếp:

- Cú pháp: ***Add A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Cộng giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu của ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
AC  =  1    ;cờ C đang mang giá trị 1
            Mov     50H,#60H
            Mov     R2,#50H
            Mov     A,#01H
            Add     A,@R2
Kết quả  :  A = #61H
            R2 = #50H
            C = 0           ;cờ C mang giá trị 0
```

2.3.4. Lệnh cộng dữ liệu trên thanh ghi A với dữ liệu xác định:

- Cú pháp: ***Add A,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy

- Công dụng: Cộng giá trị dữ liệu trên thanh ghi A với một giá trị xác định, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
Mov     A,#05h
Add     A,#06h
```

Kết quả : A = #0Bh
C = 0

2.3.5. Lệnh cộng dữ liệu trên thanh ghi A với dữ liệu trên thanh ghi Rn có số nhớ ở cờ C:

- Cú pháp: ***AddC A,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Cộng giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu trên thanh ghi Rn và cộng thêm giá trị của số nhớ trên cờ C, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
C = 1
Mov     A,#08h
Mov     R1,#10h
Addc    A,R1
```

Kết quả : A = #19h ;cộng cả cờ C
R1 = #10h
C =0

2.3.6. Lệnh cộng dữ liệu trên thanh ghi A với dữ liệu ở ô nhớ có địa chỉ direct và giá trị số nhớ ở cờ C:

- Cú pháp: ***AddC A,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Cộng giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu của ô nhớ có địa chỉ direct và cộng thêm giá trị của số nhớ trên cờ C , sau khi thực hiện

lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW

- Ví dụ: C = 0

```
Mov   A,#0A5h
Mov   10h,#96h
Addc  A,10h
```

Kết quả : A = #3Bh
10h = #96h
C =1

2.3.7. Lệnh cộng dữ liệu trên thanh ghi A với dữ liệu của ô nhớ có địa chỉ gián tiếp và số nhớ ở cờ C:

- Cú pháp: ***AddC A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Cộng giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu của ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri và cộng thêm giá trị của số nhớ trên cờ C, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
C = 1
Mov   A,#05h
Mov   50h,#10h
Mov   R2,#50h
Addc  a,@R2
```

Kết quả : A = #16h
C = 0

2.3.8. Lệnh cộng dữ liệu trên thanh ghi A với dữ liệu xác định và số nhớ ở cờ C:

- Cú pháp: ***AddC A,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy

- Công dụng: Cộng giá trị dữ liệu trên thanh ghi A với giá trị xác định và cộng thêm giá trị của số nhớ trên cờ C, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
C = 1
Mov   A,#05h
Addc  A,#16h
```

Kết quả : A = #1Ch
C = 0

2.3.9. Lệnh trừ dữ liệu trên thanh ghi A với dữ liệu trên thanh ghi Rn và số nhớ ở cờ C:

- Cú pháp: ***SubB A,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Trừ giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu trên thanh ghi Rn và trừ cho giá trị nhớ trên cờ C, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
C =  1
Mov   A,#0E5h
Mov   R3,#9Fh
Subb  A,R3
```

kết quả : A = 45h
C = 0

2.3.10. Lệnh trừ dữ liệu trên thanh ghi A với dữ liệu ở ô nhớ có địa chỉ direct và số nhớ ở cờ C:

- Cú pháp: ***SubB A,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Trừ giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu của ô nhớ có địa chỉ direct và trừ cho giá trị nhớ trên cờ C , sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW

- Ví dụ:

```
C =  0
Mov   A,#0E5h
Mov    05h,#9Fh
Subb   A,05h
```

kết quả : A = 46h
C = 0

2.3.11. Lệnh trừ dữ liệu trên thanh ghi A với dữ liệu của ô nhớ có địa chỉ gián tiếp và số nhớ ở cờ C:

- Cú pháp: ***SubB A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Trừ giá trị dữ liệu trên thanh ghi A với giá trị dữ liệu của ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri và trừ cho giá trị nhớ trên cờ C, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW
- Ví dụ:

```
C =  1
Mov   A,#0E5h
Mov   4Fh,#50h
Mov    R3,#4Fh
Subb   A,@R3
```

kết quả : A = 94h
C = 0

2.3.12. Lệnh trừ dữ liệu trên thanh ghi A với dữ liệu xác định và số nhớ ở cờ C:

- Cú pháp: ***SubB A,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Trừ giá trị dữ liệu trên thanh ghi A với giá trị xác định và trừ thêm giá trị nhớ trên cờ C, sau khi thực hiện lệnh kết quả được lưu ở thanh ghi A. Lệnh này có ảnh hưởng đến thanh thanh trạng thái PSW

- Ví dụ:

```
C =  0
Mov   A,#05h
Subb   A,#4Fh
```

kết quả : A = 0B6h
C = 1

2.3.13.Lệnh tăng giá trị dữ liệu trên thanh ghi A lên 1 đơn vị:

- Cú pháp: ***Inc A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Tăng giá trị dữ liệu lưu giữ trên thanh ghi A lên 1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW

Ví dụ:
```
Mov   A,#05h
Inc     A
```
Kết quả : A = #06h

2.3.14.Lệnh tăng giá trị dữ liệu trên thanh ghi Rn lên 1 đơn vị:

- Cú pháp: ***Inc Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Tăng giá trị dữ liệu lưu giữ trên thanh ghi Rn lên 1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW
- Ví dụ:

```
Mov   R7,#0Fh
Inc     R7
```
Kết quả : R7 = #10h

2.3.15.Lệnh tăng giá trị dữ liệu ở ô nhớ có địa chỉ direct lên 1 đơn vị:

- Cú pháp: ***Inc direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy

- Công dụng: Tăng giá trị dữ liệu ở một ô nhớ có địa chỉ direct lên 1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW
- Ví dụ:

```
Mov   50h,#0FFh
Inc    50h
```

Kết quả : 50h = #00

2.3.16.Lệnh tăng giá trị dữ liệu ở ô nhớ có địa chỉ gián tiếp lên 1 đơn vị:

- Cú pháp: ***Inc @Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Tăng giá trị dữ liệu ở ô nhớ có địa chỉ bằng giá trị dữ liệu trên Ri lên 1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW
- Ví dụ:

```
Mov   0Fh,#05h
Mov   R0,#0Fh
Inc    @R0
```

Kết quả : R0 = #06h
0Fh = #05h

2.3.17.Lệnh tăng giá trị của con trỏ dữ liệu DPTR lên 1 đơn vị:

- Cú pháp: ***Inc DPTR***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Tăng giá trị dữ liệu của thanh ghi con trỏ dữ liệu DPTR lên 1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW
- Ví dụ:

```
Mov   DPTR,#5Fh
Inc    DPTR
```

Kết quả : DPTR = #060h

2.3.18.Lệnh giảm giá trị dữ liệu trên thanh ghi A xuống 1 đơn vị:

- Cú pháp: ***Dec A***

- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Giảm giá trị dữ liệu lưu giữ trên thanh ghi A xuống 1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW
- Ví dụ:

```
Mov   A,#05h
Dec   A
```

Kết quả : A = #04h

2.3.19.Lệnh giảm giá trị dữ liệu trên thanh ghi Rn xuống 1 đơn vị:

- Cú pháp: ***Dec Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Giảm giá trị dữ liệu lưu giữ trên thanh ghi Rn xuống 1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW
- Ví dụ:

```
Mov   R6,#0Fh
Dec   R6
```

Kết quả : R6 = #0Eh

2.3.20.Lệnh giảm giá trị dữ liệu ở ô nhớ có địa chỉ direct xuống 1 đơn vị:

- Cú pháp: ***Dec direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Giảm giá trị dữ liệu ở ô nhớ có địa chỉ direct xuống 1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW
- Ví dụ:

```
Mov   7Fh,#0
Dec   7Fh
```

Kết quả : 7Fh = #0FFh

2.3.21.Lệnh giảm giá trị dữ liệu ở ô nhớ có địa chỉ gián tiếp xuống 1 đơn vị:

- Cú pháp: ***Dec @Ri***

- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Giảm giá trị dữ liệu ở ô nhớ có địa chỉ bằng giá trị dữ liệu trên Ri xuống1 đơn vị, không ảnh hưởng đến các cờ nhớ trên PSW
- Ví dụ:

```
Mov   60h,#05h
Mov   R1,#60h
Dec   @R1
```

Kết quả : R1 = #04h
60h = #05h

2.3.22.Lệnh nhân thanh ghi A với thanh ghi B:

- Cú pháp: ***Mul AB***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 4 chu kì máy
- Công dụng: Nhân hai dữ liệu là số nguyên không dấu ở thanh ghi A với thanh ghi B, kết quả là một dữ liệu 16 bit. Byte thấp của kết quả lưu ở thanh ghi A và byte cao của kết quả lưu ở thanh ghi B. Nếu tích số lớn hơn 255(0FFH), cờ tràn OV ở thanh trạng thái PSW được thiết lập lên 1, ngược lại nếu tích số nhỏ hơn 255(0FFH),cờ tràn OV được thiết lấp về 0. Cờ nhớ C luôn ở giá trị 0.
- Ví dụ:

```
Mov   A,#0B9h
Mov   B,#F7h
Mul   AB
```

Kết quả : A = #7Fh
B = #0B2h

2.3.23.Lệnh chia thanh ghi A với thanh ghi B:

- Cú pháp: ***Div AB***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 4 chu kì máy
- Công dụng: Chia hai dữ liệu là số nguyên không dấu ở thanh ghi A với thanh ghi B, dữ liệu ở thanh ghi A là số chia còn ở thanh ghi B là số bị chia, kết quả là một dữ liệu 8 bit được lưu ở thanh ghi A.số dư lưu trữ trong thanh ghi B Cờ

nhớ C luôn ở giá trị 0.

Cờ tràn OV được thiết lập giá trị 1 khi thanh ghi B mang giá trị là 00H-phép chia không thể thực hiện.

- Ví dụ:
```
Mov A,#50h
Mov B,#10h
DIV AB
```
Kết quả : A = #5h
B = #0h

2.3.24.Lệnh hiệu chỉnh thập phân nội dung của thanh ghi A đối với phép cộng:

- Cú pháp: ***DA A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 4 chu kì máy
- Công dụng: hiệu chỉnh dữ liệu là giá trị lưu giữ ở thanh ghi A từ số Hex (số nhị phân) thành số BCD (số thập phân viết dưới dạng nhị phân). Lí do có lệnh hiệu chỉnh này vì khi cộng hai giá trị là số BCD bằng các lệnh cộng, vi điều khiển chỉ hiểu hai số cộng là số nhị phân bình thường, kết quả sau lệnh cộng là một số nhị phân bình thường, không phải là một số BCD, vì vậy kết quả cần được hiệu chỉnh để dữ liệu cuối là một số BCD. Khi thực hiện lệnh, cờ nhớ C được xác lập lên 1 nếu phép cộng có kết quả vượt qua 99(số BCD). Kết quả cuối cùng, số BCD có hàng đơn vị nằm ở 4 bit thấp trên thanh ghi A, hàng chục ở 4 bit cao của thanh ghi A, hàng trăm là 1 nếu cờ C mang giá trị 1, là 0 nếu cờ C mang giá trị 0.
- Ví dụ 1:
```
Mov A,#10h
DA A
```
Kết quả : A = #10h
-
Ví dụ 2:
```
Mov A,#0Eh
DA A
```
Kết quả : A = #14h

2.4Nhóm lệnh logic

2.4.1. Lệnh And dữ liệu ở thanh ghi A với dữ liệu ở thanh ghi Rn:

- Cú pháp: ***ANL A,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là: 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic AND dữ liệu ở thanh ghi A với dữ liệu ở thanh ghi Rn, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#0Fh
mov   R1,#0F0h
ANL   A,R1
```

Kết quả : A = #0H

2.4. 2. Lệnh And dữ liệu trên thanh ghi A với dữ liệu của ô nhớ có địa chỉ direct:

- Cú pháp: ***ANL A,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic AND dữ liệu ở thanh ghi A với dữ liệu ở ô nhớ có địa chỉ direct, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#0FFh
mov   10h,#010h
ANL   A,10h
```

Kết quả : A = #010h

2.4.3. Lệnh And dữ liệu trên thanh ghi A với dữ liệu của ô nhớ gián tiếp:

- Cú pháp: ***ANL A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic AND dữ liệu ở thanh ghi A với dữ liệu của ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#0Fh
mov   70h,#0E1h
mov   R1,#070h
```

ANL A,@R1

Kết quả : A = #01h

2.4.*4. Lệnh And dữ liệu trên thanh ghi A với dữ liệu xác định:*

- Cú pháp: ***ANL A,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic AND dữ liệu ở thanh ghi A với dữ liệu cho trước, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

mov A,#0Eh
ANL A,#11h

Kết quả : A = #00

2.4.5. Lệnh And dữ liệu của ô nhớ có địa chỉ direct với dữ liệu trên thanh ghi A:

- Cú pháp: ***ANL direct,A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic AND dữ liệu ở thanh ghi A với dữ liệu của ô nhớ có địa chỉ direct, kết quả được lưu trữ ở ô nhớ có địa chỉ direct.
- Ví dụ:

mov A,#08h
mov R1,#0F7h
ANL R1,A

Kết quả : R1 = #0

2.4.6. Lệnh And dữ liệu trên ô nhớ có địa chỉ direct với dữ liệu xác định:

- Cú pháp: ***ANL direct,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic AND dữ liệu của ô nhớ có địa chỉ direct với dữ liệu cho trước, kết quả được lưu trữ ở ô nhớ có địa chỉ direct.

- Ví dụ:

```
mov   R1,#0F7h
ANL   R1,#1Fh
```

Kết quả : R1 = #017h

2.4.7. Lệnh OR dữ liệu ở thanh ghi A với dữ liệu ở thanh ghi Rn:

- Cú pháp: ***ORL A,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là: 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic OR dữ liệu ở thanh ghi A với dữ liệu ở thanh ghi Rn, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#0Fh
mov   R1,#0F0h
ORL   A,R1
```

Kết quả : A = #0FFh

2.4. 8. Lệnh OR dữ liệu trên thanh ghi A với dữ liệu của ô nhớ có địa chỉ direct:

- Cú pháp: ***ORL A,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic OR dữ liệu ở thanh ghi A với dữ liệu của ô nhớ có địa chỉ direct, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#0Eh
mov   50h,#0F0h
ORL   A,50h
```

Kết quả : A = #0FEh

2.4.9. Lệnh OR dữ liệu trên thanh ghi A với dữ liệu của ô nhớ gián tiếp:

- Cú pháp: ***ORL A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic OR dữ liệu ở thanh ghi A với dữ liệu của ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri, kết quả được lưu trữ ở thanh ghi A

- Ví dụ:

```
mov   A,#18h
mov   30h,#0F0h
mov   R1,#30h
ORL   A,@R1
```

Kết quả : A = #0F8h

2.4.10. Lệnh OR dữ liệu trên thanh ghi A với dữ liệu xác định:

- Cú pháp: ***ORL A,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic OR dữ liệu ở thanh ghi A với dữ liệu cho trước, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#00h
ORL   A,#10h
```

Kết quả : A = #010h

2.4.11. Lệnh OR dữ liệu của ô nhớ có địa chỉ direct với dữ liệu trên thanh ghi A:

- Cú pháp: ***ORL direct,A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic OR dữ liệu ở thanh ghi A với dữ liệu của ô nhớ có địa chỉ direct, kết quả được lưu trữ ở ô nhớ có địa chỉ direct.
- Ví dụ:

```
mov   A,#0Fh
mov   5Fh,#0F0h
ORL   5Fh,A
```

Kết quả : 5Fh = #0FFh

2.4.12. Lệnh OR dữ liệu trên ô nhớ có địa chỉ direct với dữ liệu xác định:

- Cú pháp: ***ORL direct,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte

- Thời gian thực hiện: 2 chu kì máy
- Công dụng: thực hiện phép logic OR dữ liệu của ô nhớ có địa chỉ direct với dữ liệu cho trước, kết quả được lưu trữ ở ô nhớ có địa chỉ direct.
- Ví dụ:

```
mov   60h,#0F0h
ORL   60h,#1Fh
```

Kết quả : 60h = #0FFh

2.4.13. Lệnh EX-OR dữ liệu ở thanh ghi A với dữ liệu ở thanh ghi Rn:

- Cú pháp: ***XRL A,Rn***
- Lệnh này chiếm dung lượng bộ nhớ ROM là: 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic EX-OR dữ liệu ở thanh ghi A với dữ liệu ở thanh ghi Rn, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#0F2h
mov   R3,#0E0h
XRL   A,R3
```

Kết quả : A = #12h

2.4.14. Lệnh EX-OR dữ liệu trên thanh ghi A với dữ liệu của ô nhớ có địa chỉ direct:

- Cú pháp: ***XRL A,direct***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic EX-OR dữ liệu ở thanh ghi A với dữ liệu của ô nhớ có địa chỉ direct, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#012h
mov   10h,#0E0h
XRL   A,10h
```

Kết quả : A = #0F2h

2.415. Lệnh EX-OR dữ liệu trên thanh ghi A với dữ liệu của ô nhớ gián tiếp:

- Cú pháp: ***XRL A,@Ri***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic EX-OR dữ liệu ở thanh ghi A với dữ liệu của ô nhớ có địa chỉ bằng giá trị của thanh ghi Ri, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#08h
mov   10h,#0E9h
mov   R0,#10h
XRL   A,@R0
```

Kết quả : A = #0E1h

2.4.16. Lệnh EX-OR dữ liệu trên thanh ghi A với dữ liệu xác định:

- Cú pháp: ***XRL A,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic EX-OR dữ liệu ở thanh ghi A với dữ liệu cho trước, kết quả được lưu trữ ở thanh ghi A
- Ví dụ:

```
mov   A,#12h
XRL   A,#12h
```

Kết quả : A = #0

2.4.17. Lệnh EX-OR dữ liệu của ô nhớ có địa chỉ direct với dữ liệu trên thanh ghi A:

- Cú pháp: ***XRL direct,A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thực hiện phép logic EX-OR dữ liệu ở thanh ghi A với dữ liệu của ô nhớ có địa chỉ direct, kết quả được lưu trữ ở ô nhớ có địa chỉ direct.
- Ví dụ:

```
mov   A,#0F2h
mov   50h,#0E0h
```

XRL 50h,A

Kết quả : 50h = #12h

2.4.18. Lệnh EX-OR dữ liệu trên ô nhớ có địa chỉ direct với dữ liệu xác định:

- Cú pháp: ***XRL direct,#data***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: thực hiện phép logic EX-OR dữ liệu của ô nhớ có địa chỉ direct với dữ liệu cho trước, kết quả được lưu trữ ở ô nhớ có địa chỉ direct.
- Ví dụ:

mov 50h,#0E0h

XRL 50h,#01h

Kết quả : 50h = #0E1h

2.4.19. Lệnh bù giá trị dữ liệu trên thanh ghi A:

- Cú pháp: ***CPL A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: lấy bù giá trị lưu giữ ở thanh ghi A, các bit có giá trị là 1 chuyển thành 0 và ngược lại các bit có giá trị là 0 chuyển thành 1.
- Ví dụ: *mov A,#01100111b ;(tương đương 67h)*

CPL A

Kết quả : A = #10011000b (tương đương 98h)

2.4.20. Lệnh xóa dữ liệu trên thanh ghi A:

- Cú pháp: ***CLR A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: tất cả các bit của thanh ghi A đều được xác lập giá trị 0 .
- Ví dụ:

mov A,#01100111b

CLR A

Kết quả : A = #0

2.4.21. Lệnh xoay trái dữ liệu trên thanh ghi A:

- Cú pháp: ***RL A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy

Công dụng: thanh ghi A gồm tám bit A7 A6 A5 A4 A3 A2 A1 A0. Khi thực hiện lệnh xoay trái ***RL A*** giá trị của các bit được chuyển trang bit ở bên trái nó, giá trị của bit A0 chuyển sang bit A1, giá trị của bit A1 chuyển sang bit A2, tương tự với các bit còn lại, và giá trị của bit A7 chuyển sang bit A0. Minh họa các bit trong thanh ghi A khi thực hiện lệnh như trong hình dưới

A7	**A6**	**A5**	**A4**	**A3**	**A2**	**A1**	**A0**
	<—	<—	<—	<—	<—	<—	<—

A7 ——————————————————— ——————>**A0**

Các bit ở thanh ghi A

Quá trình xoay dữ liệu từ A0 đến A6

Giá trị dữ liệu A7 chuyển sang bit A0

- Ví dụ:

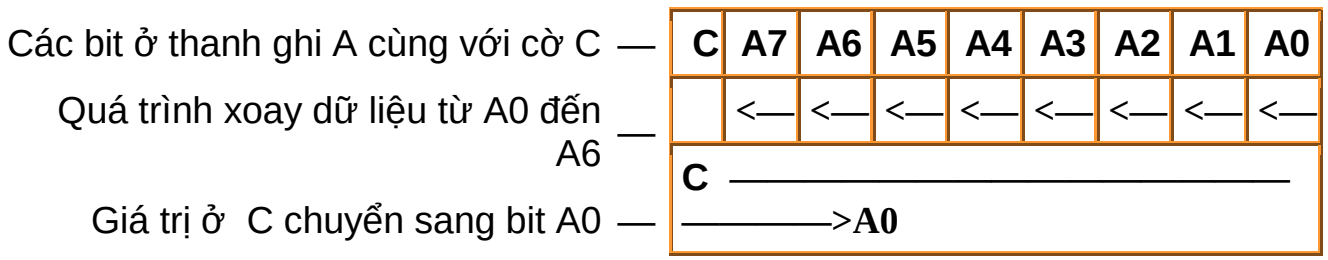
```
Mov   A,#01001001b
RL    A
```

Kết quả sau khi các lệnh được thực hiện A mang giá trị là 10010010b

	Giá trị thanh ghi A
Trước khi thực hiện lệnh xoay trái	0 1 0 0 1 0 0 1
Sau khi thực hiện lệnh xoay trái	1 0 0 1 0 0 1 0

2.4.22. Lệnh xoay trái dữ liệu trên thanh ghi A cùng với cờ nhớ C:

- Cú pháp: ***RLC A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thanh ghi A gồm tám bit A7 A6 A5 A4 A3 A2 A1 A0. Khi thực hiện lệnh xoay trái A với cờ nhớ RLC A giá trị của các bit được chuyển trang bit ở bên trái nó, giá trị của bit A0 chuyển sang bit A1, giá trị của bit A1 chuyển sang bit A2, tương tự với các bit còn lại, và giá trị của bit A7 chuyển sang cờ nhớ C, giá trị ở cờ nhớ C chuyển sang bit A0



- Ví dụ: giả sử cờ nhớ C đang mang giá trị 1

  Mov A,#11001001b

  RLC A

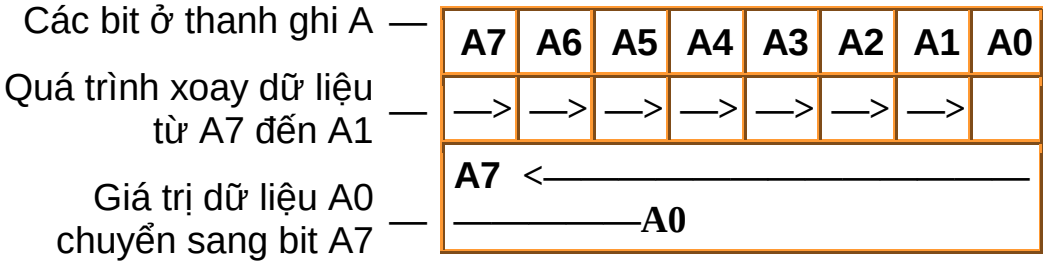
  Kết quả sau khi các lệnh được thực hiện A mang giá trị là 10010011b và C mang giá trị 1

	Cờ nhớ C	Giá trị thanh A
Trước khi thực hiện lệnh xoay trái với C	1	1 1 0 0 1 0 0 1
Sau khi thực hiện lệnh xoay trái với C	1	1 0 0 1 0 0 1 1

2.4.23. Lệnh xoay phải dữ liệu trên thanh ghi A:

- Cú pháp: ***RR A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thanh ghi A gồm tám bit A7 A6 A5 A4 A3 A2 A1 A0. Khi thực hiện lệnh xoay phải ***RR A*** giá trị của các bit được chuyển trang bit ở bên phải nó, giá trị của bit A7 chuyển sang bit A6, giá trị của bit A6 chuyển sang bit

A5, tương tự với các bit còn lại, và giá trị của bit A0 chuyển sang bit A7. Minh họa các bit trong thanh ghi A khi thực hiện lệnh như trong hình dưới



- Ví dụ:

  Mov A,#01001001b

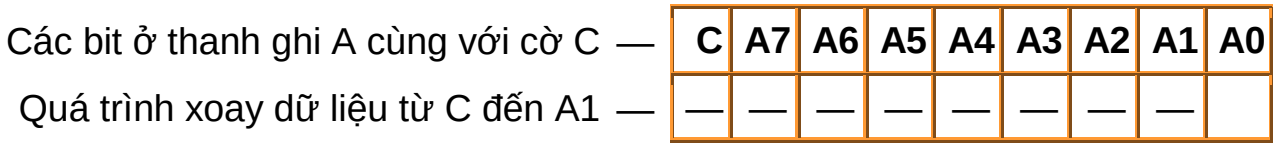
  RL A

  Kết quả sau khi các lệnh được thực hiện A mang giá trị là 10100100b

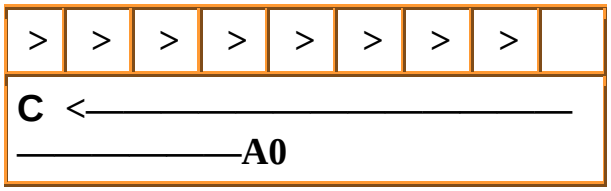
	Giá trị thanh A
Trước khi thực hiện lệnh xoay phải	0 1 0 0 1 0 0 1
Sau khi thực hiện lệnh xoay phải	1 0 1 0 0 1 0 0

2.4.24. Lệnh xoay phải dữ liệu trên thanh ghi A cùng với cờ nhớ C:

- Cú pháp: ***RRC A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thanh ghi A gồm tám bit A7 A6 A5 A4 A3 A2 A1 A0. Khi thực hiện lệnh xoay phải A với cờ nhớ -***RRC A*** -giá trị của các bit được chuyển trang bit ở bên phải nó, giá trị của bit A7 chuyển sang bit A6, giá trị của bit A6 chuyển sang bit A5, tương tự với các bit còn lại, và giá trị của bit A0 chuyển sang cờ nhớ C, giá trị ở cờ nhớ C chuyển sang bit A7



Giá trị ở A0 chuyển sang bit C —



- Ví dụ: giả sử cờ nhớ C đang mang giá trị 1

  Mov A,#11001001b

  RLC A

  Kết quả sau khi các lệnh được thực hiện A mang giá trị là 11100100b và C mang giá trị 1

	Cờ nhớ C	Giá trị thanh A
Trước khi thực hiện lệnh xoay trái với C	1	1 1 0 0 1 0 0 1
Sau khi thực hiện lệnh xoay trái với C	1	1 1 1 0 0 1 0 0

2.4.25. Lệnh xoay 4 bit trên thanh ghi A:

- Cú pháp: ***SWAP A***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: hoán chuyển dữ liệu ở 4 bit thấp lên 4 bit cao và 4 bit cao xuống 4 bit thấp

Các bit ở thanh ghi A —	**A7**	**A6**	**A5**	**A4**	**A3**	**A2**	**A1**	**A0**
Dữ liệu trước khi thực hiên lệnh —	**X7**	**X6**	**X5**	**X4**	**X3**	**X2**	**X1**	**X0**
Dữ liệu sau khi thực hiên lệnh —	**X3**	**X2**	**X1**	**X0**	**X7**	**X6**	**X5**	**X4**

- Ví dụ:

  mov A,#0E7h

  SWAP A

  Kết quả : A = # 7Eh

Nhóm lệnh rẽ nhánh.

Phần này liên quan đến các câu lệnh được lưu giữ trên ROM, vì vậy cần xem lại phần bộ nhớ ROM trước khi xem phần này.

Phần phụ chú:

Nhãn: Kí hiệu: ***rel***

Nhãn là một chuỗi kí tự do người dùng tự đặt dùng để đánh dấu các đoạn chương trình, nhãn này biểu thị địa chỉ của lệnh khi được lưu trên ROM.

Nhãn chỉ được bắt đầu bằng một kí tự chữ hoặc dấu "_" ,không được bắt đầu bằng số ,không có khoảng trắng và kết thúc bằng dấu hai chấm ":"

Trong chương trình nhãn không được đặt trùng tên với nhau, và không được trùng với các từ khóa mà chương trình đã sử dụng.

Ví dụ : Các nhãn đúng X1: ;S_2: ;_5:s10: ;...

Các nhãn sai 1X: ; S_2 ;S 5: ;DW: ,LPT :...

Chương trình con: là những đoạn chương trình thực hiện một số lệnh nào đó và được viết ngoài chương trình chính, các chương trình con này được đặt tên bằng một nhãn và kết thúc bằng lệnh RET, chương trình con có thể gọi một chương trình con khác. Chương trình con được chương trình chính sử dụng khi cần thiết bằng các lệnh gọi chương trình con; khi có lệnh gọi chương trình con, Vi điều khiển chuyển về thực hiện các đoạn chương trình của chương trình con, sau khi thực hiện chương trình con Vi điều khiển tiếp tục trở về thực hiện các câu lệnh trong chương trình chính.

Chương trình con giúp cho chương trình mạch lạc, dễ hiểu hơn, nếu trong chương trình chính có các đoạn chương trình được lặp đi lặp lại nhiều lần thì các đoạn chương trình đó thường được viết thành một chương trình con và truy xuất bằng một câu lệnh gọi chương trình con. Việc sử dụng chương trình con giúp cho việc tìm lỗi và chỉnh sửa chương trình dễ hơn, nếu chương trình chính sử dụng nhiều lần chương trình con, khi cần sửa đổi chỉ cần thay đổi các câu lệnh trong chương trình con.

Chương trình con bắt đầu bằng một *nhãn* và kết thúc bằng lệnh *Reti*, chương trình con có thể đặt ở đầu hoặc cuối chương trình.

2.5.1. Lệnh gọi chương trình con dùng địa chỉ tuyệt đối

- Cú pháp: ***ACall addr11***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte

- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Khi lệnh được thực hiện, Vi điều khiển chuyển về thực hiện các câu lệnh của chương trình con bắt đầu từ địa chỉ ***addr11*** trên Rom, địa chỉ ***addr11*** có thể thay bằng nhãn bắt đầu của một chương trình con. Câu lệnh được thực hiện khi địa chỉ addr11 cách lệnh gọi không quá 2 KByte .
- Ví dụ: *ACall 45A6H*

2.5.2. Lệnh gọi chương trình con dùng địa chỉ tuyệt đối

- Cú pháp: ***ACall addr16***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Khi lệnh được thực hiện, Vi điều khiển chuyển về thực hiện các câu lệnh của chương trình con bắt đầu từ địa chỉ ***addr16*** trên Rom, địa chỉ ***addr16*** có thay bằng nhãn bắt đầu chương trình con. Câu lệnh có thể gọi chương trình con ở bất kì vị trí nào trên Rom vì khoảng cách từ lệnh gọi đến chương trình con là 64 KByte.

2.5.3. Lệnh kết thúc chương trình con

- Cú pháp: ***Ret***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Lệnh này dùng kết thúc chương trình con, khi gặp lệnh này Vi điều khiển quay về thực hiện lệnh ở chương trình chính.

2.5.4. Lệnh kết thúc chương trình con phục vụ ngắt

- Cú pháp: ***Reti***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Lệnh này dùng kết thúc chương trình con ngắt, khi gặp lệnh này Vi điều khiển quay về thực hiện lệnh ở chương trình chính.

2.5.5. Lệnh nhảy ngắn đến địa chỉ tuyệt đối

- Cú pháp: ***AJMP addr11***

- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Khi lệnh được thực hiện, Vi điều khiển chuyển về thực hiện các câu lệnh của chương trình bắt đầu tại địa chỉ ***addr11*** trên Rom, địa chỉ ***addr11*** có thể thay bằng nhãn . Câu lệnh chỉ được thực hiện khi vị trí lưu chương trình cần thực hiện cách lệnh gọi không quá 2 KByte

2.5.6. Lệnh nhảy dài đến địa chỉ tuyệt đối

- Cú pháp: ***LJMP addr16***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Khi lệnh được thực hiện, Vi điều khiển chuyển về thực hiện các câu lệnh của chương trình bắt đầu tại địa chỉ ***addr11*** trên Rom, địa chỉ ***addr11*** có thể thay bằng nhãn . Câu lệnh có thể gọi chương trình ở bất kì vị trí nào trên Rom vì khoảng cách từ lệnh gọi đến chương trình con là 64 KByte

2.5.7. Lệnh nhảy tương đối

- Cú pháp: ***SJMP rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Khi lệnh được thực hiện, Vi điều khiển chuyển đến thực hiện các câu lệnh của chương trình được đánh dấu bằng nhãn. Câu lệnh chỉ được thực hiện địa chỉ của nhãn cách lệnh gọi không quá 128 Byte.(cả tới hoặc lùi)

2.5.8. Lệnh nhảy gián tiếp

- Cú pháp: ***JMP @A+DPTR***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Khi lệnh được thực hiện, Vi điều khiển chuyển đến thực hiện các câu lệnh của chương trình có địa chỉ trên ROM bằng giá trị của A cộng với giá trị lưu giữ trên DPTR

2.5.9. Lệnh nhảy thuận với cờ Zero

Cú pháp: ***JZ rel***

Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte

Thời gian thực hiện: 2 chu kì máy

Công dụng:

Nếu cờ Zero có giá trị 1(tức thanh ghi A có giá trị 0), Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt

Nếu cờ Zero có giá trị 0(tức thanh ghi A có giá trị khác 0), Vi điều khiển thực hiện lệnh kế tiếp (không thực hiện lệnh nhảy)

2.5.10. Lệnh nhảy nghịch với cờ Zero

- Cú pháp: ***JNZ rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

  Nếu cờ Zero có giá trị 0(tức thanh ghi A có giá trị khác 0), Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt

  Nếu cờ Zero có giá trị 1(tức thanh ghi A có giá trị 0), Vi điều khiển thực hiện lệnh kế tiếp (không thực hiện lệnh nhảy)

2.5.11. Lệnh nhảy thuận với cờ C

- Cú pháp: ***JC rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

  Nếu cờ C có giá trị 1, Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt

  Nếu cờ C có giá trị 0, Vi điều khiển thực hiện lệnh kế tiếp (không thực hiện lệnh nhảy)

2.5.12. Lệnh nhảy nghịch với cờ Zero

- Cú pháp: ***JNC rel***

- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

  Nếu cờ C có giá trị 0, Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt

  Nếu cờ C có giá trị 1, Vi điều khiển thực hiện lệnh kế tiếp (không thực hiện lệnh nhảy)

2.5.13. Lệnh nhảy thuận với giá trị của bit nhớ

- Cú pháp: ***JB bit,rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

  Nếu bit nhớ có giá trị 1, Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt

  Nếu bit nhớ có giá trị 0, Vi điều khiển thực hiện lệnh kế tiếp (không thực hiện lệnh nhảy)

2.5.14. Lệnh nhảy nghịch với giá trị của bit nhớ

- Cú pháp: ***JNC bit,rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

  Nếu bit nhớ có giá trị 0, Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt

  Nếu bit nhớ có giá trị 1, Vi điều khiển thực hiện lệnh kế tiếp (không thực hiện lệnh nhảy)

2.5.15. Lệnh nhảy thuận với giá trị của bit nhớ và xóa bit

- Cú pháp: ***JBC bit,rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy

- Công dụng:

Nếu bit nhớ có giá trị 1, Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt, đồng thời xóa giá trị chứa trong bit nhớ đó tức là đưa bit nhớ đó về giá trị 0

Nếu bit nhớ có giá trị 0, Vi điều khiển thực hiện lệnh kế tiếp (không thực hiện lệnh nhảy)

2.5.16. Lệnh nhảy có điều kiện(so sánh giá trị của thanh ghi A và Rn)

- Cú pháp: ***CJNE A,direct,rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

Vi điều khiển nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt nếu giá trị của thanh ghi A khác giá trị của ô nhớ có địa chỉ direct, nếu bằng nhau Vi điều khiển không nhảy và thực hiện lệnh kế

Ảnh hưởng của lệnh đến cờ nhớ C:

Nếu giá trị của thanh ghi A ≥ giá trị của ô nhớ có địa chỉ direct thì bit C có giá trị 0

Nếu giá trị của thanh ghi A < giá trị của ô nhớ có địa chỉ direct thì bit C có giá trị 1

2.5.17. Lệnh nhảy có điều kiện(so sánh giá trị của thanh ghi A và dữ liệu cho trước)

Cú pháp: ***CJNE A,#data,rel***

Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte

Thời gian thực hiện: 2 chu kì máy

Công dụng:

Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt, nếu giá trị của thanh ghi A khác giá trị dữ liệu cho trước, nếu bằng nhau Vi điều khiển không nhảy và thực hiện lệnh kế

Ảnh hưởng của lệnh đến cờ nhớ C:

Nếu giá trị của thanh ghi A ≥ giá trị dữ liệu cho trước thì bit C có giá trị 0

Nếu giá trị của thanh ghi A < giá trị dữ liệu cho trước thì bit C có giá trị 1

2.5.18. Lệnh nhảy có điều kiện(so sánh giá trị của thanh ghi Rn và dữ liệu cho trước)

- Cú pháp: ***CJNE Rn,#data,rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

  Vi điều khiển sẽ nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt nếu giá trị của thanh ghi Rn khác giá trị dữ liệu cho trước, nếu bằng nhau Vi điều khiển không nhảy và thực hiện lệnh kế.Ảnh hưởng của lệnh đến cờ nhớ C:

  Nếu giá trị của thanh ghi A ≥ giá trị dữ liệu cho trước thì bit C có giá trị 0

  Nếu giá trị của thanh ghi A < giá trị dữ liệu cho trước thì bit C có giá trị 1

2.5.18. Lệnh nhảy có điều kiện(so sánh giá trị của ô nhớ có địa chỉ gián tiếp và dữ liệu cho trước)

- Cú pháp: ***CJNE @Ri,#data,rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

  Vi điều khiển nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt nếu giá trị của ô nhớ có địa chỉ bằng giá trị của Ri khác giá trị dữ liệu cho trước, nếu bằng nhau Vi điều khiển không nhảy và thực hiện lệnh kế Ảnh hưởng của lệnh đến cờ nhớ C:
- Nếu giá trị của ô nhớ có địa chỉ gián tiếp ≥ giá trị dữ liệu cho trước thì bit C có giá trị 0
- Nếu giá trị của ô nhớ có địa chỉ gián tiếp < giá trị dữ liệu cho trước thì bit C có giá trị 1

2.5.19. Lệnh nhảy có điều kiện kết hợp với lệnh giảm trên thanh ghi Rn

Cú pháp: ***DJNZ Rn,rel***

- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy

- Công dụng:

  Giảm giá trị của thanh ghi Rn xuống 1 đơn vị, và

  -Nếu giá trị trong thanh ghi Rn khác 0, Vi điều khiển nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt.

  -Nếu giá trị trong thanh ghi Rn bằng 0, Vi điều khiển thực hiện lệnh kế tiếp

2.5.20. Lệnh nhảy có điều kiện kết hợp với lệnh giảm trên ô nhớ có địa chỉ direct

- Cú pháp: ***DJNZ direct,rel***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 3 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng:

  Giảm giá trị của ô nhớ có địa chỉ direct xuống 1 đơn vị

  Nếu giá trị trong ô nhớ có địa chỉ direct khác 0, Vi điều khiển nhảy đến thực hiện chương trình tại địa chỉ mà nhãn được đặt.Nếu giá trị trong ô nhớ có địa chỉ direct bằng 0, Vi điều khiển thực hiện lệnh kế tiếp

2.5.21. Lệnh delay 1 chu kì máy

- Cú pháp: ***NOP***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: delay trong 1 chu kì máy

2.6 Nhóm lệnh xử lí từng bit

Qui ước: trong câu lệnh "***bit***" đại diện cho một địa chỉ của bit nhớ

2.6.1. Lệnh xoá cờ nhớ C

- Cú pháp: ***CLR C***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Xóa cờ nhớ C - tức là đưa giá trị của cờ nhớ C về 0

2.6.2. Lệnh xoá bit

- Cú pháp: ***CLR bit***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Xóa giá trị của bit nhớ có địa chỉ xác định - tức là đưa giá trị bit đó về 0

2.6.3. Lệnh thiết đặt cờ nhớ C

- Cú pháp: ***SetB C***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: thiết đặt cờ nhớ C - tức là đưa giá trị của cờ nhớ C lên 1

2.6.4. Lệnh thiết đặt giá trị cho bit nhớ

- Cú pháp: ***SetB bit***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Thiết đặt giá trị bit nhớ có địa chỉ xác định - tức là đưa giá trị bit đó lên 1

2.6.5. Lệnh bù cờ nhớ C

- Cú pháp: ***CPL C***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 1 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: đổi giá trị của cờ nhớ C, nếu trước đó C có giá trị 0 chuyển thành 1, và ngược lại nếu trước đó C có giá trị 1 chuyển thành 0

2.6.6. Lệnh bù bit

- Cú pháp: ***CPL bit***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy

- Công dụng: đổi giá trị của bit có địa chỉ xác định, nếu trước đó bit đó có giá trị 0 chuyển thành 1, và ngược lại nếu trước đó bit đó có giá trị 1 chuyển thành 0

2.6.7. Lệnh And cờ nhớ C với bit

- Cú pháp: ***ANL C,bit***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Thực hiện phép And cờ nhớ C và bit có địa chỉ xác định, kết quả lưu ở C

2.6.8. Lệnh And cờ nhớ C với bit đã được lấy bù

- Cú pháp: ***ANL C,/bit***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Thực hiện phép And cờ nhớ C và bit có địa chỉ xác định đã được lấy bù, kết quả lưu ở C

2.6.9. Lệnh OR cờ nhớ C với bit

- Cú pháp: ***ORL C,bit***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Thực hiện phép And cờ nhớ C và bit có địa chỉ xác định, kết quả lưu ở C

2.6.10. Lệnh OR cờ nhớ C với bit đã được lấy bù

- Cú pháp: ***ORL C,/bit***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Thực hiện phép And cờ nhớ C và bit có địa chỉ xác định đã được lấy bù, kết quả lưu ở C

2.6.11. Lệnh chuyển giá trị bit có địa chỉ xác định vào cờ nhớ C

- Cú pháp: ***Mov C,bit***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 1 chu kì máy
- Công dụng: Thực hiện chuyển giá trị của bit có địa chỉ xác định vào cờ nhớ C

2.6.12. Lệnh chuyển giá trị cờ nhớ C vào bit có địa chỉ xác định

- Cú pháp: ***Mov bit,C***
- Lệnh này chiếm dung lượng bộ nhớ ROM là 2 Byte
- Thời gian thực hiện: 2 chu kì máy
- Công dụng: Thực hiện chuyển giá trị của cờ nhớ C vào bit có địa chỉ xác định.

TÀI LIỆU SƯU TẦM TẠI TRANG WEB:

WWW.CODIENTU.INFO

TÁC GIẢ: NGUYỄN DŨNG.

Email:*vdk@codientu.info*