THỰC HÀNH LẬP TRÌNH HỢP NGỮ TRÊN 8086

CẤU TRÚC RẼ NHÁNH, VÒNG LẶP

MỤC TIÊU

- Hiểu cách so sánh hai số trong hợp ngữ
- Hiểu cách thay đổi thứ tự thực hiện các lệnh
- Biết cách sử dụng các lệnh so sánh, nhảy và lặp

1. TÓM TẮT LÝ THUYẾT

Lệnh so sánh

Cú pháp: CMP Trái, Phải ; Cờ ← Trái – Phải

Nếu Trái > Phải ⇒ Trái - Phải > 0 : CF = 0 và ZF = 0
Nếu Trái < Phải ⇒ Trái - Phải < 0 : CF = 1 và ZF = 0
Nếu Trái = Phải ⇒ Trái - Phải = 0 : CF = 0 và ZF = 1
Trái, Phải: Immed, Reg, Mem
Bản chất của lệnh CMP là lệnh SUB Đích, Nguồn (thực hiện phép tính Đích – Nguồn) nhưng kết quả của phép tính không được lưu vào Đích như trong lệnh SUB mà tính chất của kết quả được thể hiện thông qua cờ

Ví dụ: so sánh hai số nguyên dương
MOV AH, 1 ; AH ← 1
MOV AL, 2 ; AL ← 2
CMP AH, AL ; CF ← 1, ZF ← 0 vì AH < AL
Sau khi thực hiện các lệnh trên, cờ Carry bật (CF=1), báo hiệu rằng AH < AL

1. TÓM TẮT LÝ THUYẾT

Lệnh so sánh nhị phân

Cú pháp: TEST Trái, Phải ; Cờ ← Trái and Phải

Nếu Trái and Phải = 0 thì ZF = 1, ngược lại thì ZF = 0

Bản chất của lệnh TEST là lệnh AND Đích, Nguồn nhưng kết quả của phép tính không được lưu vào Đích như trong lệnh AND mà ảnh hưởng lên cờ.

Ví dụ: kiểm tra hai bit cuối cùng của AL

TEST AL, 3 ; 3h = 11b

Nếu cờ Zero bật (ZF=1), có nghĩa là cả hai bit 0 và 1 của AL đều bằng 0.

1. TÓM TẮT LÝ THUYẾT

Lệnh nhảy không điều kiện

Cú pháp: JMP <target> ; Nhảy đến địa chỉ <Target>

Có các trường hợp sau:

• JMP SHORT <nhãn> ; (short jump). Kiểu này chỉ nhảy trong phạm vi từ –128 đến +127 byte so với vị trí hiện tại.

Ví dụ: JMP SHORT Calculate

• JMP <nhãn> ; (near jump). Kiểu này nhảy tùy ý trong phạm vi segment.

Ví dụ: JMP Calculate

• JMP FAR PTR <nhãn> ; (far jump). Kiểu này nhảy đến bất kì chỗ nào.

Ví dụ: JMP FAR PTR Calculate

• JMP <con trỏ 2 byte> ; (near indirect jump). Khi thực hiện, thanh ghi PC sẽ được gán bằng giá trị lưu tại địa chỉ này. Có thể kết hợp dùng với định vị chỉ số.

• JMP <con trỏ 4 byte> ; (far indirect jump). Tương tự trường hợp trên, nhưng con trỏ gồm cả segment và offset. Chỉ khác ở khai báo con trỏ

• JMP <thanh ghi 2 byte> ; (indirect jump via regs). Nhảy đến địa chỉ lưu trong thanh ghi AX.

1. TÓM TẮT LÝ THUYẾT

Lệnh nhảy có điều kiện

Cú pháp: J<điềukiện> <Label>

Các lệnh nhảy có điều kiện bắt đầu bằng chữ J sau đó là các chữ cái biểu thị điều kiện (ví dụ JGE: Jump if Greater than or Equal, nhảy nếu lớn hơn hay bằng), tiếp sau là một tên nhãn.

Điều kiện để lệnh nhảy xem xét khi thi hành là giá trị các cờ được tạo ra từ lệnh CMP hay TEST. Khi sử dụng lệnh nhảy có điều kiện sau khi thực hiện phép so sánh,phải đặc biệt lưu ý toán hạng trong phép so sánh là số có dấu (signed) hay không có dấu (unsigned) để lựa chọn lệnh cho phù hợp.

1. TÓM TẮT LÝ THUYẾT

Lệnh nhảy có điều kiện

Một số lệnh nhảy có điều kiện thường dùng:

Lệnh	Ý Nghĩa	Điều Kiện
JB JNAE	Nhảy nếu nhỏ hơn (Jump if Below) Nhảy nếu không lớn hơn hoặc bằng	CF = 1
JAE JNB	Nhảy nếu lớn hơn hoặc bằng (Jump if Above or Equal) Nhảy nếu không nhỏ hơn	CF = 0
JBE JNA	Nhảy nếu nhỏ hơn hoặc bằng (Jump if Below or Equal) Nhảy nếu không lớn hơn	CF = 1 và ZF = 1
JA JNBE	Nhảy nếu lớn hơn (Jump if Above) Nhảy nếu không nhỏ hơn hoặc bằng	CF = 0 và ZF = 0
JE JZ	Nhảy nếu bằng (Jump if Equal) Nhảy nếu bằng (Jump if Zero)	ZF = 1
JNE JNZ	Nhảy nếu không bằng (Jump if Not Equal) Nhảy nếu không bằng (Jump if Not Zero)	ZF = 0

1. TÓM TẮT LÝ THUYẾT

Lệnh lặp

Bằng cách dùng các lệnh nhảy có thể tạo ra vòng lặp. Tuy nhiên, để viết chương trình tiện lợi và ngắn gọn, có thể dùng thêm các lệnh lặp như LOOP, LOOPZ,…

Cú pháp: LOOP <Label>

tự động giảm CX một đơn vị, sau đó kiểm tra xem CX có bằng 0, nếu không bằng thì nhảy đến nhãn <Label>

Cú pháp: LOOPZ <Label>

tự động giảm CX một đơn vị, sau đó kiểm tra xem CX có bằng 0 hoặc cờ ZF có bật không (ZF=1), nếu cả hai điều này không xảy ra thì nhảy đến nhãn <Label>

1. TÓM TẮT LÝ THUYẾT

Lệnh lặp

Ví dụ: Nhập mảng A gồm 10 ký tự

```
MOV SI, 0 ; chỉ số mảng
MOV CX, 10 ; số lần lặp
LAP: MOV AH, 1 ; nhập ký tự
INT 21H
MOV A[SI], AL
INC SI
```

2. THỰC HÀNH

2.1. Cấu trúc rẽ nhánh

Chương trình sau đây nhận 1 ký tự. Nếu là ký tự HOA thì in ra màn hình "Ky tu HOA". Ngược lại in ra câu "Ky tu thuong". (Mã ASCII của ký tự HOA <= 'Z').

Soạn thảo và lưu với tên BAI_3A.ASM.

- Dịch và chạy CT ở những trường hợp khác nhau để xem kết quả trên màn hình.
- Vẽ lưu đồ điều khiển của chương trình.

2. THỰC HÀNH

2.1. Cấu trúc rẽ nhánh

- Tại sao cần phải có lệnh **JMP EXIT**? Nếu không có lệnh ấy thì chương trình thực hiện như thế nào? Chạy chương trình để kiểm chứng.
- Thay lệnh **JA NHAN** bằng lệnh **JNA NHAN.** Sửa chương trình sao cho kết quả không thay đổi.
- Khi ký tự nhập vào không phải là chữ cái thì kết quả in ra màn hình là gì? Tại sao?

2. THỰC HÀNH

2.1. Cấu trúc rẽ nhánh

```
DSEG SEGMENT
tbao1 DB "Ky tu HOA.$"
tbao2 DB "Ky tu thuong.$"
DSEG ENDS
CSEG SEGMENT
ASSUME CS: CSEG, DS: DSEG
start:mov ax, DSEG
mov ds, ax
mov ah, 01h
int 21h
jmp exit
```

2. THỰC HÀNH

2.1. Cấu trúc rẽ nhánh

```
jmp exit
nhan: mov ah, 09 ; in "Ky tu thuong"
lea dx, tbao2
int 21h
exit:mov ah, 7
int 21h
mov ah, 4Ch ; trở về hệ điều hành
int 21h
CSEG ENDS
END start
```

2. THỰC HÀNH

2.2. Cấu trúc vòng lặp

- Xem chương trình in ra màn hình lần lượt các ký tự từ A đến Z được viết dưới. Hãy soạn thảo và đặt tên tập tin là BAI_3B.ASM.
- Dịch và chạy chương trình để xem kết quả trên màn hình.
- Vòng lặp trong chương trình bao gồm đoạn lệnh nào? Viết theo kiểu while do hay repeat ... until hay for? Vẽ lưu đồ chương trình.

2. THỰC HÀNH

2.2. Cấu trúc vòng lặp

```
CSEG SEGMENT
ASSUME CS: CSEG
start:mov dl, 'A' ; DL chứa ký tự đầu tiên 'A'
nhan:mov ah, 02h ; in ký tự trong DL ra màn hình
int 21h
inc dl ; DL chứa ký tự kế cần in
cmp dl, 'Z' ; So sánh DL với 'Z'
jna nhan ; Nếu <= 'Z' thì tiếp tục in
mov ah, 08h ; Nếu > 'Z' thì thoát (không in tiếp)
int 21h
mov ah, 4Ch
int 21h
CSEG ENDS
END start
```

2. THỰC HÀNH

2.2. Cấu trúc vòng lặp

- Sửa chương trình để in ra màn hình lần lượt các ký tự từ 'Z' đến 'A'.
- Tiếp tục sửa chương trình sao cho giữa các ký tự có 1 khoảng trống (Z Y ....B A)
- Dùng lệnh LOOP để viết lại chương trình BAI_3B.ASM theo cấu trúc vòng lặp **for**

2. THỰC HÀNH

2.3. Bài tập

1 Viết chương trình cho nhập 1 ký tự từ màn hình và xuất câu thông báo tương ứng
sau:
- Nếu ký tự nhập là ‘S’ hay ‘s’ thì in ra “Good morning!”
- Nếu ký tự nhập là ‘T’ hay ‘t’ thì in ra “Good Afternoon!”
- Nếu ký tự nhập là ‘C’ hay ‘c’ thì in ra “Good everning!”

2 Viết lại chương trình BAI_3A.ASM sao cho chương trình có thể phân biệt được 3 loại ký tự nhập từ bàn phím: "Ký tự HOA", "ký tự thường" và "ký tự khác".

2. THỰC HÀNH

2.3. Bài tập

3 Viết chương trình nhập từ bàn phím 1 ký tự thường. Sau đó in ra màn hình lần lượt các ký tự từ ký tự nhận được đến 'z' sao cho giữa các ký tự có 1 khoảng trống.

4 Không dùng hàm 0Ah/21h, hãy dùng lệnh lặp để viết chương trình nhập vào 1 chuỗi ký tự. Sau khi nhập xong đếm xem chuỗi có bao nhiêu ký tự. In ra màn hình chuỗi nhận được và số ký tự có trong chuỗi.

Ví dụ: S = "Hello world !" ==> Số kí tự trong chuỗi là 13.

2. THỰC HÀNH

2.3. Bài tập

5 Viết chương trình cho phép nhập vào một chuỗi bất kỳ. Sau đó:
- Đổi tất cả ký tự thường thành ký tự hoa và in ra màn hình.
- Đổi tất cả ký tự hoa thành ký tự thường và in ra màn hình.

Ví dụ: S = ‘weLcOme To AssEmblY’

In ra: welcome to assembly - WELCOME TO ASSEMBLY

6 Nhập vào 2 chuỗi số, đổi 2 chuỗi thành số, sau đó cộng hai số, đổi ra chuỗi và xuất chuỗi tổng.

Ví dụ: S1 = "123" => N1 = 123

S2 = "456" => N2 = 456

N = N1 + N2 = 123 + 456 = 579 => S = "579" (xuất S ra màn hình)

2. THỰC HÀNH

2.3. Bài tập

7 Nhập 2 số nguyên dương A, B. Tính A/B, A*B (không dùng lệnh DIV, MUL) và in ra màn hình kết quả.

Ví dụ: A=18, B=3

Tính A/B: 18 - 3 - 3 - 3 - 3 - 3 - 3 = 0, vậy A/B = 6 (tổng trừ B cho đến khi A = 0).

Tính A*B = 18 + 18 + 18 = 54