

Chương 8 : Cấu trúc dữ liệu khi làm việc với Vòng lặp

Mục tiêu

- Biết cách mô phỏng cấu trúc dữ liệu khi làm việc với vòng lặp nhờ ưu tiên ngõ lặp trình cấp cao.
- Nắm vững các lệnh nhúng trong lập trình Assembly.
- Trên cơ sở đó, vận dụng để lập trình giải quyết 1 số bài toán.

Noäi dung

- ✓ Söi caàn thieát cuöa leänh nhaü trong laäp trình ASM.
- ✓ Leänh JMP (Jump) : nhaü khoäng ñieàu kieän.
- ✓ Leänh LOOP : cho pheùp laäp 1 công vieäc vöüi 1 soá laàn naøo ñoù.
- ✓ Caùc leänh so saùnh vaø luaän lyù.
- ✓ Leänh laäp coù ñieàu kieän.
- ✓ Leänh nhaü coù ñieàu kieän.
- ✓ Bieäu dieän mô phöng cáú truùc luaän lyù möüc cao.
- ✓ Chöông trình con.
- ✓ Moät soá chöông trình minh hoä.

Sõi caàn thieát cuõa leänh

nhaûy

- Ôû cauc chõng trìn viéat baèng ngoân ngõõ caáp cao thì viéc nhaûy (leänh GoTo) laø ñieàu neân traùnh nhõng ôû laäp trìn heä thoáng thì ñây laø viéc caàn thieát vaø laø ñieäm maïn cuõa 1 chõng trìn viéat baèng Assembly.
- Moät leänh nhaûy → CPU phaûi thõic thì 1 ñoain leänh ôû 1 choã khac vùi nôï maø cauc leänh ñang
- **ñõõic thõic thì**
Trong laäp trìn, cõ nhõng nhõm phaùt bieäu caàn phaûi laäp ñi laäp laïi nhieàu laàn trong 1 ñieàu kieän naø ñõ. Ñeä ñaùp õng ñieàu kieän naøy ASM cung caáp 2 leänh JMP vaø LOOP.

Leänh JMP (Jump)

- **Coäng ðuäng** : Chyeän ñieän khiän khoäng ñieän kieän
• **Cuè phaùp** : JMP **ñích**
- **Nhaüy gaän (NEAR)** : 1 taùc vui nhaüy trong cuøng 1 segment.
- **Nhaüy xa (FAR)** : 1 taùc vui nhaüy sang segment khaùc.

Cacù leänh chuyeån ñieàu khieån

Chuyeån ñieàu khieån voâ ñieàu kieån

JMP [SORT | NEAR PTR | FAR PTR] DEST

Chuyeån ñieàu khieån coù ñieàu kieån

JConditional destination

Ex : JNZ nhaõn ñích ;

LEÄNH LOOP

Coâng ðuïng : cho pheùp laëp 1 coâng vieäc vôùi 1 soá laàn naøo ñoù.
Moãi laàn laëp CX giaûm ñi 1 ñôn vò. Voøng laëp chaám döùt khi CX =0.

Ex 1 : xuaát ra maøn hình 12 döøng goàm caùc kyù töï A.

```
MOV CX, 12 * 80
```

```
MOV DL, 'A'
```

NEXT :

```
MOV AH, 2
```

```
INT 21H
```

```
LOOP NEXT
```

LOOP (tt)

Ex : còu 1 Array A goàm 6 bytes, cheùp A sang array B
- duøng SI vaø DI ñeã laáy Offset

```
MOV SI, OFFSET A  
MOV DI, OFFSET B  
MOV CX, 6  
MOVE_BYTE :  
    MOV AL, [SI]  
    MOV [DI], AL  
    INC SI  
    INC DI  
LOOP MOVE_BYTE  
A DB 10H,20H,30H,40H,50H,60H  
B DB 6 DUP (?)
```

CAÙC LEÄNH LUAÄN LYÙ

Löu yù veà caùc toaùn töü LOGIC :

AND 2 Bit : keát quaü laø 1 khi vaø chæ khi 2 bit laø 1

OR 2 Bit : keát quaü laø 1 khi 2 Bit coü bit laø 1

XOR 2 Bit : keát quaü laø 1 chæ khi 2 bit khaüc nhau

NOT 1 Bit : laáy ñaü cuüa Bit naøy

Löu yù veà thanh ghi côø :

Côø ZERO ñöôic laäp khi taüc vui cho keát quaü laø 0.

Côø CARRY ñöôic laäp khi coäng keát quaü bò traøn hay tröø phaüi mööin.

Côø SIGN ñöôic laäp khi bit daáu cuüa keát quaü laø 1, töüc keát quaü laø soá âm.

Leänh AND

Cuù phaùp : **AND** Destination , Source

Coäng düng :

Leänh này thöic hieän pheùp AND giöõa 2 toaùn haïng, keát quaû cuoái cuøng chöùa trong toaùn haïng ních.

Duøng ñeä xöùa caùc bit nhaát ñönh cuûa toaùn haïng ních giöõ nguyêän caùc bit coøn laii.

Muoán vaäy ta duøng 1 maãu bit goüi laø maët naï bit (MASK), caùc bit maët naï ñöôic choïn ñeä sao cho caùc bit töông öùng cuûa ních ñöôic thay ñoãi nhö mong muoán.

Leänh AND

Ex1 : xoaù bit daáu của AL, giöõ nguyêñ caùc bit cöøn laii :
duøng AND vöüi **01111111b** laøm maët naä

```
AND AL, 7FH
```

Ex2 :

```
MOV AL, '5' ; Nöai maõ ASCII của số
```

```
AND AL, 0FH ; thaønh số tổng öùng.
```

Ex3 :

```
MOV DL, 'a' ; Nöai chöõ thöøng thaønh chöõ hoa.
```

```
AND DL, 0DFH ; thaønh số tổng öùng.
```



LEÄNH OR

Công dụng : dùng ñeå baät lên 1 số bit vàø giõõ nguyên cùc bit khaùc.

Cùc phaùp : `OR destination, source`

Ex1 :

`OR AL , 10000001b` ; baät bit cao nhất vàø bit thấp nhất trong thanh ghi AL lên 1

Ex 2:

`MOV AL , 5` ; ñoải 0..9 thaønh kyù số

`OR AL , 30h` ; ASCII töông öùng.

Ex 3:

`OR AL , AL` ; kiểm tra möt thanh ghi cù = 0.

Neáu : cø ZF ñöïc laäp \rightarrow `AL = 0`

cø SIGN ñöïc laäp \rightarrow `AL < 0`

cø ZR vàø cø SIGN không ñöïc laäp \rightarrow `AL > 0`

LEÄNH XOR

Công dụng : dùng để tạo ra mã toán học cao.

Cú pháp : `XOR destination, source`

Ex : lật bit cao của AL 2 lần

```
MOV AL , 00111011b ;
```

```
XOR AL, 11111111b ; AL = 11000100b
```

```
XOR AL, 11111111b ; AL = 00111011b
```

LEÄNH TEST

Cuù phaùp : TEST destination, source

Coâng duïng : duøng ñeã khaùu saùt trò cuûa töøng bit hay nhuòm bit.

Test thöïc hieän gioáng leänh AND nhöng khoâng laøm thay ñoái toaøn haïng ñích.

Ex : kieåm tra bit 13 trong DX laø 0 hay 1

TEST DX, 2000h

JZ BitIs0

BitIs1 : bit 13 is 1

BitIs0 : bit 13 is 0

Ñeã kieåm tra 1 bit naøo ñoù chæ caàn ñaët bit 1 vaøo ñuùng vò trí bit caàn kieåm tra vaø khaùu saùt côø ZF. (neáu bit kieåm laø 1 thì ZF seõ xoaù, ngöôïc laïi ZF ñöôïc laäp.

MINH HOÏA LEÄNH TEST

Ex : kieåm tra traing thaui maùy in. Interrupt 17H trong BIOS seõ kieåm tra traing thaui maùy in, sau khi kieåm tra AL seõ chõua traing thaui maùy in. Khi bit 5 cuõa AL laø 1 thì maùy in heát giaáy.

```
MOV AH, 2
```

```
INT 17h
```

```
TEST AL, 00100000b ; Test bit 5, neáu bit 5 = 1 → maùy in heát giaáy.
```

Leãnh TEST cho pheùp test nhieàu bit 1 löõt.

MINH HOÏA LEÄNH TEST(tt)

Ex :vieát ñoain leänh thöic hieän leänh nhaüy ñeán nhaõn A1
neáu AL chöua soá chaün.

TEST AL, 1 ; AL chöua soá chaún ?

JZ A1 ; neáu ñuùng nhaüy ñeán A1.

Leänh CMP

Cuù phaùp : CMP destination , source

Coäng düng : so saùnh toaùn haïng ñích vòu
toaùn haïng nguòan baèng caùch laáy toaùn
haïng ñích - toaùn haïng nguòan.

Hoait ñoäng : duøng pheùp trøø nhöng khoâng
coù toaùn haïng ñích naøo bò thay ñoái.

Caùc toaùn haïng cuûa leänh CMP khoâng theå cuøng laø
caùc oâ nhòu.

leänh CMP gioáng heät leänh SUB trøø vieäc toaùn haïng ñích
khoâng thay ñoái.

LEÄNH NHAÛY COÙ ÑIEÀU KIEÄN

Cuù phaùp : Jconditional destination

Coâng ñuïng : nhôø caùc leänh nhaÛy coù ñieàu kieän, ta môùi môô phoûng ñöôïc caùc phaùt bieäu coù caáu truùc cuûa ngoân ngöõ caáp cao baèng Assembly.

Phaïm vi

- Chæ nhaÛy ñeán nhaõn coù khoaûng caùch töø -128 ñeán +127 byte so vôùi vò trí hieän haønh.
- Duøng caùc traïng thaùi côø ñeå quyéat ñònh coù nhaÛy hay khoaûng?

LEÄNH NHAÛY COÙ ÑIEÀU KIEÄN

Hoait ñoäng

- ñeå thöïc hieän 1 leänh nhaûy CPU nhìn vaøo caùc thanh ghi côø.
- neáu ñieàu kieän cuûa leänh nhaûy thoûa, CPU seõ ñieàu chænh IP troû ñeán nhaõn ñích caùc leänh sau nhaõn naøy seõ ñöôïc thöïc hieän.

.....

MOV AH, 2

MOV CX, 26

MOV DL, 41H

PRINT_LOOP :

INT 21H

INC DL

DEC CX

JNZ PRINT_LOOP

MOV AX, 4C00H

INT 21H

LEÄNH NHAÛY DÖIA TREÄN KEÁT QUAÛ SO SAÛNH CAÛC TOAÛN HAÏNG KHOÄNG DAÁU.

Thööøng duøng leänh CMP Opt1 , Opt2 ñeä xeùt ñieäu kieän nhaÛy hoaëc döia treän caùc côø.

JZ	NhaÛy neáu keát quaÛ so saÛnh = 0
JE	NhaÛy neáu 2 toaÛn haïng baèng nhau
JNZ	NhaÛy neáu keát quaÛ so saÛnh laø khaùc nhau.
JNE	NhaÛy neáu 2 toaÛn haïng khaùc nhau.
JA	NhaÛy neáu $Opt1 > Opt2$
JNBF	NhaÛy neáu $Opt1 \leq Opt2$

LEÄNH NHAÛY DÖIA TREÄN KEÁT QUAÛ SO SAÛNH
CAÛC TOAÛN HAÏNG KHOÄNG DAÁU (ctn) .

JNC	NhaÛy neáu khoâng coù Carry.
JB	NhaÛy neáu $Opt1 < Opt2$
JNAE	NhaÛy neáu $Not(Opt1 \geq Opt2)$
JC	NhaÛy neáu coù Carry
JBE	NhaÛy neáu $Opt1 \leq Opt2$
JNA	NhaÛy neáu $Not (Opt1 > Opt2)$

LEÄNH NHAÛY DÖIA TREÄN KEÄT QUAÛ SO SAÛNH
CAÛC TOAÛN HAÛNG COÛ DAÄU .

JG	NhaÛy neáu $Opt1 > Opt2$
JNLE	NhaÛy neáu $Not(Opt1 \leq Opt2)$
JGE	NhaÛy neáu $Opt1 \geq Opt2$
JNL	NhaÛy neáu $Not(Opt1 < Opt2)$
JL	NhaÛy neáu $Opt1 < Opt2$
JNGE	NhaÛy neáu $Not(Opt1 \geq Opt2)$
JLE	NhaÛy neáu $Opt1 \leq Opt2$
JNG	NhaÛy neáu $Not(Opt1 > Opt2)$

LEÄNH NHAÛY DÖIA TREÄN CAUC CÖÖ .

JCXZ	NhaÛy neáu $CX=0$
JS	NhaÛy neáu $SF=1$
JNS	NhaÛy neáu $SF = 0$
JO	NhaÛy neáu ñaõ traøn trò
JL	NhaÛy neáu $Opt1 < Opt2$
JNGE	NhaÛy neáu $Not (Opt1 \geq Opt2)$
JLE	NhaÛy neáu $Opt1 \leq Opt2$
JNO	NhaÛy neáu traøn trò
JP	NhaÛy neáu parity chaún
JNP	NhaÛy neáu $PF = 0$

CAÙC VÒ DUÏ MINH HOÏA LEÄNH NHAÛY COÙ ÑÏK

Ex1 : tìm số lớn hơn trong
2 số ch÷a trong thanh ghi
AX và BX . Kết quả ñể
trong DX

```
MOV DX, AX           ; giá trị AX là số lớn hơn.  
CMP DX, BX          ; IF AX >=BX then  
JAE QUIT             ; nếu ñếán QUIT  
MOV DX, BX           ; ngược lại chép BX vào DX  
QUIT :  
    MOV AH,4CH  
    INT 21H  
    .....
```


CAÙC VÍ DUÏ MINH HOÏA LEÄNH NHAÛY COÙ ÑK

Ex1 : tìm số nhỏ nhất trong 3 số ch÷a trong thanh ghi AL, BL và CL . Kết quả ñể trong biến SMALL

```
MOV SMALL, AL
CMP SMALL, BL
JBE L1
MOV SMALL, BL
L1 :
  CMP SMALL, CL
  JBE L2
MOV SMALL, CL
L2 : ...
```

; giá trị AL nhỏ nhất
; nếu SMALL <= BL thì
Nhập ñến L1
; nếu SMALL <= CL thì
; Nhập ñến L2
; CL là số nhỏ nhất

Caùc leänh dòch vaø quay bit

SHL (Shift Left) : dòch caùc bit cuûa toaøn haïng ních sang traùi

Cuù phaùp : SHL toaøn haïng ních ,1

Dòch 1 vò trí.

Cuù phaùp : SHL toaøn haïng ních ,CL

Dòch n vò trí trong ñoù CL chöùa soá bit caàn dòch.

Hoait ñoäng : moät giaù trò 0 seõ ñöôïc ñöa vaøo vò trí beân phaûi nhaát cuûa toaøn haïng ních, coøn bit msb cuûa ñoù ñöôïc ñöa vaøo côø CF

Cài lệnh dòch vàø quay bit

Ex : DH chöua 8Ah, CL chöua 3.

SHL DH, CL ; 01010000b

? Cho bieát keát quaû cuûa :

SHL 1111b, 3

**MT thöïc hieän pheùp nhaân baèng
dòch traùi**

leänh dòch phaûi SHR

Coäng düng : dòch caùc bit cuûa toaøn haäng ñích sang beân phaûi.

Cuù phaùp : **SHR toaøn haäng ñích , 1**

SHR toaøn haäng ñích , CL ; dòch phaûi n bit trong ñoù CL chõua n

Hoait ñoäng : 1 giaù trò 0 seõ ñöôic ñõa vaøo bit msb cuûa toaøn haäng ñích, coøn bit beân phaûi nhaát seõ ñöôic ñõa vaøo côø CF.

MT thõic hieän pheùp chia baèng
dòch phaûi

leänh dòch phaui SHR

Ex : shr 0100b, 1 ; 0010b = 2

Ñoái vôi cauc soá leu, dòch phaui seõ chia ñoái
noù vaø laøm troøn xuoáng soá nguyêân gaàn
nhát

Ex : shr 0101b, 1 ; 0010b = 2

Chöông trìn con

Còu vai troø gioáng nhö chöông trìn
con ôu ngoân ngöõ caáp cao.

ASM còu 2 daing chöông trìn con : daing FAR vaø
daing NEAR.

Leänh goii CTC
naèm cương
ñoaìn boä
nhòu vòuì CTC
ñöôic goii

Leänh goii CTC
naèm khaùc
ñoaìn boä
nhòu vòuì CTC
ñöôic goii

BIEAU DIEÃN CAÁU TRUÙC LOGIC MÖÙC CAO

Dù Assembly không có phầt biểu IF, ELSE, WHILE, REPEAT, UNTIL, FOR, CASE nhöng ta vẫn có thể tả hõp cầc leãnh của Assembly ñể hiệñ thõic cầu trùc logic của ngôn ngữ cấp cao.

Caáu truùc IF Ñôn giaân

Phaùt bieáu IF seõ kieám tra 1
ñieàu kieän vaø theo sau ñoù laø 1
soá caùc phaùt bieáu ñöôïc thöïc
thi khi ñieàu kieän kieám tra coù
giaù trò truo.

Caáu truùc logic

```
IF (OP1=OP2)  
  <STATEMENT1>  
  <STATEMENT2>  
ENDIF
```

HIEÄN THÖÏC BAÈNG ASM

```
CMP OP1,OP2  
JNE CONTINUE  
  <STATEMENT1>  
  <STATEMENT2>  
CONTINUE : .....
```

Caáu truùc IF vôùi OR

Phaùt bieäu IF coù keøm toaøn töû OR

Caáu truùc logic

```
IF (A1>OP1) OR  
(A1>=OP2) OR  
(A1=OP3) OR  
(A1<OP4)  
  <STATEMENT>  
ENDIF
```

HIEÄN THÖÏC BAÈNG ASM

```
CMP A1,OP1  
JG EXECUTE  
CMP A1,OP2  
JGE EXECUTE  
CMP A1,OP3  
JE EXECUTE  
CMP A1,OP4  
JL EXECUTE  
JMP CONTINUE  
EXECUTE : <STATEMENT>  
CONTINUE : .....
```

Caáu truùc IF vôùi AND

Phaùt bieão IF coù keøm toaùn töü AND

Caáu truùc logic

```
IF (A1>OP1) AND  
(A1>=OP2) AND  
(A1=OP3) AND  
(A1<OP4)  
<STATEMENT>  
ENDIF
```

HIEÄN THÖÏC BAÈNG ASM

```
CMP A1,OP1  
JNG CONTINUE  
CMP A1,OP2  
JL CONTINUE  
CMP A1,OP3  
JNE CONTINUE  
CMP A1,OP4  
JNL CONTINUE  
<STATEMENT>  
JMP CONTINUE  
CONTINUE : .....
```

CHUÙ YÙ : khi ñieàu kieän coù toaùn töü AND,
caùch hay nhaát laø duøng nhaùy vôùi ñieàu
kieän ngöôïc laii ñeán nhaõn, boù qua phaùt
bieão trong caáu truùc Logic.

VOØNG LAÏP WHILE

Caáu truùc WHILE

Caáu truùc logic

```
DO WHILE (OP1<OP2)  
<STATEMENT1>  
<STATEMENT2>  
ENDDO
```

HIEÄN THÖÏC BAÈNG ASM

```
DO_WHILE :  
  CMP OP1, OP2  
  JNL ENDDO  
  <STATEMENT1>  
  <STATEMENT2>  
  JMP DO_WHILE  
ENDDO : .....
```

VOØNG LAÏP WHILE COÙ LOÀNG IF

Caáu truùc WHILE
coù loàng IF

Caáu truùc logic

```
DO WHILE (OP1<OP2)
<STATEMENT>
IF (OP2=OP3) THEN
<STATEMENT2>
<STATEMENT3>
ENDIF
ENDDO
```

HIEÄN THÖIC BAÈNG ASM

```
WHILE :
CMP OP1, OP2
JNL WHILE_EXIT
<STATEMENT1>
CMP OP2, OP3 ; phaàn If
JNE ELSE ; khoâng thoûa If
<STATEMENT2> ; thoûa If
<STATEMENT3>
JMP ENDIF; thoûa If neän
boû qua Else
ELSE : <STATEMENT4>
ENDIF : JMP WHILE
WHILE_EXIT : .....
```

Cấu trúc REPEAT UNTIL

VOØNG LAÏP REPEAT UNTIL

Cấu trúc logic

REPEAT

<STATEMENT1>

<STATEMENT2>

<STATEMENT3>

UNTIL (OP1=OP2) OR
(OP1>OP3)

Baøng
nhau
thoàit
Repeat

HIEÄN THÖIC BAÈNG ASM

REPEAT :

<STATEMENT1>

<STATEMENT2>

<STATEMENT3>

TESTOP12:

CMP OP1, OP2

JE ENDREPEAT

TESTOP13 :

CMP OP1, OP3

JNG REPEAT

ENDREPEAT :

Caáu truùc CASE

Caáu truùc logic
CASE INPUT OF

'A' : Proc_A

'B' : Proc_B

'C' : Proc_C

'D' : Proc_D

End ;

HIEÄN THÖIC BAÈNG ASM

```
CASE : MOV AL, INPUT
```

```
CMP AL, 'A'
```

```
JNE TESTB
```

```
CALL PROC_A
```

```
JMP ENDCASE
```

```
TESTB :
```

```
    CMP AL, 'B'
```

```
    JNE TESTC
```

```
    CALL PROC_B
```

```
    JMP ENDCASE
```

```
TESTC :
```

```
    CMP AL, 'C'
```

```
    JNE TESTD
```

```
    CALL PROC_C
```

```
    JMP ENDCASE
```

```
TESTD : CMP AL, 'D'
```

```
    JNE ENDCASE
```

```
    CALL PROC_D
```

```
ENDCASE : .....
```

LookUp Table

Rất hiệu quả khi xử lý phát biểu CASE
lạc dưng bằng OFFSET chừa nòa chæ của
nhấn hoặç của høm sẽ nhây ñén tuyø
vào ñieàu kieän.
Baúng Offset này ñôi gọi Lookup Table
rất hiệu quả khi dựng phát biểu Case
cò nhiều trò löia chöin.

LookUp Table

```

Case_table db 'A'      ; giá trị tìm kiếm
Dw Proc_A             ; Nào chæ caùc procedure
                       giáù sốu ôu ñò chæ
Db 'B'                0120
Dw Proc_B
Db 'C'                giáù sốu ôu ñò chæ
Dw Proc_C             0130
Db 'D'                giáù sốu ôu ñò chæ
Dw Proc_D             0140
    
```

'A'	0120	'B'	0130	'C'	0140	'D'	0150
	↑		0150				

Cấu trúc lưu
 trữ của
 CaseTable như
 sau

LookUp Table

Case :

MOV AL, INPUT

MOV BX, OFFSET CASE_TABLE

MOV CX, 4 ; lặp 4 lần số entry của table

TEST :

CMP AL, [BX] ; kiểm tra Input

JNE TESTAGAIN ; không thỏa kiểm tra tiếp

CALL WORD PTR [BX+1] ; gọi thủ tục tương ứng

JMP ENDCASE

TESTAGAIN : ADD BX , 3 ; sang entry sau của CaseTable

LOOP TEST

ENDCASE :

Chương trình con

Cấu trúc CTC :

```
TeânCTC PROC <Type>  
; câu lệnh  
RET  
TeânCTC ENDP
```

CTC có thể gọi 1 CTC khác hoặc gọi chính nó.

CTC nội gọi bằng lệnh **CALL** <TenCTC>.

CTC gần (near) gọi chương trình con nằm chung
segment với nó gọi nó.

CTC xa (far) gọi chương trình con không nằm
chung segment với nó gọi nó.

Kỹ thuật lặp trình

■ Hãy tưởng tượng chương trình → các chương trình con → nên giao hoán cấu trúc lựa chọn ly của CT làm cho CT dễ nhớ, dễ hiểu, dễ kiểm tra sai sót..

■ Nếu CTC hãy viết thành ghi vào Stack bằng lệnh PUSH nếu lỗi trình bày hiện hành.

■ Sau khi hoàn tất công việc của CTC nên phục hồi lại trình các thành ghi lúc trước để Push bằng lệnh POP .

■ Trình tự lặp ngược nhau nếu trình thành ghi nào trước cho thành ghi này.

■ Nên tối ưu quá trình vì có thể làm cho CT kém thông minh, khó nhớ.

Kỹ thuật lặp trình (tt)

- Có gắng tối ưu chương trình cho tốt → phải thiết kế tốt các bước chương trình sẽ phải thực hiện.
- Kinh nghiệm : khi viết nên càng lâu thì càng phải tối ưu logic chương trình càng chặt chẽ.
- Bằng sự tối ưu của lãnh này ta hoàn toàn có thể mô phỏng cấu trúc nhiều khiếm và vòng lặp.

SUMMARY

- ✓ **Còn thể mô phỏng cấu trúc logic nhỏ ngôn ngữ cấp cao trong Assembly bằng lệnh JMP và LOOP.**
- ✓ **các lệnh nhảy : còn nhiều kiện và vô nhiều kiện.**
- ✓ **Khi gặp lệnh nhảy, CPU sẽ quyết định nhảy hay không bằng cách đưa vào giá trị thành ghi nhớ.**
- ✓ **các lệnh luận lý dùng để làm nhiều kiện nhảy là AND, OR, XOR, CMP ...**
- ✓ **Bắt đầu khi nào còn thể, hãy tạo chức công trình thành các công trình con → nên giảm nhiều cấu trúc luận lý của công trình.**

Caâu hoûi

1. Giaû söû $DI = 2000H$, $[DS:2000] = 0200H$. Cho bieát ñòa chæ ôa nhôu toaùn haïng nguoaùn vaø keát quaû löu trong toaùn haïng ñích khi thöïc hieän leänh $MOV DI, [DI]$
2. Giaû söû $SI = 1500H$, $DI=2000H$, $[DS:2000]=0150H$. Cho bieát ñòa chæ ôa nhôu toaùn haïng nguoaùn vaø keát quaû löu trong toaùn haïng ñích sau khi thöïc hieän leänh $ADD AX, [DI]$
3. Coù khai baùo A DB 1,2,3
Cho bieát trò cuûa toaùn haïng ñích sau khi thi haønh leänh $MOV AH, BYTE PTR A$.
4. Coù khai baùo B DB 4,5,6
Cho bieát trò cuûa toaùn haïng ñích sau khi thi haønh leänh $MOV AX, WORD PTR B$.

Bài tập LẬP TRÌNH

Bài 1 : Cù vøng nhò VAR1 dæi 200 bytes trong ñoài ñòic chæ bôu DS.

Vieát chöng trìn ñéám sốá chö 'S' trong vøng nhò nøy.

Bài 2 : Cù vøng nhò VAR2 dæi 1000 bytes. Vieát chöng trìn chuyè ñoài cùc chö thöøng trong vøng nhò nøy thønh cùc kỳ tõi hoa, cùc kỳ tõi cøñ læi khöng ñoài.

Bài 3 : Vieát chöng trìn nhæp 2 sốá nhò hñ 10.
In ra tởng cùu 2 sốá ñò.

Baøi taäp LAÄP TRÌNH

Baøi 4 : Vieát chöông trình nhaäp 2 soá baát kyø.

In ra toång vaø tích cuûa 2 soá ñoù. Chöông trình cou daïng sau :

Nhaäp soá 1 : 12

Nhaäp soá 2 : 28

Toång laø : 40

Tích laø : 336

Baøi 5 : Vieát chöông trình nhaäp 1 kyù töï.
Hieån thò 5 kyù töï keá tieáp trong boä maõ
ASCII.

Ex : nhaäp kyù töï : a

5 kyù töï keá tieáp : b c d e f

Baøi taäp LAÄP TRÌNH

Baøi 6 : Vieát chöông trình nhaäp 1 kyù töi.
Hieån thò 5 kyù töi ñöùng tröôùc trong boã mã ASCII.

Ex : nhaäp kyù töi : f

5 kyù töi keá tieáp : a b c d e
Baøi 7 : Vieát chöông trình nhaäp 1 chuoãi kyù töi.

In chuoãi ñaõ nhaäp theo thòu töi ngöôïc.

Ex : nhaäp kyù töi : abcdef

5 kyù töi keá tieáp : fedcba