Istruzioni di modifica della sequenza di elaborazione

- **Permettono di modificare la sequenza di esecuzione delle istruzioni di un programma, normalmente controllata dal meccanismo automatico di avanzamento del P.C.**

  - ***Salti***
    - ***Condizionati***
    - ***Incondizionati***
  - ***Iterazioni o loop***
  - ***Chiamate a procedure e Ritorno da procedure***

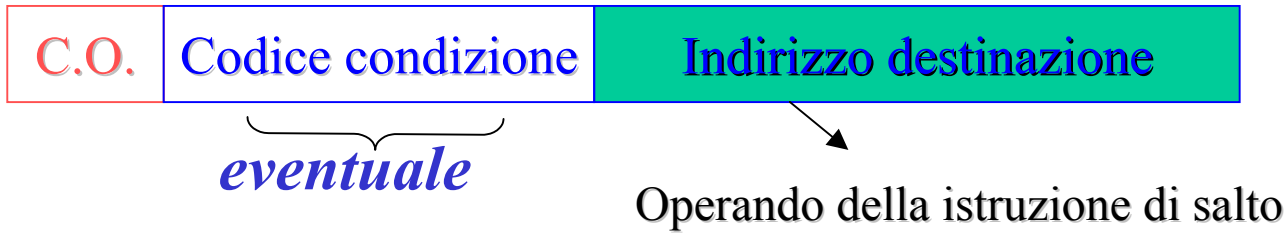
ISTRUZIONI DI TRASFERIMENTO DEL CONTROLLO
nel processore INTEL 8088

✓**Istruzioni di Salto :** **JMP JNO**
JE JNE
JZ JNZ JCXZ
JL JG
JNGE JNLE….

✓**Istruzioni di Ciclo :** **LOOP**
LOOPE LOOPZ
LOOPNE LOOPNZ

✓**Istruzioni di salto a Procedura :** **CALL**
RET

Formato di una istruzione di salto



- **L'indirizzo di destinazione può essere specificato con:**
  - *Indirizzamento assoluto*: l'indirizzo è nella istruzione
  - *Indirizzamento a registro indiretto*: l'indirizzo è nel registro specificato nell'istruzione
  - *Indirizzamento relativo*: la distanza (displacement) tra destinazione ed istruzione di salto è nella istruzione
    - Spostamento rispetto al contenuto del contatore di programma (codifica con pochi bits)

Istruzioni di salto

Salto incondizionato (*Jump*)

- Effettuano il salto all'indirizzo di destinazione senza condizioni

Salto condizionato (*Jcond*)

- Controllano una condizione ed effettuano il salto all'indirizzo di destinazione solo se la condizione è verificata
- Il controllo della condizione è effettuato testando l'informazione presente in bit singoli (flag dei codici di condizione presenti nello Status Word register)

Salto incondizionato nel processore Intel 8088

JMP ***(JUMP)***

Trasferisce il controllo all'operando specificato

Sintassi : ***JMP etichetta***

Esempio :

```
JMP   ETICHETTA_ NEAR
JMP   SHORT   ETICHETTA
JMP   ETICHETTA_FAR
JMP   VET[ DI ]
JMP   AX
```

Esistono cinque tipi diversi di JMP a seconda del tipo di etichetta :

- **NEAR (STESSO SEGMENTO) diretto**
- **SHORT diretto**
- **FAR (DIVERSO SEGMENTO) diretto**
- **FAR indiretto**
- **NEAR indiretto**

Salto incondizionato nel processore Intel 8088

JMP *modifica...*

- **IP** se all'interno di uno stesso segmento di programma
  INTRASEGMENTALE
- **IP e CS** se all'esterno del segmento di programma
  INTERSEGMENTALE

JMP

- **Diretto** *es: JMP 80* { **da -128** ; **a + 127** }
- **Indiretto** *es: JMP SI*

SALTI CONDIZIONATI

SI BASANO SUL VALORE DEL FLAG DI CONDIZIONE DELLA STATUS WORD.

Si dividono in 4 categorie :

- ***FLAG SEMPLICE***
- ***RELAZIONI ARITMETICHE TRA NUMERI CON SEGNO***
- ***RELAZIONI ARITMETICHE TRA NUMERI SENZA SEGNO***
- ***COSTANTI BOOLEANE***

IF CC THEN PC = PC + DISPL. ELSE PC = PC + 1

NB : NON ESISTONO ISTRUZIONI DI SELEZIONE.

Istruzioni di salto condizionato

Istruzioni di elaborazione dati
Istruzioni di confronto

Istruzioni di salto condizionato

Modificano il ...

Flag di condizione

Status Word register
o PSW (program status word)

Flag Carry

Flag zero

= 0 se il risultato dell'operazione è diverso da zero

=1 se il risultato dell'operazione è uguale a zero

Salto condizionato nel processore Intel 8088

JE / JZ / JCXZ

(JUMP if EQUAL / JUMP if ZERO / JUMP if CX=0)

Trasferiscono il controllo all'operando, in base al risultato della operazione precedente.

Flag coinvolti: **ZERO, SEGNO, RIPORTO, OVERFLOW, PARITA'**

Sintassi : **JE | JZ | JCXZ etichetta**

Esempio :

```
CMP  AX, BX        oppure      SUB  AX, BX
JE   ET2                       JZ   UGUALI
INC  AX                        INC AX
ET2: INC BX                    UGUALI: INC BX
```

JCXZ PIPPO **> salta a PIPPO se CX = 0 > CMP CX, 0**

```
JE   PIPPO
.....
PIPPO: .....
```

Effetto: PC ← PC + (PIPPO – PC)

ITERAZIONI o LOOP

Necessità di gestire il ciclo mediante:

- ***UN CONTATORE DELLE ITERAZIONI***
- ***UN MECCANISMO DI CONTROLLO DEL CICLO***

SCHEMI GENERALI DELLE ITERAZIONI:

a) COUNT = COUNT – 1
IF COUNT # 0 THEN PC = PC + DISPL. ELSE PC = PC + 1

b) COUNT = COUNT – 1
IF ZFLAG=1 and COUNT # 0 THEN PC = PC + DISPL. ELSE PC = P

c) IF CC THEN Dn = Dn – 1
NEXT ... IF Dn# 1 THEN PC = PC + DISPL. ELSE PC = PC +

Istruzioni di iterazione

- Istruzioni che facilitano la esecuzione di un *gruppo di istruzioni* per un numero fisso di volte
- Comprendono :
  - Operazione di aggiornamento di una variabile (contatore)
  - Operazione di controllo di una condizione
    - A fine ciclo
    - A inizio ciclo
  - Operazione di salto condizionato

Iterazioni nel processore Intel 8088

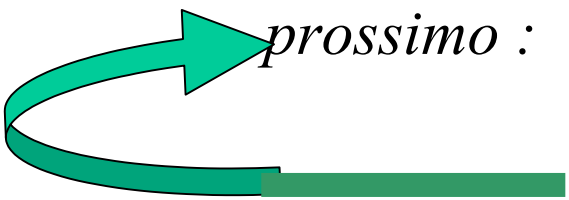
LOOP / LOOPE / LOOPZ / label

Decrementa il contatore CX e trasferisce il controllo all'etichetta se CX non e' zero ed in base al risultato del valore del flag Z.

Sintassi :

LOOP | LOOPE |LOOPZ | LOOPNE| LOOPNZ *etichetta*

Esempio >



```
                MOV CX, LENGHT VET
prossimo :      INC SI
                CMP VET[ SI], 0
                LOOPE  prossimo
                JNE nonzero
                ……..
nonzero:        ……..
```

CX=CX-1
se (CX=0) and (ZF=1) salta a prossimo
altrimenti prosegui

Nota : L'etichetta deve essere di tipo SHORT

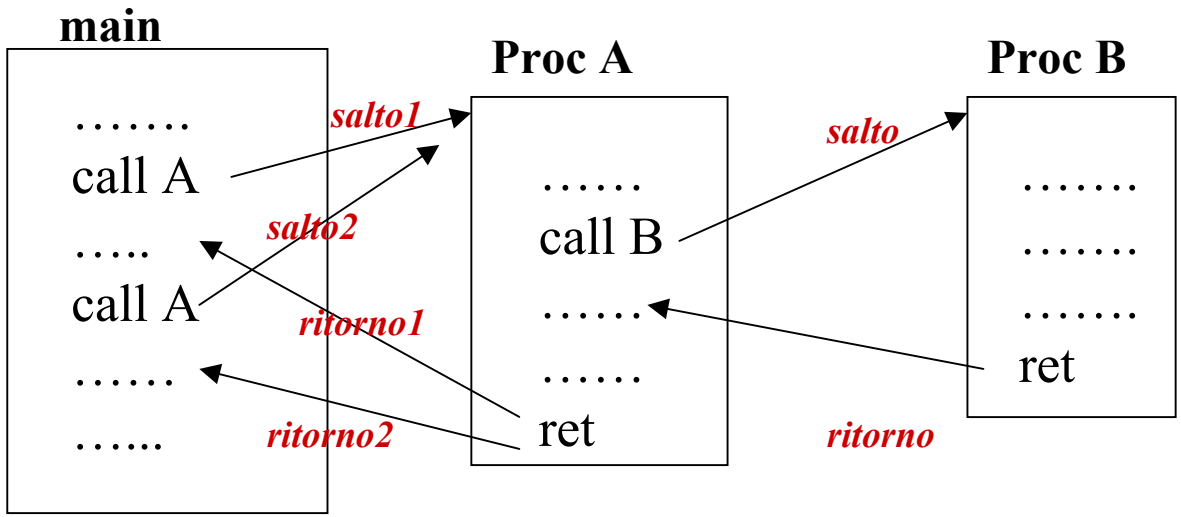
Istruzioni di chiamata a procedura

- Un programma può essere organizzato in *"moduli "*, denominati ***procedure***

  - Procedure
  - Subroutine
  - Function
  - Coroutine

  → Insieme di istruzioni (definito dal programmatore) richiamabile più volte all'interno del programma

- Una istruzione di chiamata a procedura può essere vista come una *istruzione macchina complessa*



PROCEDURA = UNITA' DI PROGRAMMA

PROGRAMMA
PROG A
.....
.....
.....
.....
END PROC A
PROC B
PROC C
.....
END PROC C
END PROC B
END PROGRAMMA

Nidificazione di procedure

esempio di procedure nidificate

```
MAIN    PROC FAR
.....
CALL Calcolo
CALL Output
RET
;------PROCEDURE
Calcolo PROC NEAR
  CMP totale,0
  JE  fine    ;nulla da sommare!
  MOV  CL,totale ;Iniz. CX per ciclo
Cic1: ADD  AL,vettore[BX]
  ADC  AH,0    ;AL <- AH + (0) + CF
  INC  BX      ;Incremento indice
  LOOP Cic1
  MOV  somma,AX
  fine:  RET
Calcolo  ENDP
Output  PROC NEAR
   CALL    Visual   ;Stampa stringa
   .....
   RET
Output ENDP
Visual  PROC NEAR
      PUSH    AX
      MOV     AH,9
      INT     21h
      POP     AX
      RET
Visual  ENDP
MAIN      ENDP
CODE      ENDS
END       MAIN
```

Istruzioni di chiamata a procedura

- ✓ L' istruzione di chiamata (***Call***) è un'istruzione di salto, il cui operando è l'indirizzo della prima istruzione della procedura, detto "***entry-point address***"

- ✓ L'istruzione di ritorno (***Return***), al termine della esecuzione della procedura chiamata, è una istruzione di salto indietro alla procedura chiamante (alla istruzione che segue immediatamente la istruzione di call):
  - Necessità di conservare l'indirizzo di tale istruzione, detto " ***return address*** "

Istruzioni di chiamata a procedura

- Il ***return address*** può essere conservato in:
  - **Registri o locazioni di memoria**
    - Difficoltà di chiamare un'altra procedura all'interno di quella chiamata (procedure nidificate)
  - **Locazione di memoria associata alla procedura**
    - L'istruzione di return è una istruzione di salto alla locazione di memoria che conserva l'address della istruzione della procedura chiamante da eseguire
    - È possibile la nidificazione delle procedure
  - **Stack**
    - È possibile sia la nidificazione delle procedure che la ricorsività (abilità di una procedura a chiamare se stessa)
    - Soluzione adottata in quasi tutte le architetture ISA

Passaggio dei parametri tra procedure (1)

- La procedura chiamante può passare alla procedura chiamata dei valori detti “parametri” o argomenti
- La procedura chiamata fornisce dei “parametri” (risultati) alla procedura chiamante
- I parametri presenti nella lista della procedura chiamata all’atto della sua definizione sono detti ***parametri formali o virtuali***
- I parametri presenti nella lista della procedura chiamante all’atto della chiamata sono detti ***parametri effettivi o reali***
- I parametri effettivi devono essere in corrispondenza esatta con quelli formali

Metodi di passaggio di parametri (2)

- La procedura chiamante ***pone i parametri*** in una "parameter list" e trasmette l'address della lista alla procedura chiamata
- La procedura chiamante ***pone gli address dei parametri*** in una "parameter-address list" e trasmette l'address della lista alla procedura chiamata
- La procedura chiamante ***pone i parametri nello stack***, prima di effettuare la chiamata ***(trasmissione per valore)***
- La procedura chiamante pone ***gli address dei parametri nello stack (trasmissione per indirizzo)***

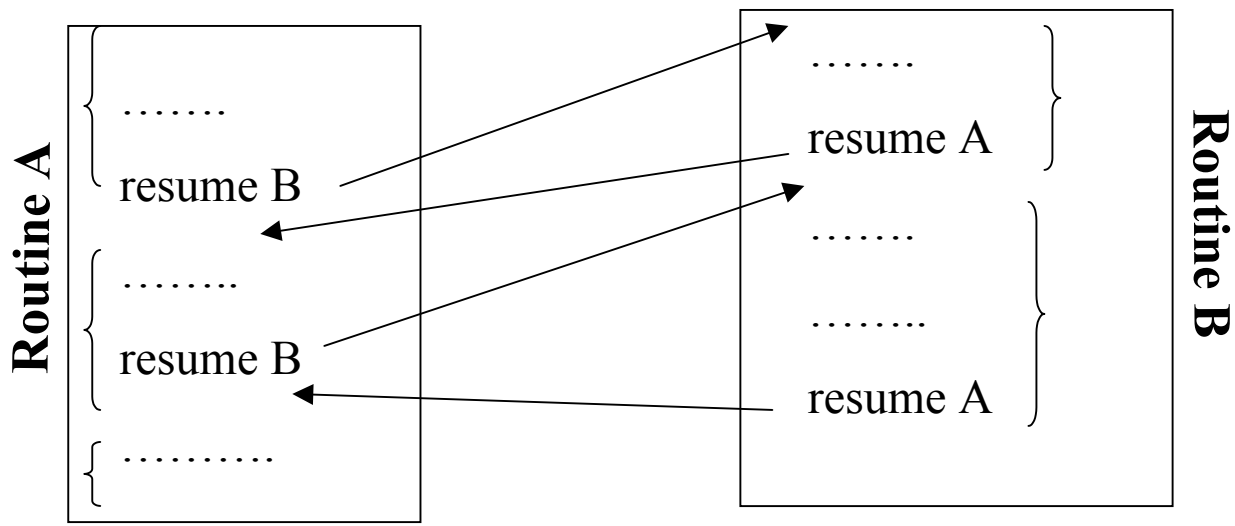
SALTO a CO-ROUTINE

LE COROUTINE SONO PROGRAMMI CHE SI TRASMETTONO IL CONTROLLO ALTERNATIVAMENTE.

✓**L'ISTRUZIONE DI SALTO IN QUESTO CASO CONSISTE IN UNO SCAMBIO TRA IL P.C. E IL TOP DELLO STACK, che contiene l'indirizzo della coroutine.** *(SP) < B*

tmp < ((SP)), (SP) < (PC), PC < (tmp)

✓**NECESSITA' DI SALVARE IL CONTENUTO DELLA STATUS WORD DELLE 2 COROUTINE OGNI VOLTA CHE AVVIENE UN SALTO E RIPRISTINARE TALI INFORMAZIONI AL SALTO SUCCESSIVO**



MACRO

(da non confondere con le subroutine)

PSEUDO – OPERATORE per generare una data sequenza in istruzioni, richiamabile più volte all'interno del programma.

NON PROVOCA UN SALTO !! ma una sostituzione in fase di compilazione !

FORMATO : *nome MACRO dummy list*

……..

ENDM

Esempio :

```
GEN  MACRO  XX, YY, ZZ
     MOV   AX, XX
     ADD   AX, YY
     MOV   ZZ, AX
ENDM
```

Definizione della macro nel programma :

Uso della macro nel programma :

```
GEN    AD, KISER, SUM
```

Che viene tradotto dall'assemblatore in

```
MOV    AX, AD
ADD    AX, KISER
 MOV   SUM, AX
```

Istruzioni per elaborare stringhe

- Stringa: insieme di caratteri alfanumerici (byte)
- Per elaborare una stringa occorrono:
  - puntatore alla stringa, contatore, istruzioni di ripetizione
- Due categorie:
  - Istruzioni per elaborare un elemento della stringa (un singolo carattere)
    - Orientate al byte
  - Istruzioni per elaborare tutta la stringa (tradotta dall'interprete in una sequenza di passi)

Istruzioni per elaborare stringhe

Tipi di istruzioni:

- ⇨ ***trasferimento (MOVE)***
- ⇨ ***caricamento in aree di memoria (LOAD e STORE)***
- ⇨ ***confronto (COMPARE, SCAN STRING)***
- ⇨ ***ripetizione operazione (REPEAT)***

Gli operandi in queste istruzioni sono in genere impliciti.

Ad es. nel processore Intel808x i registri SI e DI vengono implicitamente usati come puntatori all'inizio della stringa sorgente e destinazione e CX come registro puntatore

Istruzioni per elaborare stringhe nel processore 8088

Utilizzano le locazioni indirizzate dai registri SI (origine) e DI (destinazione) come indirizzo iniziale e finale delle stringhe con un controllo del flag DF per stabilire la direzione dell'operazione e CX come contatore

MOVS	**MOVSB**	**MOVSW**
LODS	**LODSB**	**LODSW**
STOS	**STOSB**	**STOSW**
CMPS	**CMPSB**	**CMPSW**
SCAS	**SCASB**	**SCASW**

REP REPE REPNE REPZ REPNZ

Istruzioni per elaborare stringhe nel processore 8088

MOVS str_dest, str_sorg MOVSB MOVSW

trasferisce un byte o una parola dalla stringa sorgente alla stringa destinazione

es: *MOV SI, OFFSET SORG* *oppure* *REP MOVSB* (senza operandi)
MOV DI, OFFSET DEST
REP MOVS DEST, SORG *REP* = spostamento effettuato CX volte

LODS stringa LODSB LODSW

trasferisce l'operando dall'indirizzo SI all'accumulatore (AX) ed aggiorna SI di 1 o 2 a seconda del tipo di operando

es: *CLD*
LODS STRINGA *oppure* *LODSB* (senza operando)

STOS stringa STOSB STOSW

trasferisce il byte o la parola dall'accumulatore (AX) nella locazione specificata da DI e lo aggiorna di 1 o 2

es: *STOS STRING* *oppure* *STOSB* (senza operando)

Istruzioni per elaborare stringhe nel processore 8088

CMPS destin,origin CMPSB CMPSW

sottrae il byte o la parola della stringa sorgente dalla stringa destinazione e setta il flag zero

es: *MOV SI, OFFSET SORG* *oppure* *CMPSB* (senza operandi)
MOV DI, OFFSET DEST
CMPS DEST, SORG ovvero effettua DEST - SORG da testare con un jump condizionato sul flag zero

SCAS destin SCASB SCASW

sottrae il byte o la parola della stringa specificata (in destin) dall'accumulatore (AL) e setta il flag zero

es: *MOV AL, OFFSET SORG* *oppure* *SCASB* (senza operandi)
MOV DI, OFFSET DEST
SCAS DEST ovvero effettua AL - DEST , da testare con un jump condizionato sul flag zero

REP REPE REPNE REPZ REPNZ

ripete l'istruzione su stringa che segue finchè CX è maggiore di zero

es: *REP MOVS DEST,ORIG* oppure *REPNZ SCAS DESTINAZIONE*

Istruzioni per elaborare indirizzi

- Indirizzo: dato "intero binario senza segno"

  Un indirizzo può essere contenuto in:

  - Registri specifici (es. Stack pointer, registri indice)
  - Registri concatenabili
  - Locazioni di memoria

- Istruzioni che accedono all'indirizzo
  - L'indirizzo è prima calcolato secondo le modalità descritte nel campo operandi e poi trasferito