LABORATORIO DI ARCHITETTURA DEI CALCOLATORI

lezione n° 15

Prof. Rosario Cerbone

rosario.cerbone@libero.it

http://digilander.libero.it/rosario.cerbone

a.a. 2005-2006

L'INDIRIZZAMENTO NEL PROCESSORE MC 68000

- L'indirizzamento è la modalità con cui si specificano gli operandi nelle istruzioni.
- Questi operandi possono essere delle costanti espresse direttamente nelle istruzioni, dei registri o degli indirizzi in memoria, e possono comparire come sorgente o come destinazione delle operazioni da eseguire.
- Le tecniche di indirizzamento specificano come la CPU determina gli indirizzi effettivi (EA) degli operandi.

Modi di indirizzamento

Indirizzamenti Diretti	Notazione	Esempio
Diretto con Registro Dati	Dn	D3
Immediato	im	$A3F7
Diretto con Registro Indirizzi	An	A5
Assoluto	Addr16 Addr 32	$6527 $54329876

Modi di indirizzamento

Indirizzamenti Indiretti	Notazione	Esempio
con Registro Indirizzi	(An)	(A5)
con Registro Indirizzi con PostIncremento	(An)+	(A1)+
con Registro Indirizzi con PreDecremento	-(An)	-(A3)
con Registro Indirizzi e 16bit Displacement (detto anche **based**)	Offset16(An)	$f54(A2)
con Registro Indirizzi e con Indice e Spiazzamento	Offset8(An,Rn) Offset8(An,Rn.L)	$30(A5,A6) $65(A2,A3.L)
Relativo a PC	Offset16(PC)	$68A9(PC)
Relativo a PC con indice e spiazzamento	Offset8(PC,Rn)	$2EF$(PC;A4)

Modi di indirizzamento

- ***Register Direct***
  - Data-register Direct
  - Address-register Direct
- ***Address-register Indirect***
  - Address-register Indirect
  - Auto-Increment
  - Auto-Decrement
  - Based
  - Based Indexed
- ***Absolute***
  - Short
  - Long
- ***PC Relative***
- ***PC Relative Indexed***
- ***Immediate (or Literal)***

La Dimensioni dei Dati Indirizzati

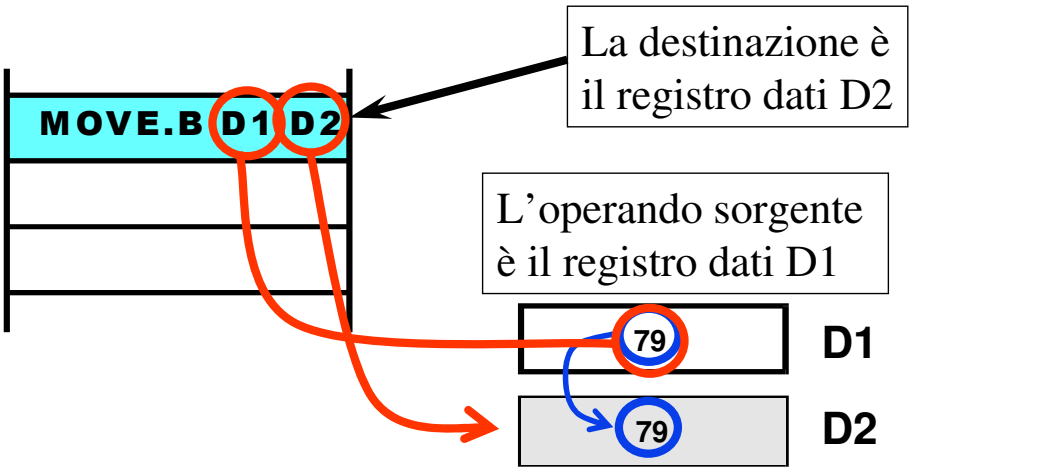
- Le istruzioni finora viste specificano ogni volta la dimensione dei dati cui deve essere fatto accesso (B .W .L).
- In generale è sempre una buona idea specificare la dimensione del dato, ma può non essere necessario.
- Qualora la dimensione del dato non sia specificata, l'assemblatore assume che si intenda accedere ad un dato di tipo Word (2 byte) in accordo con l'organizzazione della memoria a word.

Register Direct Addressing

Indirizzamento Diretto con Registro Dati e con Registro Indirizzi

- L'operando sorgente/destinazione di un istruzione è un registro dati/indirizzi:
- operando sorgente - il contenuto del registro specificato fornisce l'operando sorgente.
- operando destinazione - il registro viene caricato con il valore specificato dall'istruzione.
- MOVE.B D0,D3 Copia l'operando sorgente in D0 nel registro D3
- SUB.L A0,D3 Sottrae l'operando sorgente nel registro A0 dal registro D3
- CMP.W D2,D0 Confronta l'operando sorgente nel registro D2 con il registro D0
- ADD D3,D4 Somma l'operando sorgente nel registro D3 al registro D4

Register Direct Addressing



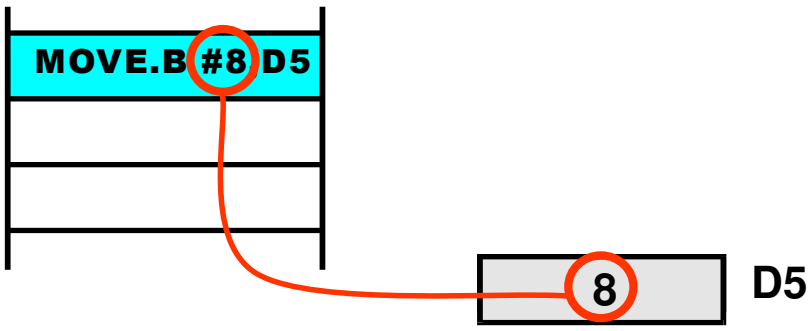
Register Direct Addressing - Caratteristiche

- È veloce: non richiede accessi alla memoria esterna
- Istruzioni corte: il Register Direct Addressing richiede soltanto tre bit (otto registri)
  - Mode = 0, reg = 0-7 per Dn
  - Mode = 1, reg = 0-7 per An
- I programmatori usano il ***Register Direct Addressing*** per memorizzare variabili usate di frequente (***scratchpad storage***)

Immediate Addressing o literal

- L'operando è espresso direttamente nell'istruzione
- Può essere usato solo per operandi sorgente
- Si esprime anteponendo il simbolo **#** all'operando sorgente
- Un operando immediato è anche chiamato ***literal***
- Esempio:
- MOVE.B #8,D5 Usa l'operando sorgente immediato 8

Immediate Addressing - Funzionamento



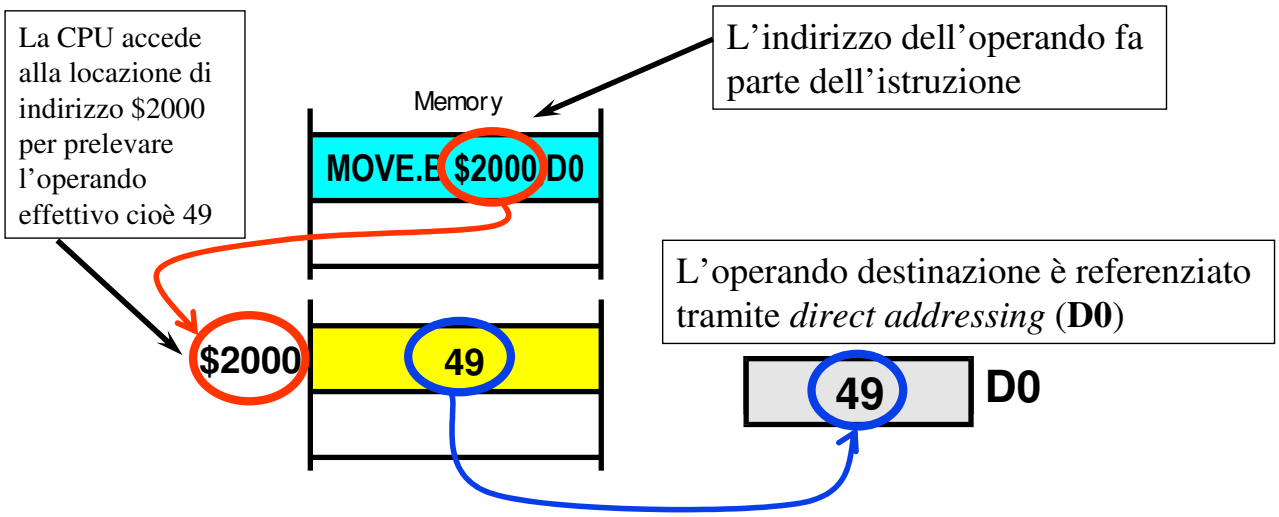
MOVE.B #8,D5

L'effetto di questa istruzione è quello di copiare il valore del literal 8 nel registro dati D5

Absolute Addressing (o Direct Addressing)

- L'operando espresso nell'istruzione specifica l'indirizzo di memoria a cui accedere per leggere\scrivere il valore su cui agire
- Richiede un ulteriore accesso in memoria (oltre quello già eseguito per prelevare l'istruzione) per accedere all'operando effettivo
- Esempio:
  - CLR.B $2000 azzera il contenuto della locazione di memoria $2000

Absolute Addressing - Funzionamento



L'effetto di MOVE.B $2000,D0 è quello di leggere il contenuto della locazione di memoria $2000 e copiarlo nel registro D0

Esempio modi fondamentali

Consideriamo questo statement in linguaggio di alto livello:
char Z, Y = 27;
Z = Y + 24;

Il seguente frammento di codice implementa questo costrutto

```
        ORG     $8000       Inizio del codice
        MOVE.B  Y,D0        Direct addressing
        ADD     #24,D0      Operando literal
        MOVE.B  D0,Z         Direct addressing

        ORG     $8100       Inizio dell'area dati
Y       DC.B    27          Memorizza la costante 27 in memoria
Z       DS.B    1           Riserva un byte per Z
```

Riepilogo modi fondamentali

Register direct addressing- usato per variabili d'uso frequente che conviene mantenere nei registri di macchina

Literal (immediate) addressing- usato per valori costanti espressi direttamente dal programmatore

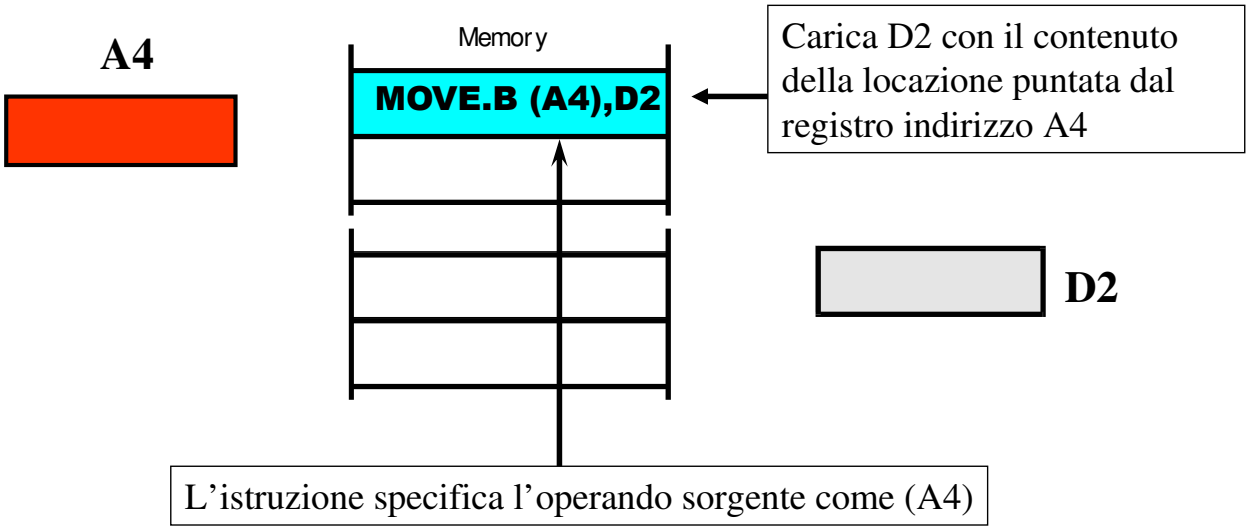
Direct (absolute) addressing- usato per variabili che risiedono in memoria centrale

Address Register Indirect Addressing

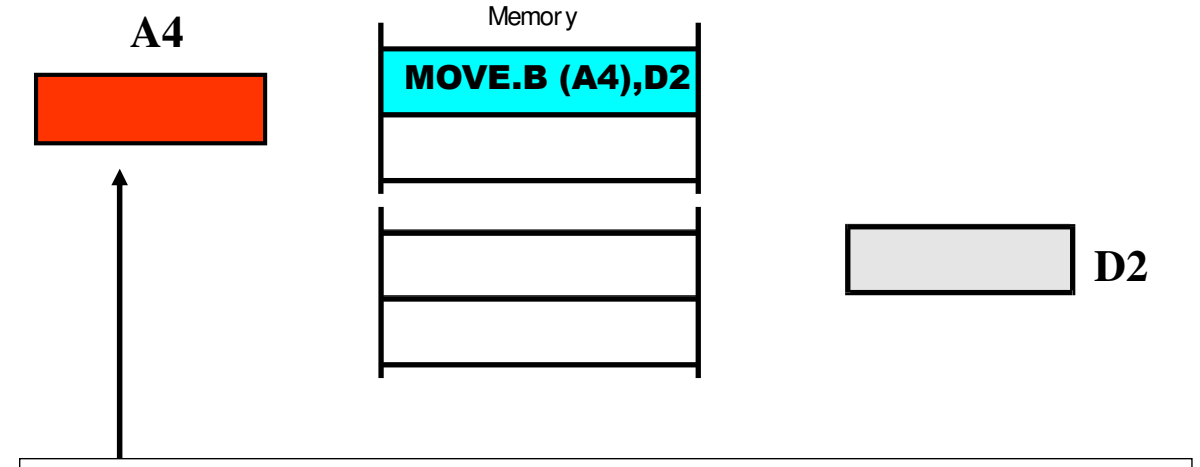
- L'istruzione specifica uno dei registri indirizzo
- Il registro indirizzo specificato contiene l'indirizzo effettivo dell'operando
- Il processore accede all'operando puntato dal registro indirizzo

- Esempio:
  - MOVE.B (A4),D2

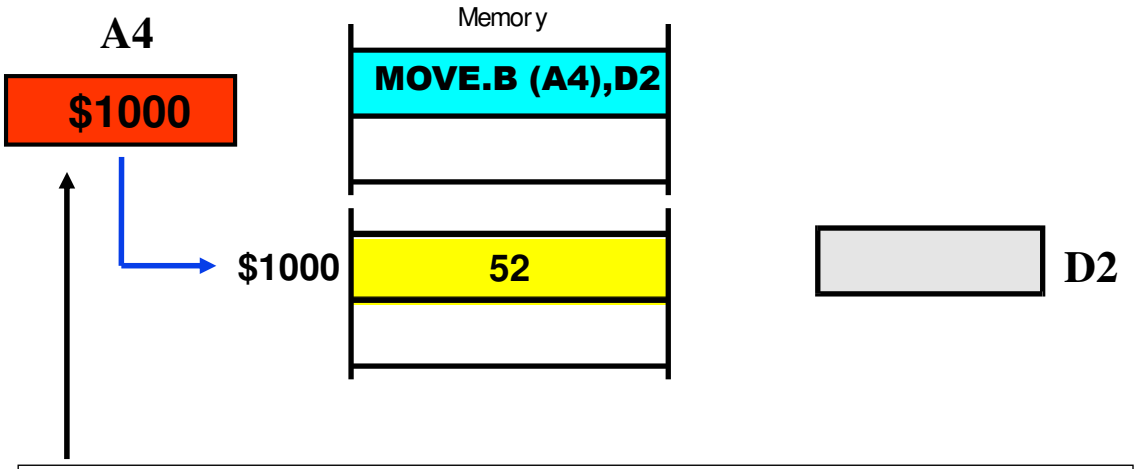
Address Register Indirect - Funzionamento



Address Register Indirect - Funzionamento

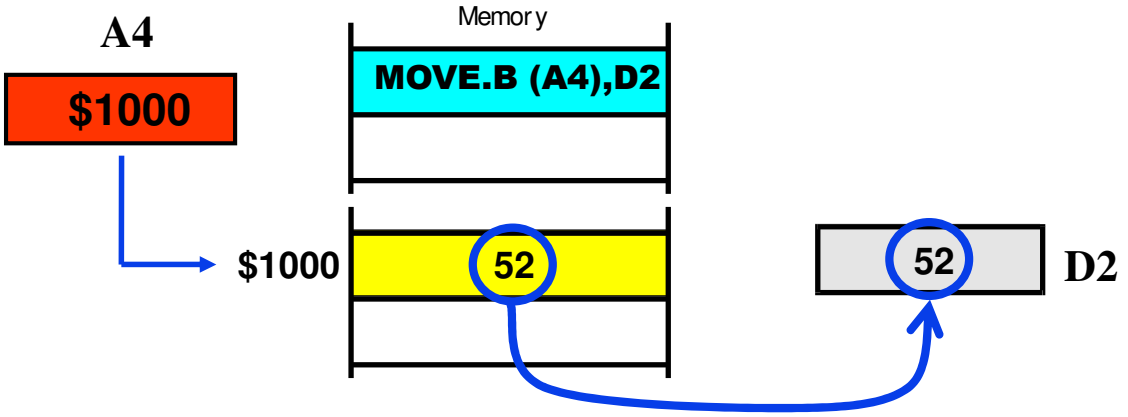


Address Register Indirect - Funzionamento



Il registro indirizzo è dunque usato per accedere all'operando in memoria

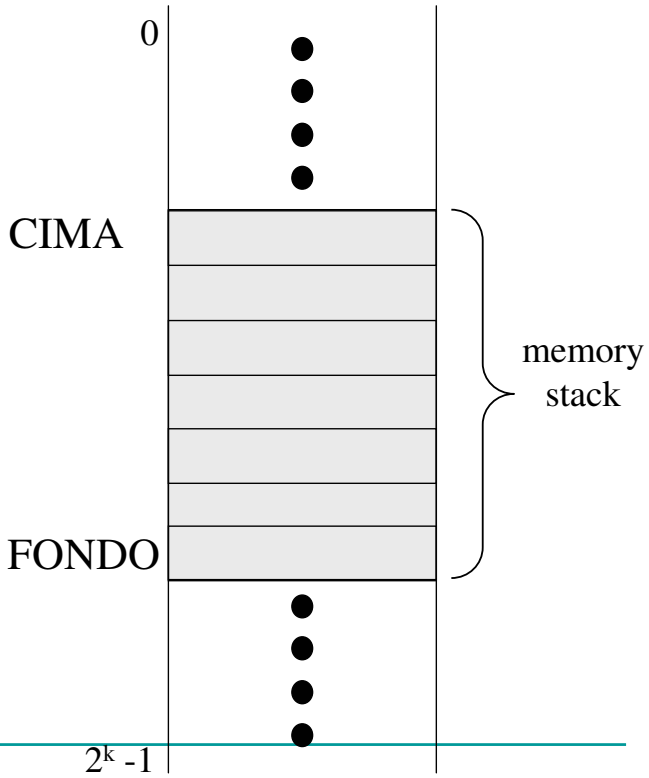
Address Register Indirect - Funzionamento



Alla fine, il contenuto della locazione puntata da A4 viene copiato nel registro dati

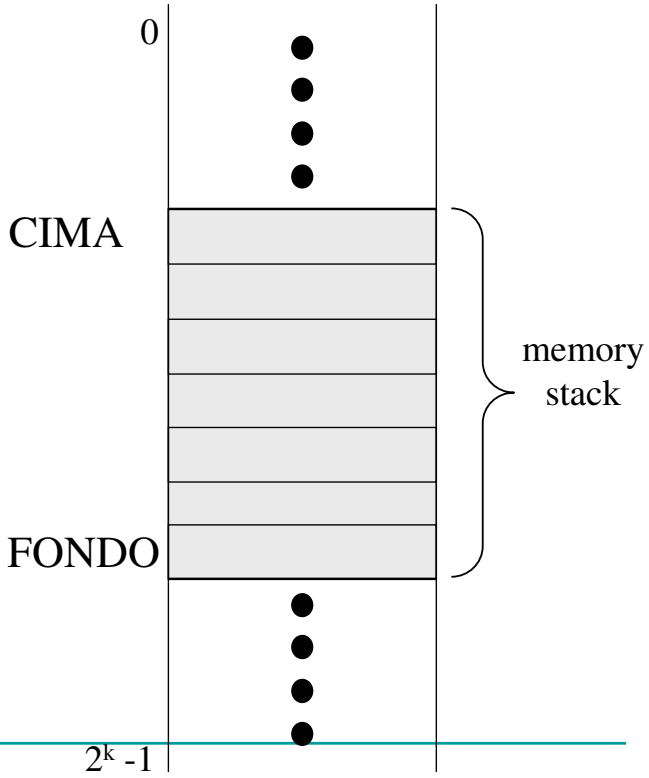
Lo stack - allocazione

Si riserva una zona di memoria destinata ad implementare lo stack



Lo stack - allocazione

Si riserva una zona di memoria destinata ad implementare lo stack



Lo **Stack Pointer** (SP) punta al fondo dello stack

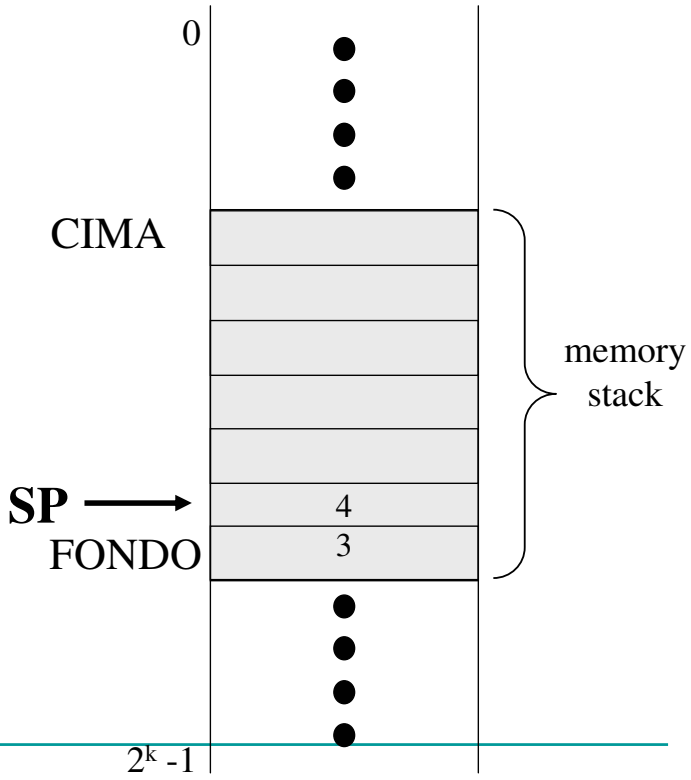
Lo stack - PUSH

PUSH:

Decrementa SP

Move DATO, (SP)

SP (**Stack Pointer**) punta all'ultima locazione occupata



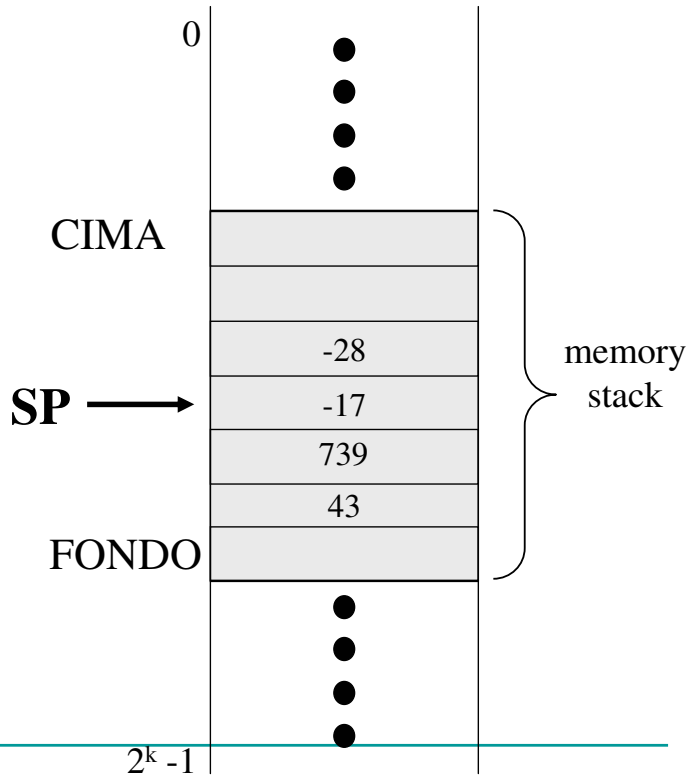
Lo stack - POP

POP:

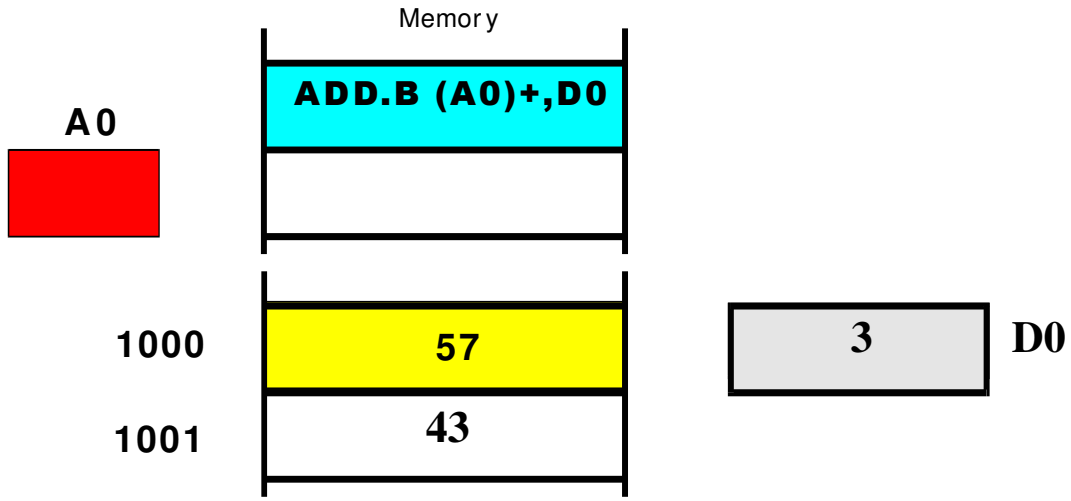
Move (SP), DEST

Incrementa SP

SP (**Stack Pointer**) punta all'ultima locazione occupata

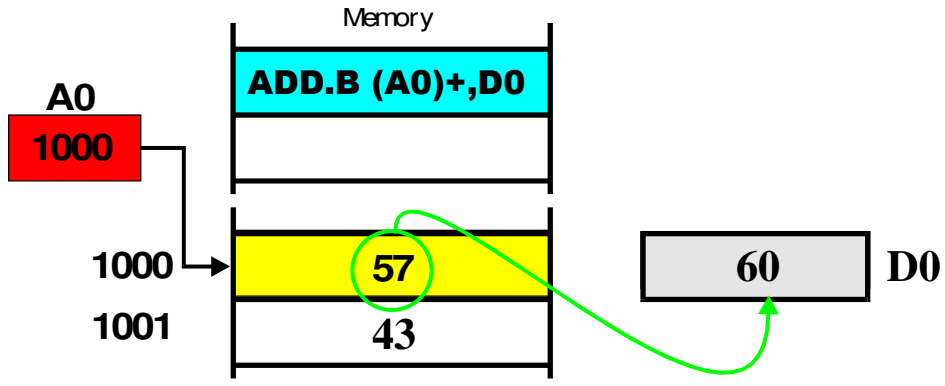


Auto-increment - Funzionamento



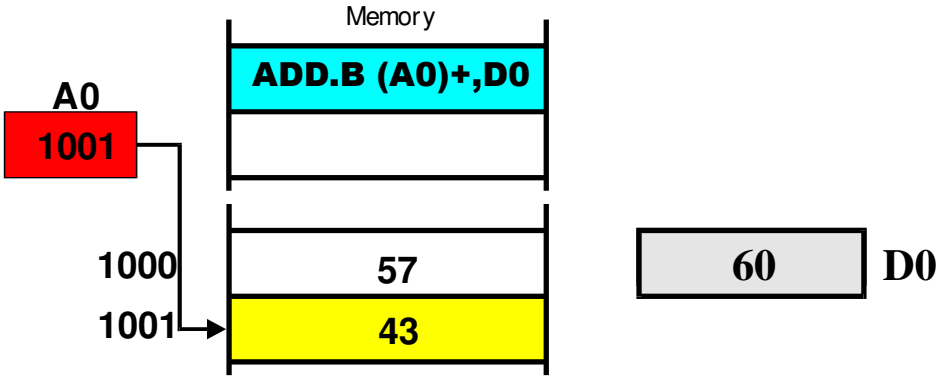
AO funge da stack pointer

Auto-increment - Funzionamento



Se **A0** contiene il valore 1000 il contenuto della locazione di memoria puntata da **A0** (57) viene sommato al contenuto di **D0** (57 + 3 = 60)

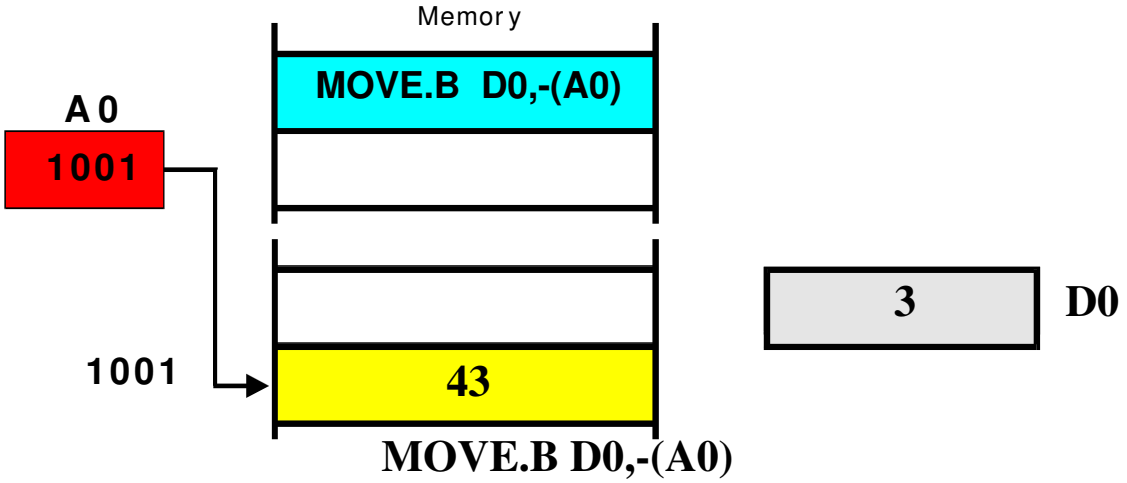
Auto-increment - Funzionamento



Eseguita la somma, il contenuto di **A0** viene automaticamente incrementato (di **1** perché il codice operativo ADD**.B** opera su byte) e quindi punta alla locazione successiva

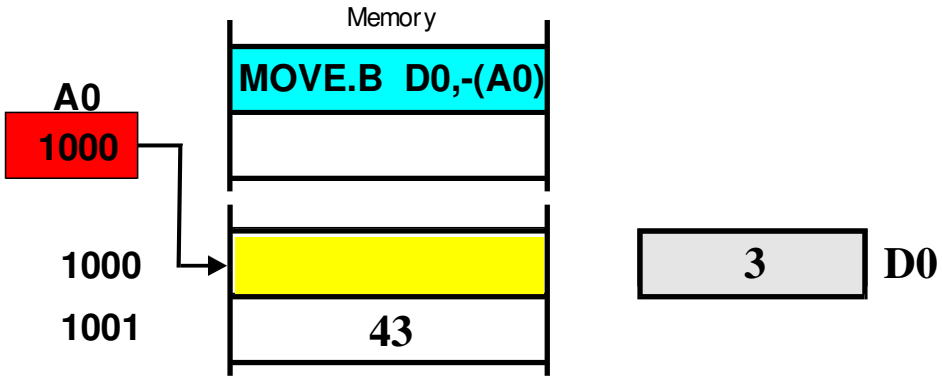
Auto-decrement (pre-decrement)

- **-(An)**: il contenuto del registro indirizzo è decrementato prima dell'uso di una quantità pari alla dimensione dell'operando



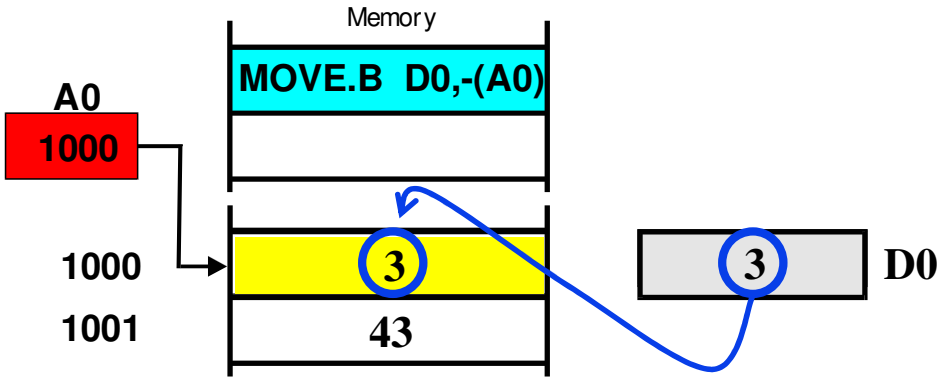
L'istruzione esegue il push di D0 sullo stack a cui punta A0 che funge da stack pointer.

Auto-decrement - Funzionamento



A0 viene preventivamente decrementato di 1 (MOVE.**B**), quindi punta alla locazione di memoria 1000

Auto-decrement - Funzionamento



Il contenuto di **D0** viene copiato nella locazione a cui punta **A0**

Based Addressing

- Based Addressing (Indiretto con displacement) determina l'EA sommando due componenti:

  $EA=disp_{16}+[A_n]$

  - *Displacement* ($disp_{16}$)- *espresso* da 16 bit nell'istruzione
  - *base address* ($[A_n]$)- contenuto in un registro

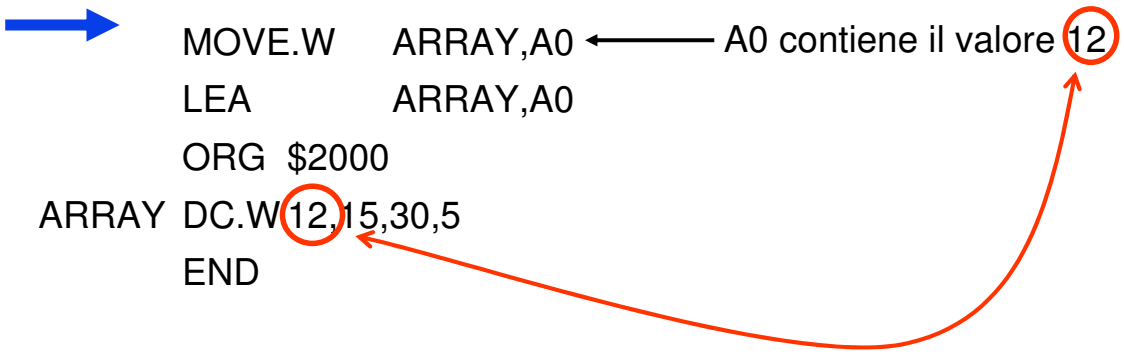
Es. **MOVE 4(A0),D1**
EA = 4 + [A0]; effettua l'operazione M[EA] → D1

Based Addressing

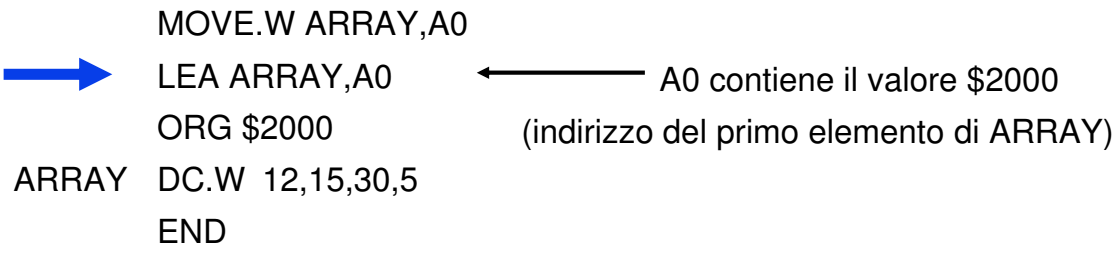
- Usato per accedere array e tabelle di cui si conosca, ad assembly time, la posizione relativa:

```
    MOVE.L  #TABLE,A0
    MOVE    0(A0),D0  ←——  D0 contiene il primo valore della tabella
    MOVE    2(A0),D1  ←——  D1 contiene il secondo valore
    MOVE    4(A0),D2  ←——  D2 contiene il terzo valore
TABLE   DC.W 1000,2000,3000
```

Istruzione LEA – Load Effective Address



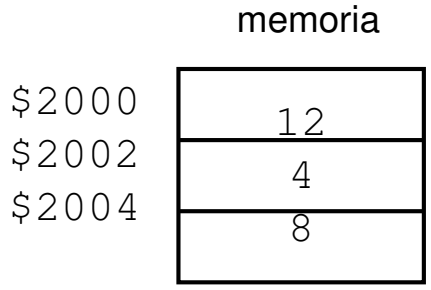
- LEA – Carica l'indirizzo dell'operando anziché il valore.



L'istruzione LEA

Un altro esempio di impiego dell'istruzione LEA:

```
        LEA     ARRAY,A0     ; A0 ← $2000
        MOVE.W  4(A0),D0     ; D0 ← 8
        LEA     4(A0),A1     ; A1 ← $2004
        ORG     $2000
ARRAY   DC.W    12,4,8
```



BASED INDEX

- ***Based Indexed Addressing***: determina l'***EA*** sommando tre componenti:

$$EA = \#d + A_j + R_i$$

  - **Displacement** (opzionale)- espresso da un intero con segno su 8 bit
  - ***Base register*** - registro di ***indirizzi*** contenente il base address
  - ***Index register*** – registro indirizzi o registro dati contenente l'offset

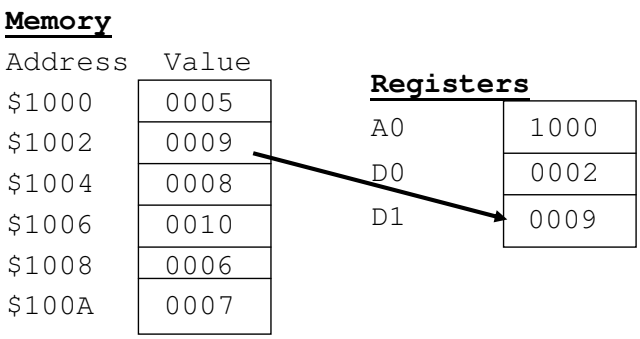
```
MOVE.W   (A0, D6), D2          *D2=M[A0+D6]
MOVE.W   DISP(A4,D3), D5       *D5=M[A4+D3+DISP]
```

Displacement Base Register Index Register

Based Indexed with displacement

MOVE 0(A0,D0),D1 effettua l'operazione M[EA] → D1

EA=0+1000+2=1002



PC relative Addressing

EA=disp16+[PC]

Calcola l'***EA*** sommando al valore corrente del ***PC*** il displacement (*8 - 16 bit*) specificato nell'istruzione

Es. **MOVE $100(PC),D1**

EA = $100 + [PC] effettua l'operazione M[EA] → D1

Program Counter indexed: $EA=disp_8+[PC]+[Xn]$

Es. **MOVE $0A(PC,D0),D1**

EA = $0A + [PC] + [D0] effettua l'operazione M[EA] → D1

Relative Indexed Addressing

- Funziona come il Based Indexed, ma il base register è sostituito dal PC
- Usato in genere per saltare ad aree di memoria read-only contenenti dati o istruzioni

ESEMPI DI INDIRIZZAMENTO

- I seguenti programmi realizzano l'equivalente dell'istruzione A=B+C+D usando vari modi di indirizzamento.
- Per ciascun esempio, oltre al programma sorgente, è indicato anche il programma oggetto.

ESEMPI DI INDIRIZZAMENTO

```
00000000                    1  * Es. 1
00000000                    2  * Ind. immediato, ass. e diretto
00000000                    3  * A=B+C+D
00000000                    4
00008000                    5            ORG    $8000
00008000 303C 0004          6  START     MOVE   #4,D0
00008004 5640               7            ADD    #3,D0
00008006 0640 0009          8            ADD    #9,D0
0000800A 33C0 00008800      9            MOVE   D0,A
00008010 4E72 2000          10           STOP   #$2700
00008014                    11
00008014                    12
00008800                    13           ORG    $8800
00008800                    14       A   DS     1
00008802                    15           END START
```

ESEMPI DI INDIRIZZAMENTO

```
00000000                          1  * Es. 2
00000000                          2  * Ind. assoluto e diretto
00000000                          3  * A=B+C+D
00000000                          4
00008000                          5              ORG     $8000
00008000  3039 00008802           6      START   MOVE    B,D0
00008006  D079 00008804           7              ADD     C,D0
0000800C  D079 00008806           8              ADD     D,D0
00008012  33C0 00008800           9              MOVE    D0,A
00008018  4E72 2000              10              STOP    #$2700
0000801C                         11
0000801C                         12
00008800                         13              ORG     $8800
00008800                         14  A   DS      1
00008802  0004                   15  B   DC      4
00008804  0003                   16  C   DC      3
00008806  0009                   17  D   DC      9
00008808                         18              END START
```

ESEMPI DI INDIRIZZAMENTO

```
00000000                          1  * Es. 3
00000000                          2  * Ind. immediato, diretto, indiretto
00000000                          3  * A=B+C+D
00000000                          4
00008000                          5              ORG     $8000
00008000  207C 00008802           6      START   MOVEA.L  #B,A0
00008006  227C 00008804           7              MOVEA.L  #C,A1
0000800C  247C 00008806           8              MOVEA.L  #D,A2
00008012  267C 00008800           9              MOVEA.L  #A,A3
00008018  3010                   10              MOVE    (A0),D0
0000801A  D051                   11              ADD     (A1),D0
0000801C  D052                   12              ADD     (A2),D0
0000801E  3680                   13              MOVE    D0,(A3)
00008020  4E72 2000              14              STOP    #$2700
00008024                         15
00008024                         16
00008800                         17              ORG     $8800
00008800                         18  A   DS      1
00008802  0004                   19  B   DC      4
00008804  0003                   20  C   DC      3
00008806  0009                   21  D   DC      9
00008808                         22              END START
```

ESEMPI DI INDIRIZZAMENTO

```
00000000                          1  * Es. 4
00000000                          2  * Ind. imm., dir., ind., ind. con post-inc
00000000                          3  * A=B+C+D
00000000                          4
00008000                          5                ORG       $8000
00008000  207C 00008800           6          START MOVEA.L #B,A0
00008006  3018                    7                MOVE      (A0)+,D0
00008008  D058                    8                ADD       (A0)+,D0
0000800A  D058                    9                ADD       (A0)+,D0
0000800C  3080                    10               MOVE      D0,(A0)
0000800E  4E72 2000               11               STOP      #$2700
00008012                          12
00008012                          13
00008800                          14               ORG       $8800
00008800  0004                    15 B      DC      4
00008802  0003                    16 C      DC      3
00008804  0009                    17 D      DC      9
00008806                          18 A      DS      1
00008808                          19               END START
```

ESEMPI DI INDIRIZZAMENTO

```
00000000                          1  * Es. 5
00000000                          2  * Ind. imm.,dir.,ass.,ind. con offset16
00000000                          3  * A=B+C+D
00000000                          4
00008000                          5                ORG       $8000
00008000  207C 00008800           6          START MOVEA.L #A,A0
00008006  3028 0002               7                MOVE      2(A0),D0
0000800A  D068 0004               8                ADD       4(A0),D0
0000800E  D068 0006               9                ADD       6(A0),D0
00008012  33C0 00008800           10               MOVE      D0,A
00008018  4E72 2000               11               STOP      #$2700
0000801C                          12
0000801C                          13
00008800                          14               ORG       $8800
00008800                          15 A      DS      1
00008802  0004                    16 B      DC      4
00008804  0003                    17 C      DC      3
00008806  0009                    18 D      DC      9
00008808                          19               END START
```

ESEMPI DI INDIRIZZAMENTO

```
00000000                                1  * Es. 6
00000000                                2  * Ind. imm.,dir.,ass.,ind. con offset8 ed ind.
00000000                                3  * A=B+C+D
00000000                                4
00008000                                5                ORG       $8000
00008000  207C 00008800                 6       START    MOVEA.L   #A,A0
00008006  3028 0002                     7                MOVE      2(A0),D0
0000800A  323C 0002                     8                MOVE      #2,D1
0000800E  D070 1002                     9                ADD       2(A0,D1),D0
00008012  5441                         10               ADD       #2,D1
00008014  D070 1002                    11               ADD       2(A0,D1),D0
00008018  33C0 00008800                12               MOVE      D0,A
0000801E  4E72 2000                    13               STOP      #$2700
00008022                               14
00008022                               15
00008800                               16               ORG       $8800
00008800                               17 A     DS        1
00008802  0004                         18 B     DC        4
00008804  0003                         19 C     DC        3
00008806  0009                         20 D     DC        9
00008808                               21               END START
```

Esercizio 15.1

Identificare l'addressing mode usato per gli operandi in ciascuna delle seguenti istruzioni

- **ADD.B (A5),(A4)**
- **MOVE.B #12,D2**
- **ADD.W TIME,D4**
- **MOVE.B D6,D4**
- **MOVE.B (A6)+,TEST**

Esercizio 15.2

- Scrivere un programma che sommi i 5 numeri componenti un array.
- Verificarne la correttezza con l'ASIM.

Esercizio 15.3

- Scrivere un programma che trovi il massimo ed il minimo tra i valori del vettore definito nell'esercizio precedente.
- Verificarne la correttezza con l'ASIM.

Esercizio 15.4

- Convertire una stringa in maiuscolo (sottrarre 32 al codice ASCII delle lettere minuscole).
- La stringa sia terminata da '0'
- Si supponga di chiamare STRINGA l'etichetta relativa all'indirizzo di partenza della stringa.

Esercizio 15.5

Tradurre il seguente frammento di pseudo-codice in linguaggio assembly:

- SUM = 0
- FOR J = 5 TO 19
- SUM = SUM + X(J)*Y(J)
- END FOR

- Suggerimenti:
- SUM è una variabile temporanea. La si può mettere in un registro ed usare il register direct addressing.
- J è una variabile temporanea, che tipicamente viene messa in un registro. J viene inizializzata al valore del literal 5.
- X(J) e Y(J) sono elementi di array, a cui tipicamente si accede mediante address register indirect addressing.