

**Architettura  
Elaboratori  
Elettronici**  
**ESERCITAZIONI**  
MODI INDIRIZZAMENTO

**Franco Liberati**  
**liberati@di.uniroma1.it**

# Argomenti

- Modi di indirizzamento MARS
  - Immediato
  - Assoluto
  - Immediato con spiazzamento
  - Simbolico con spiazzamento
  - A registro
  - Immediato a registro
  - Simbolico con spiazzamento a registro





# Modi di indirizzamento

(Generalità)

Un **modo di indirizzo esprime un indirizzo effettivo**, cioè consente di reperire un operando in Memoria Dati o accedere ad un'area della Memoria Istruzioni

Il modo di indirizzamento è espresso nel campo MODE delle istruzioni

Esistono modi di indirizzamento più o meno complessi che caratterizzano anche la tipologia di elaboratore (RISC s CISC)



# Modi di indirizzamento

MARS

Il MARS fornisce, per comodità, più modi di indirizzamento derivati dal suo **unico modo di indirizzamento** che si può esprimere come:

**etichetta + spiazzamento + contenuto di un registro**

Esempio:

**`lw $t1,array+256($t0)`**

*(mette nel registro \$t1, l'operando a 32bit sito all'indirizzo 268501260 assumendo: array = 268501000 e \$t0 = 4)*

NB: lo spiazzamento può essere riportato in esadecimale antepoendo il simbolo **`0xvaloreesadecimale`** es: **`lw $t1,array+0x100($t0)`** equivale a **`lw $t1, array+256($t0)`**

IMMEDIATO





# IMMEDIATO

Il **modo di indirizzamento immediato** (*immediate*) specifica un valore nel campo MODE.

L'operando è presente nel campo dell'istruzione subito dopo lo OPCODE

Nel MIPS si risolve con una pseduo-istruzione.

ESEMPIO

**li \$t0,12345**

*L'operando è espresso in decimale (o in esadecimale con il suffisso **0x**)*



# IMMEDIATO

Se il valore è rappresentabile in complemento a due con non più di 16 bit si risolve con una sola istruzione

Se il valore è rappresentabile in complemento a due con più di 16 bit l'istruzione si risolve con LUI e OR suddividendo il valore in due parti da 16 bit

Esempio:

li \$t0, 256  
li \$t1, 0x100  
#il contenuto di \$t0 è  
#uguale al contenuto di \$t1

Esempio

li \$t1, 0x11170 # 0000000000000001 0001000101110000

diventa

lui \$at, 0x1 #copia il valore nei 16bit più  
#significativi di \$at

or \$t1, \$at, 0x1170

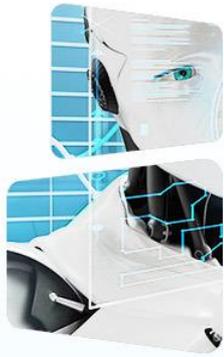
NB:

Il MIPS traduce l'istruzione

li \$t0, 256

in

addiu \$t0, \$zero, 256



# IMMEDIATO

**li \$t0, 0x100**

OPCODE LI

00000000 10000000

**addi \$t0,\$zero, 0x100**

**\$t0**

00000000 00000000 00000000 10000000



# IMMEDIATO

**li \$t0, 0xB3D05E00**

NB: 0xB2D05E00=10110010 11010000 01011110 00000000

**OPCODE LUI**

10110010 11010000

**\$at**

10110010 11010000 00000000 00000000

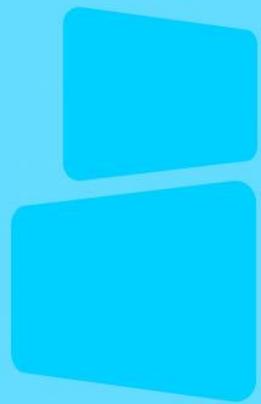
**OPCODE OR**

01011110 00000000

**\$t0**

10110010 11010000 01011110 00000000

**ASSOLUTO**





# ASSOLUTO

Il **modo di indirizzamento assoluto** (*absolute*) specifica un indirizzo

L'operando è presente nel campo dell'istruzione subito dopo l'OPCODE

ESEMPIO

**lw \$t0, 268501000**

*Il segmento dati in SPIM-MARS inizia dalla locazione 268500992 (0x1001000) fino a 268501500 (0x100101FC)*



# ASSOLUTO

Il modo di indirizzamento assoluto, quando necessario (cioè per indirizzi grandi), si risolve con una composizione di istruzioni

Esempio:

**lw \$t3, 268501265**

Il MIPS traduce l'istruzione

**lw \$t3,0x10010111**

in

**lui \$at,0x1001**

**lw \$t3,0x111(\$at)**

**# sposta il valore nei 16bit più significativi di \$at**

**# l'indirizzo effettivo è dato dal contenuto del**

**# registro \$at incrementato dei sedici bit meno**

**# significati del valore espresso nella pseudo istruzione**



# ASSOLUTO

```
lb $t0, 0x00 00 00 91
```

10001101

\$t0

00000010



...  
10010000  
10010001  
10010010

...

**SIMBOLICO**





# SIMBOLICO

Il **modo di indirizzamento simbolico** (*Relocatable-symbol*)  
specifica un indirizzo tramite una etichetta

L'assemblatore crea il modulo oggetto con le informazioni di  
rilocazione: sostituisce l'etichetta con l'indirizzo dove risiede  
l'operando

ESEMPIO

**lw \$t0, operandA**



# SIMBOLICO

Il modo di indirizzamento simbolico, quando necessario, si risolve con una composizione di istruzioni

Esempio:

**lw \$t0, operandA**

NB:

Il MIPS traduce l'istruzione in

Se l'indirizzo di operandA è 0x1001345A

**lui \$at,0x1001**

**lw \$t0,0x345A(\$at)**

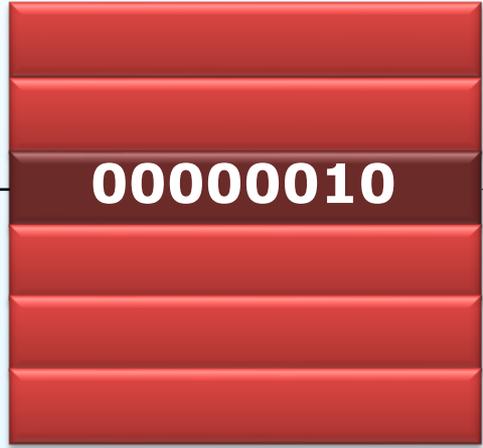


# SIMBOLICO

**lb \$t0, operandoA**

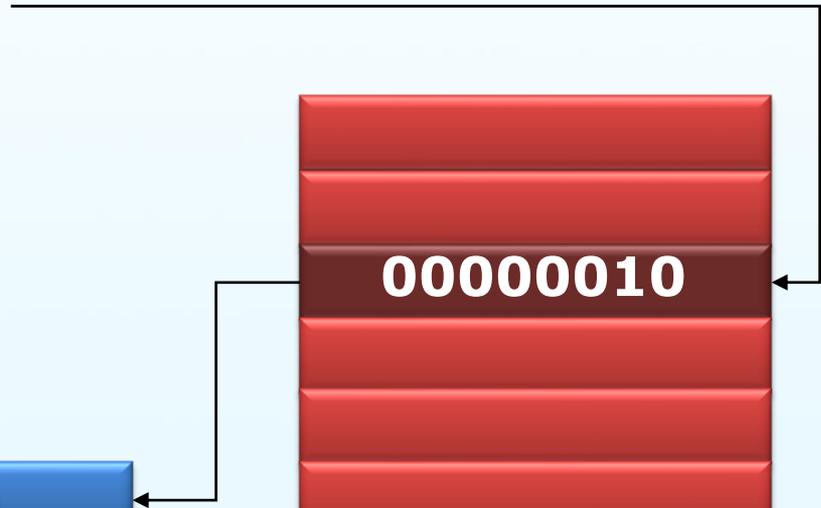
**operandoA=10010001**

**\$t0** **00000010**

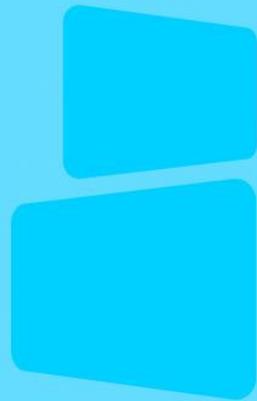


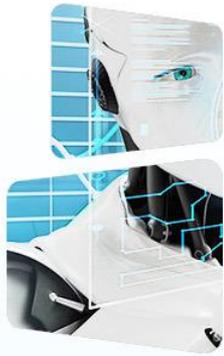
...

10010000  
10010001  
10010010  
10010011



# SIMBOLICO CON SPIAZZAMENTO





# SIMBOLICO CON SPIAZZAMENTO

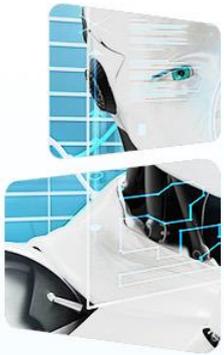
Il **modo di indirizzamento simbolico con spiazzamento** (*Relocatable-symbol ± expression*) specifica un indirizzo tramite una etichetta, mentre lo spiazzamento (offset) è un'espressione numerica

L'assemblatore risolve l'etichetta e somma il risultato dell'espressione

**ESEMPIO**

**lw \$t0, operandA+4**

*Si può anche utilizzare un valore negativo ma si deve sempre anteporre il segno +*



# SIMBOLICO CON SPIAZZAMENTO

Si tratta di una pseudo istruzione

ESEMPIO

**lw \$t0, operandA+4**

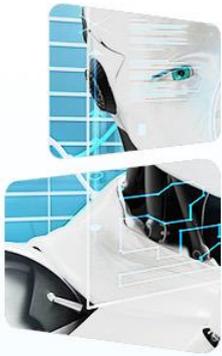
**.data**

**operandA: . Word 67 #si trova in 268500992 in esadecimale 0x10010000**

Diventa

**li \$at,0x1001**

**lw \$t0,4(\$at)**



# SIMBOLICO CON SPIAZZAMENTO

(Esempio)

**.text**

**.globl main**

**main:**

**lw \$t0,pippo**

**lw \$t1,pippo+4**

**lw \$t2,pluto**

**lw \$t3, paperino+-4**

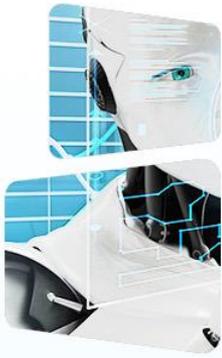
**# il contenuto di t1 e t3 è uguale al contenuto di t2**

**.data**

**pippo:.word 15 #si trova alla locazione 0x1001000**

**pluto:.word 256 #si trova alla locazione 0x1001004**

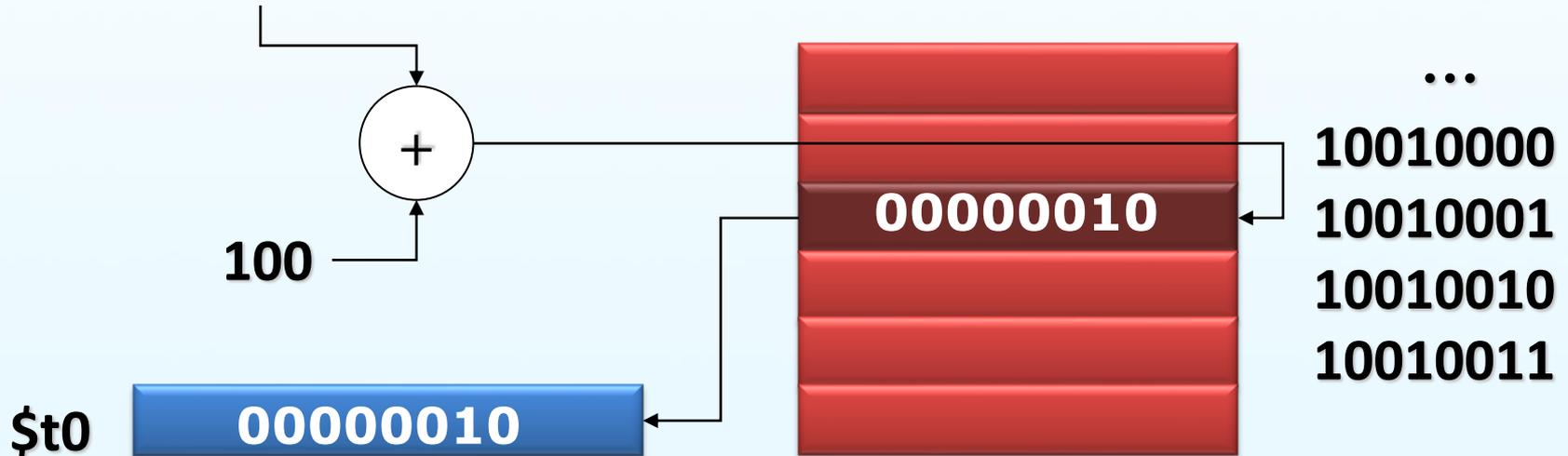
**paperino: .word 10 #si trova alla locazione 0x10010008**



# SIMBOLICO CON SPIAZZAMENTO

**lb \$t0, operandoB+4**

**operandoB=10001101**



**INDIRETTO A REGISTRO**





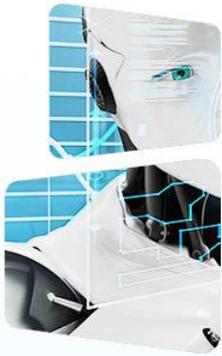
# INDIRETTO A REGISTRO

Il modo di indirizzamento indiretto a registro (*base register*) prevede che l'indirizzo in cui risiede l'operando sia contenuto in un registro

Il contenuto del registro, durante la fase di esecuzione, lo si invia come indirizzo alla Memoria Dati

ESEMPIO

**`lw $t0,($a0)`**



# INDIRETTO A REGISTRO

(Esempio)

```
.text
```

```
.globl main
```

```
main:
```

```
    lw $t2,pippo
```

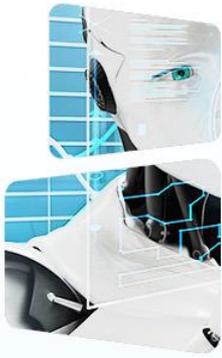
```
    la $t0,pippo
```

```
    lw $t1,($t0)
```

**#\$t1 e \$t2 hanno lo stesso valore**

```
.data
```

```
pippo: .word 256
```



# INDIRETTO A REGISTRO

**lb \$t0, (\$a0)**

**\$a0**

**000010010001**

**\$t0**

**00000000010**



...

10010000

10010001

10010010

10010011

...

# INDIRETTO A REGISTRO CON SPIAZZAMENTO NUMERICO





# INDIRETTO A REGISTRO CON SPIAZZAMENTO NUMERICO

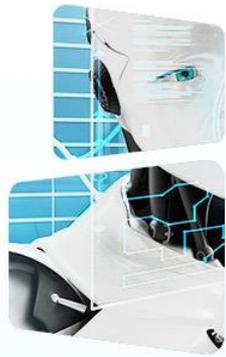
Il **modo di indirizzamento indiretto a registro con spiazzamento** (*expression base register*) specifica un indirizzo tramite la somma tra il contenuto del registro e uno spiazzamento (offset) descritto da una espressione

In fase di esecuzione si preleva il contenuto del registro e si incrementa (o decrementa) il valore considerando lo spiazzamento ottenendo così l'indirizzo effettivo

ESEMPIO

**`lb $t0,4($a0)`**

*NB: qualora lo spiazzamento superi i 16bit si ricorre ad uno sdoppiamento con LUI e ADDU (come per lo indirizzamento assoluto/immediato)*



# INDIRETTO A REGISTRO CON SPIAZZAMENTO NUMERICO

(Esempio)

**.text**

**.globl main**

**main:**

**la \$t0,pippo**

**lw \$t1,4(\$t0) #valore di pluto**

**.data**

**pippo:.word 15           #si trova alla locazione 0x10010000**

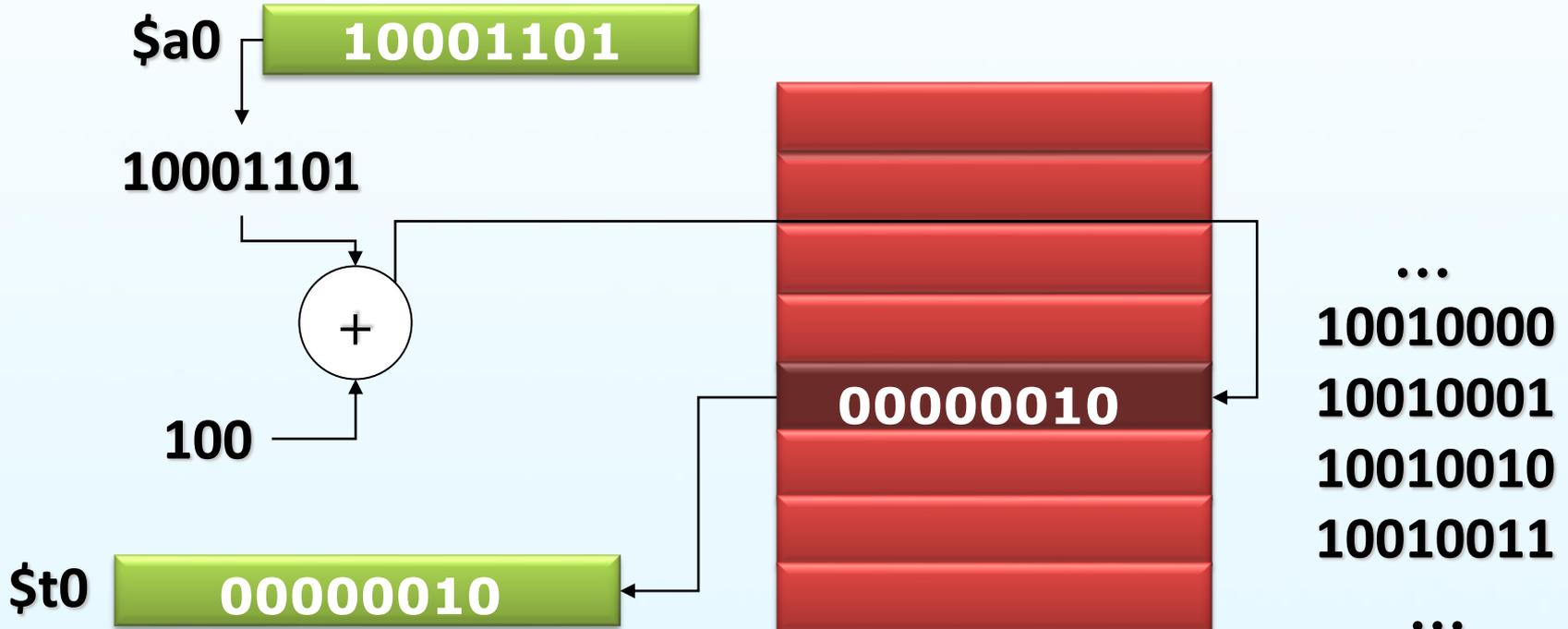
**pluto:.word 256       #si trova alla locazione 0x10010004**

**paperino: .word 10   #si trova alla locazione 0x10010008**



# INDIRETTO A REGISTRO CON SPIAZZAMENTO NUMERICO

**lb \$t0, 4(\$a0)**



# A REGISTRO CON SPIAZZAMENTO SIMBOLICO





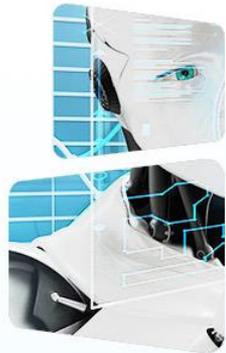
# INDIRETTO A REGISTRO CON SPIAZZAMENTO SIMBOLICO

Il **modo di indirizzamento indiretto a registro con spiazzamento simbolico** (*Relocatable-symbol(index register)*) specifica un indirizzo tramite la somma tra il contenuto del registro e uno spiazzamento (offset) descritto da una etichetta

L'assemblatore risolve l'etichetta. Durante la fase di esecuzione il valore dell'etichetta è sommato al contenuto del registro. L'indirizzo risultante lo si invia come indirizzo alla Memoria Dati

ESEMPIO

**lb \$t0, Vettore(\$a0)**



# INDIRETTO A REGISTRO CON SPIAZZAMENTO SIMBOLICO

(Esempio)

```
.text
```

```
.globl main
```

```
main:
```

```
    li $t0,4
```

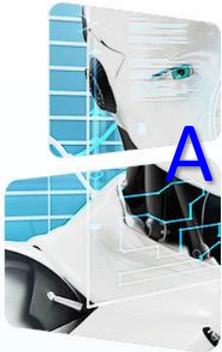
```
    lw $t1,pippo($t0) #valore di pluto
```

```
.data
```

```
pippo:.word 15          #si trova alla locazione 0x10010000
```

```
pluto:.word 256        #si trova alla locazione 0x10010004
```

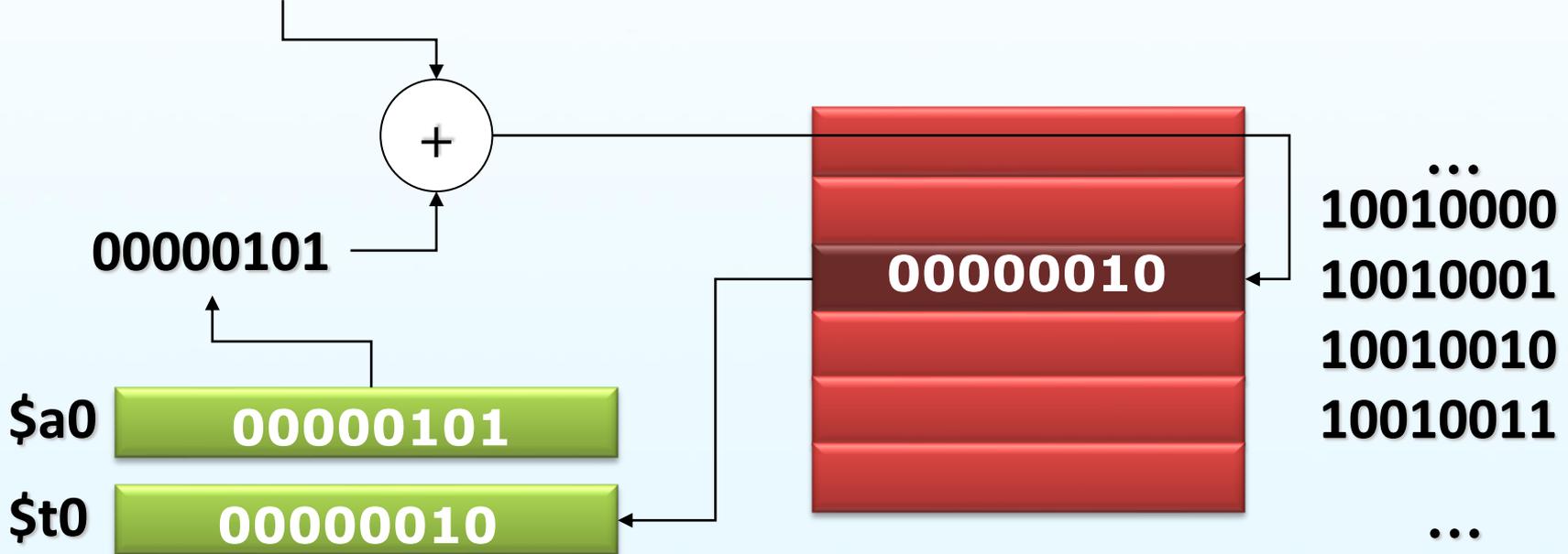
```
paperino: .word 10     #si trova alla locazione 0x10010008
```



# A REGISTRO CON SPIAZZAMENTO SIMBOLICO

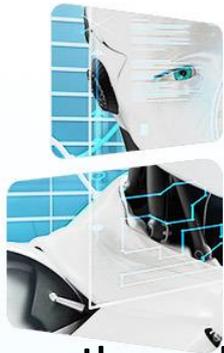
**lb \$t0, Vettore(\$a0)**

**Vettore=10001100**



# INDIRETTO A REGISTRO CON SPIAZZAMENTO SIMBOLICO E NUMERICO





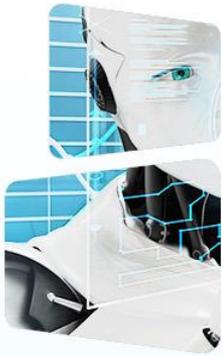
## INDIRETTO A REGISTRO CON SPIAZZAMENTO SIMBOLICO E NUMERICO

Il modo di indirizzamento a registro con spiazzamento simbolico e numerico (*Relocatable-symbol  $\pm$  expression(index register)*) specifica un indirizzo tramite la somma tra il contenuto del registro e uno spiazzamento (offset) descritto da una etichetta con un incremento o decremento individuato da una espressione aritmetica

L'assemblatore risolve l'etichetta. In fase di esecuzione il contenuto del registro si aggiunge (o si sottrae) al valore dell'etichetta risolta e allo spiazzamento e l'indirizzo risultante lo si invia come indirizzo alla Memoria Dati

ESEMPIO

**`lb $t0,Vettore+4($a0)`**



# INDIRETTO A REGISTRO CON SPIAZZAMENTO SIMBOLICO E NUMERICO

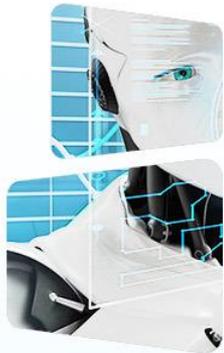
(Esempio)

```
.text
.globl main
main:

    xor $t7,$t7,$t7
    lw $t0,stud+0($t7)
    lh $t1,stud+4($t7)
    lb $t2,stud+6($t7)

li $v0,10
syscall

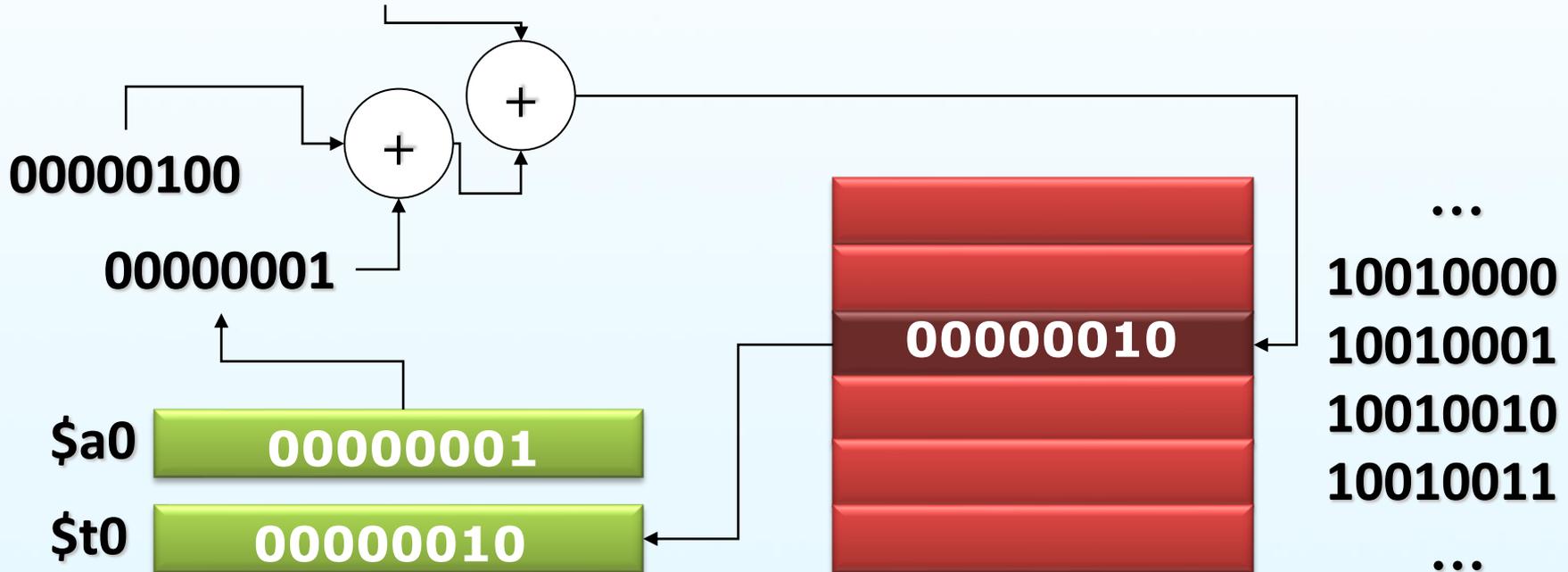
.data
stud:.word 11098231 #matricola
    .half 1974 #datanascita
    .byte 28 #mediavoti (interi)
```



# INDIRETTO A REGISTRO CON SPIAZZAMENTO SIMBOLICO E NUMERICO

**lb \$t0, Vettore+4(\$a0)**

**Vettore=10001100**



FINE

