

# Architettura degli Elaboratori Elettronici Esercitazioni

*Architettura MIPS*

**Franco Liberati**  
**[liberati@di.uniroma1.it](mailto:liberati@di.uniroma1.it)**

# MIPS

*Microprocessor without Interlocked  
Pipeline Stage*

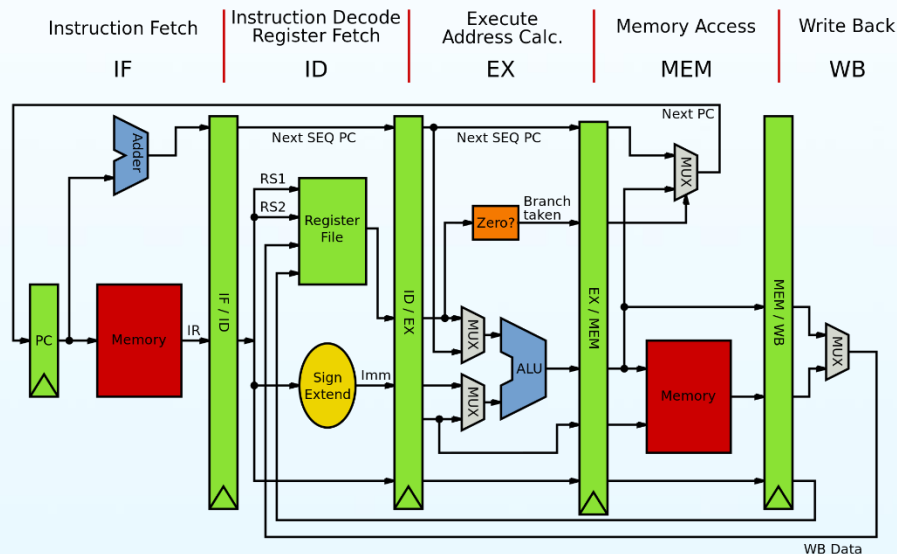




# Architettura MIPS

## Definizione

Il **MIPS** (acronimo di *microprocessor without interlocked pipeline stages*) è un'architettura informatica per microprocessori RISC sviluppata da MIPS Computer Systems Inc. (oggi MIPS Technologies Inc.)





# Architettura MIPS

Storia - L'inizio

- ❑ Progetto sviluppato nel 1981 da John L. Hennessy dell'Università di Stanford
- ❑ Suddivisione della **Memoria Centrale in due parti** fisiche (Memoria Istruzioni e Memoria Dati) per garantire l'esecuzione di una istruzione in un solo ciclo di clock
- ❑ Realizzazione di una architettura di tipo **RISC** e in grado di realizzare la tecnica della **canalizzazione (pipeline)**



# Architettura MIPS

*Storia - Finalità del progetto*

❑ **Istruzioni RISC** sono caratterizzate da:

❑ **semplicità** (nei primi modelli le moltiplicazioni e le divisioni tra interi furono realizzate con somme e sottrazioni successive)

❑ **pochi modi di indirizzamento** (per lo più **elementari** come **LOAD** e **STORE**) per garantire l'esecuzione di ciascuna istruzione in **un solo ciclo di clock**

❑ L'esecuzione in un solo ciclo di clock di una istruzione, la suddivisione delle varie unità funzionali e la loro sincronizzazione (ottenuta con l'interposizione di blocchi, un insieme di registri e linee di controllo che sorvegliano il completamento delle varie istruzioni) **consentono un pipeline regolare e quindi una prestazione della macchina più efficiente** in termine di quantità di calcoli in unità temporale



# Architettura MIPS

*Storia – La produzione industriale*

- ❑ Nel 1984 Hennessy fonda la MIPS Computer Systems
- ❑ Nel 1985 la società presenta il **processore R2000 con lunghezza della parola a 32bit** nel 1988 è commercializzato il modello **R3000** (riduzione della dimensione e del numero dei transistori; frequenza: 20Mhz, 33mhz a 35Mhz). Entrambi i processori sono impiegati come CPU delle workstation della società Silicon Graphics
- ❑ Nel 1991 MIPS presenta **R4000**, il suo **primo processore a 64 bit**
- ❑ Acquisizione della società da parte di Silicon Graphics (MIPS Technologies)





# Architettura MIPS

*Storia – Il successo industriale*

- ❑ Agli inizi degli anni Novanta, MIPS Technologies invade il mercato dei microprocessori grazie al basso prezzo e le ottime prestazioni di calcolo offerte dalla canalizzazione
- ❑ Nel 1997 il MIPS supera il numero di processori venduti da Motorola (uno dei leader del mercato mondiale)
- ❑ Nel 1999 MIPS annuncia la possibilità di acquistare la licenza per due processori base: **MIPS32** a 32 bit e **MIPS64** a 64 bit



# Architettura MIPS

Storia – *Lo sviluppo industriale*

- ❑ Nascita della SandCraft e sviluppo del processore **R7100** (eseguiva istruzioni *fuori ordine*)
- ❑ Creazione della SiByte e produzione del modello **SB-1250**, uno dei primi processori *systems-on-a-chip* (SOC) ad alte prestazioni basato su architettura MIPS
- ❑ Fondazione di Alchemy Semiconductor (in seguito acquisita da AMD), che produce il processore di tipo SOC dal nome **Au-1000** che necessita di un basso consumo energetico





# Architettura MIPS

*Storia – La situazione attuale*

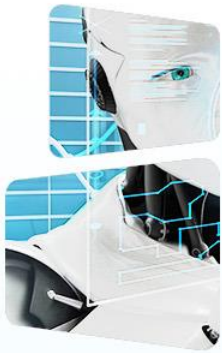
❑ Nella prima metà del XXII secolo l'architettura MIPS trova grossa diffusione nell'ambito dei sistemi embedded, dei device di Windows CE e nei router di Cisco e anche nelle console Nintendo 64, Sony PlayStation, PlayStation 2 e PlayStation Portable



# *MIPS*

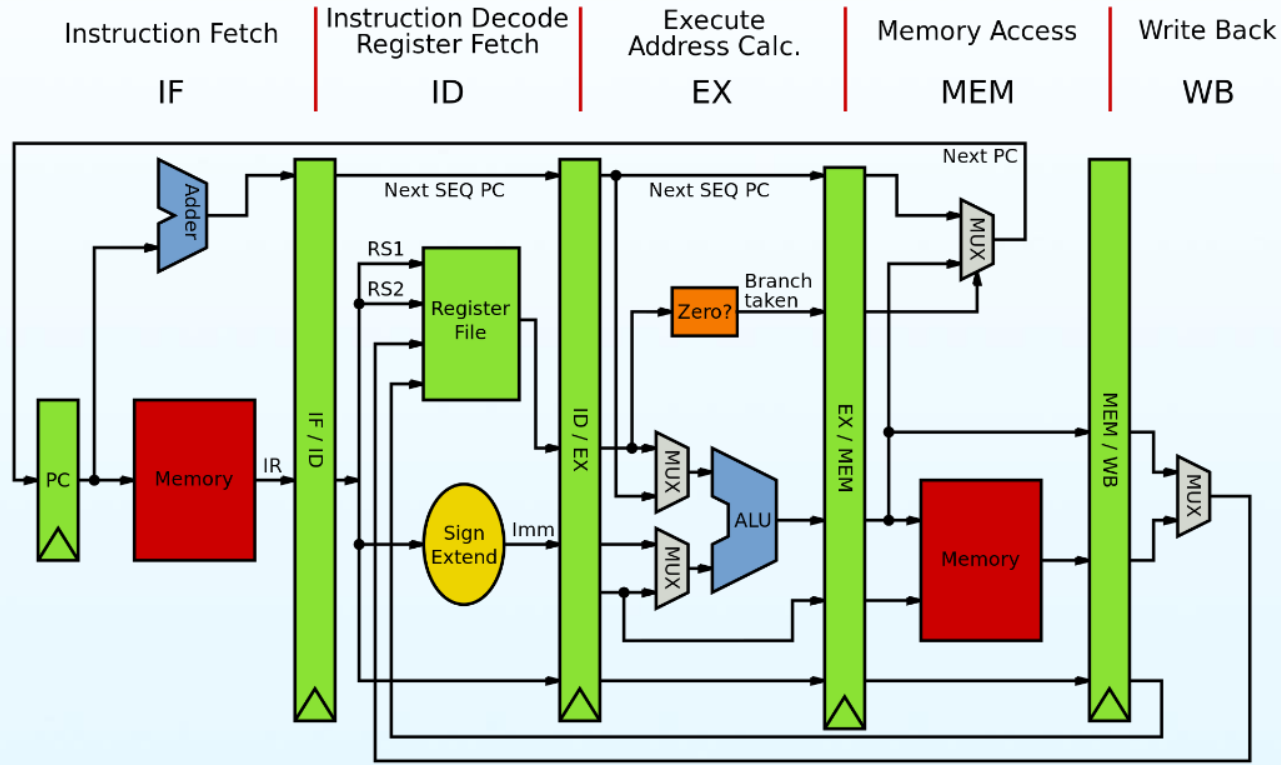
## *Elementi essenziali*





# Architettura MIPS

## Componenti – Schema generale

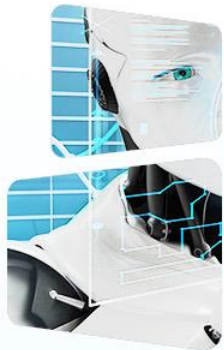




# Architettura MIPS

## Componenti – Registri

- ❑ I **registri** sono **locazioni di memoria** in cui ospitare temporaneamente degli **operandi/indirizzi/istruzioni**



# Architettura MIPS

## Componenti – Registri

❑ **REGISTRI TEMPORANEI NON PRESERVANTI:** si azzerano dopo un salto a sub-routine

\$t0	\$t1	\$t2	\$t3	\$t4
\$t5	\$t6	\$t7	\$t8	\$t9

❑ **REGISTRI TEMPORANEI PRESERVANTI:** mantengono sempre i valori

\$s0	\$s1	\$s2	\$s3	\$s4
\$s5	\$s6	\$s7	\$s8	\$s9



# Architettura MIPS

## Componenti – Registri

### ❑ REGISTRI TEMPORANEI PER LE SUB ROUTINE

\$a0

\$a1

\$a2

\$a3

Parametri di **ingresso**  
della sub-routine

\$v0

\$v1

**Risultati** della sub-routine

### ❑ REGISTRI PER I NUMERI REALI (NUMERI IN VIRGOLA MOBILE, *floating point*)

\$fp0

-----

\$fp31



# Architettura MIPS

## Componenti – ALU

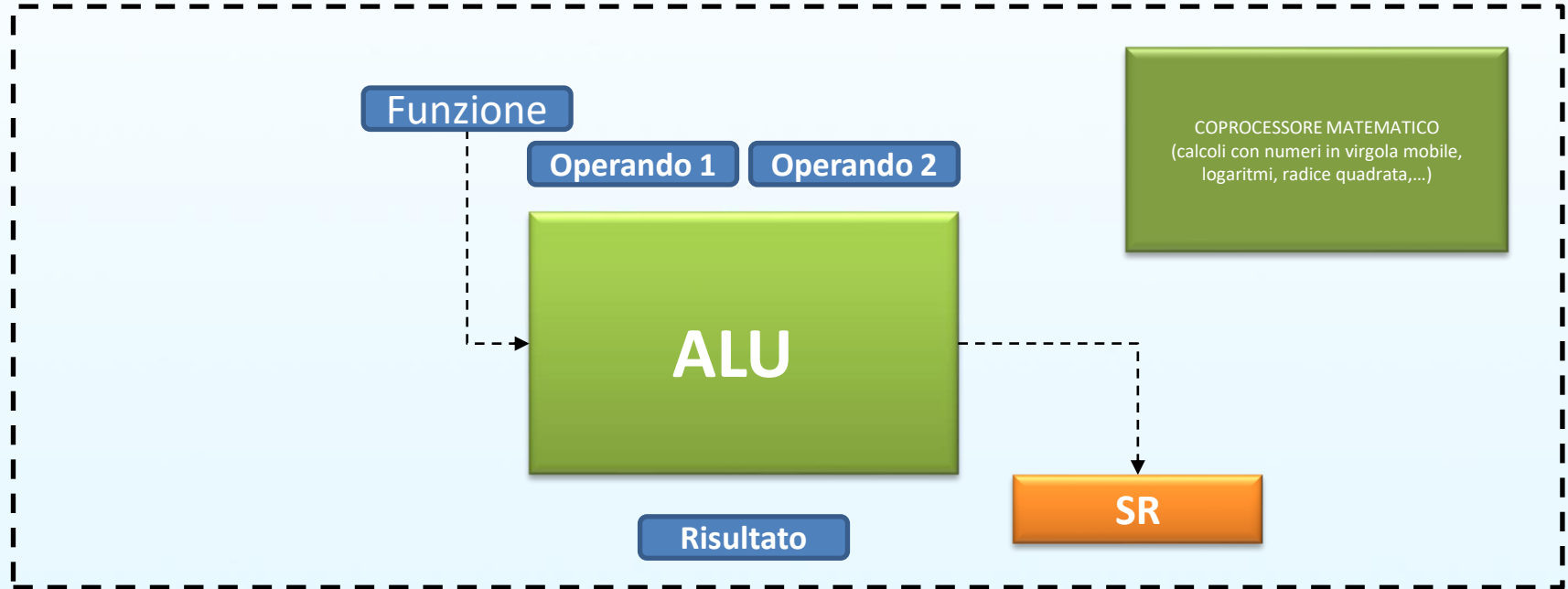
- ❑ La **ALU** è il **modulo** nel quale è presente la circuiteria utile per svolgere **operazioni logiche – aritmetiche** (*adder, shifter, comparator,...*)
- ❑ Il MIPS è dotato di un **coprocessore matematico**, visto come una **unità di I/O** con un set di istruzioni specifico, utile per svolgere **operazioni aritmetiche con operandi in virgola mobile** (espressi in singola e doppia precisione)
  - ❑ Questa caratteristica è presente anche nel simulatore MARS





# Architettura MIPS

## Componenti – ALU e Coprocessore Matematico





# Architettura MIPS

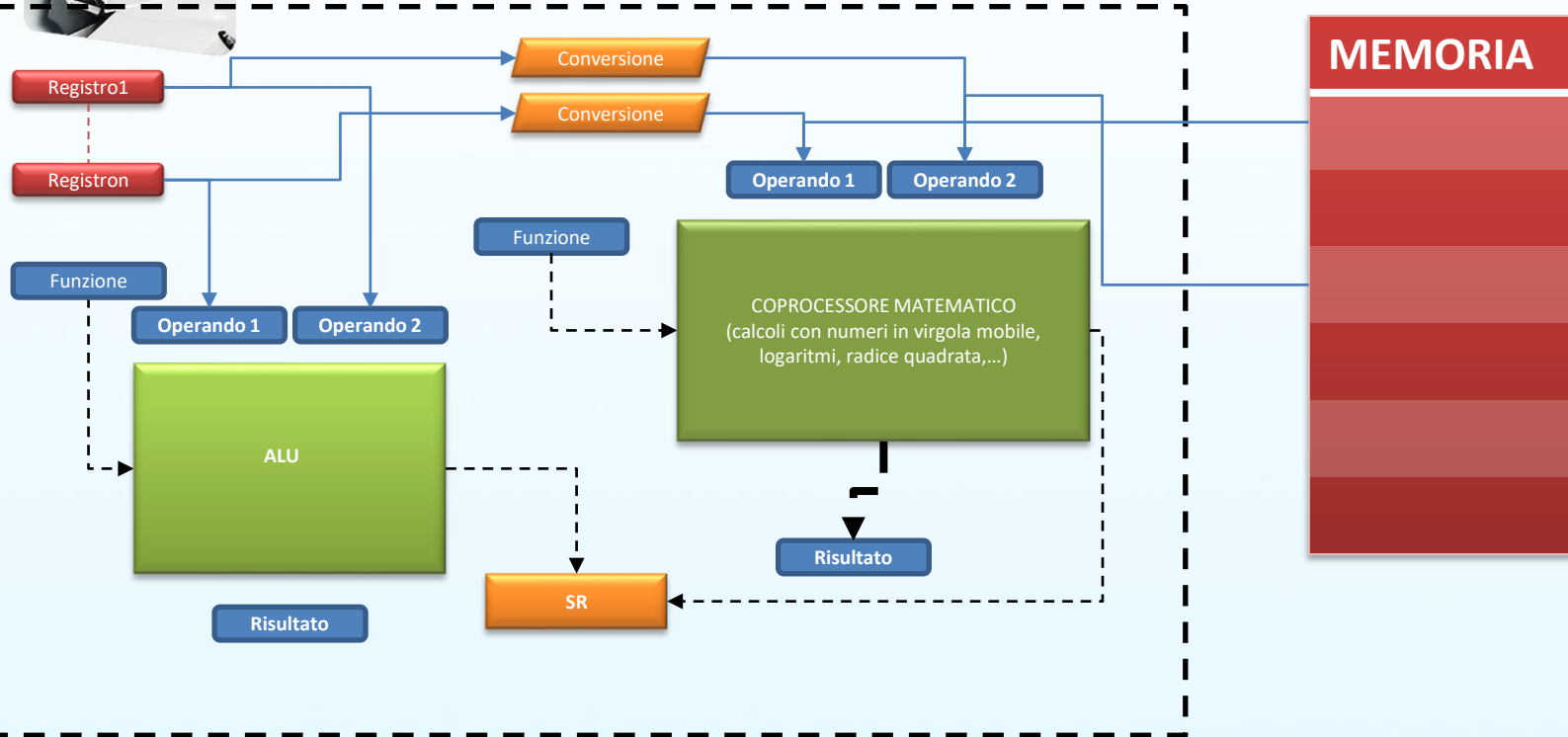
## Componenti – Coprocessore Matematico

- ❑ Il coprocessore matematico opera su numeri reali - rappresentati in Virgola Mobile Singola Precisione e Doppia Precisione - siti in memoria oppure numeri interi derivati dal calcolo dell'ALU e stipati nei registri (o in memoria) previa **conversione nel formato IEEE754** mediante apposite funzioni di trasformazione



# Architettura MIPS

## Componenti – ALU e Coprocessore Matematico

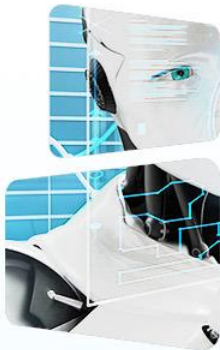




# Architettura MIPS

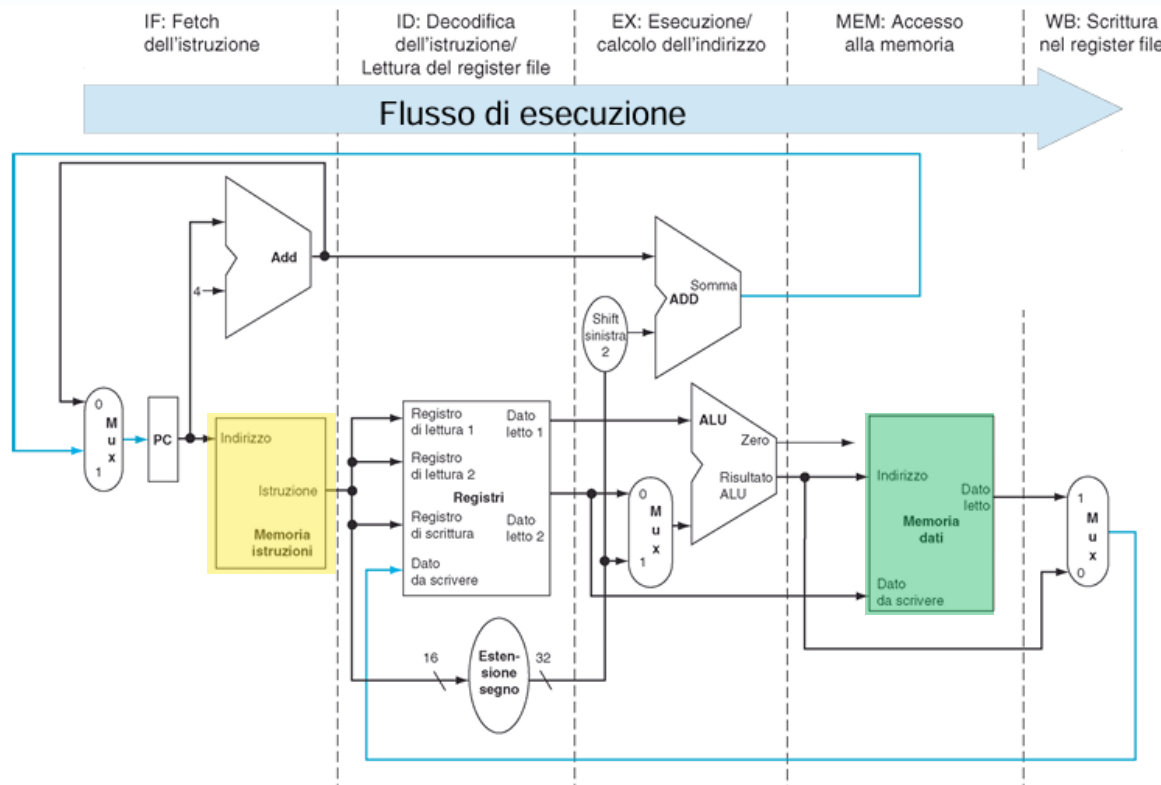
## Componenti – Memoria

- ❑ Il MIPS ha due memorie distinte:
  - ❑ **Memoria Istruzioni:** area nella quale sono stipate le istruzioni afferenti ad un programma, e una serie di informazioni ausiliarie (stack, kernel S.O.,...)
  - ❑ **Memoria Dati:** area nella quale sono stipati i dati utilizzati dal programma



# Architettura MIPS

## Componenti – Memoria





# Architettura MIPS

## Componenti – Memoria: esempio

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000043
\$t1	9	0x00000064
\$t2	10	0x10010043
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$a0	16	0x00000000
\$a1	17	0x00000000
\$a2	18	0x00000000
\$a3	19	0x00000000
\$a4	20	0x00000000
\$a5	21	0x00000000
\$a6	22	0x00000000
\$a7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$a0	26	0x00000000
\$t1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffcfc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x0040001c
hi		0x00000000
lo		0x00000000

Text Segment					
Bkpt	Address	Code	Basic		
<input type="checkbox"/>	0x00400000	0x3c011001	lui \$1,0x00001001	2:	lw \$t0,x
<input type="checkbox"/>	0x00400004	0x8c280000	lw \$8,0x00000000(\$1)		
<input type="checkbox"/>	0x00400008	0x3c011001	lui \$1,0x00001001	3:	lw \$t1,y
<input type="checkbox"/>	0x0040000c	0x8c290004	lw \$9,0x00000004(\$1)		
<input type="checkbox"/>	0x00400010	0x01015020	add \$10,\$8,\$1	4:	add \$t2,\$t0,\$1
<input type="checkbox"/>	0x00400014	0x3c011001	lui \$1,0x00001001	5:	sw \$t2,z
<input type="checkbox"/>	0x00400018	0xac2a0008	sw \$10,0x00000008(\$1)		

Data Segment									
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)	
0x10010000	0x00000043	0x00000064	0x10010043	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	

.text

```
lw $t0,x
lw $t1,y
add $t2,$t0,$1
sw $t2,z
```

.data

```
x: .word 67
y: .word 100
z: .word 0
```



# Architettura MIPS

## Componenti – I/O

- ❑ I **dispositivi di Ingresso e di Uscita** (Input Output device o I/O device) permettono l'interazione con il mondo esterno
- ❑ Il MIPS interagisce con molte periferiche (scheda audio, video, terminale video, tastiera)



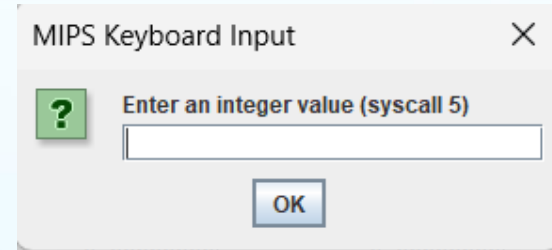


# Architettura MIPS

## Componenti – I/O

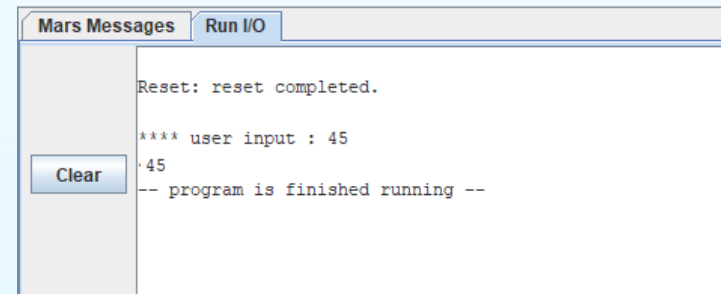
Finestra di input per l'immissione dei dati da tastiera

```
li $v0,5      # servizio di lettura di un intero da tastiera
syscall       #chiamata di sistema
move $t0,$v0  #spostamento del valore letto da tastiera
              #residente in $v0 dopo la syscall
```



Finestra di output (mostrata dal simulatore)  
per la visualizzazione dei risultati

```
move $a0,$t0  #spostamento del valore intero da stampare
              #da $t0 a $a0
li $v0,1      #servizio di stampa di un intero
syscall       #chiamata di sistema
```





# Architettura MIPS

## Esempio 1 (Interazione con i dispositivi di I/O)

```
.text
.globl main
main:
    li $v0,5      # servizio di lettura di un intero da tastiera
    syscall       #chiamata di sistema
    move $t0,$v0  #spostamento del valore letto da tastiera residente in $v0 dopo la syscall
    add $t0,$t0,1  #calcolo del valore successivo
    move $a0,$t0   #spostamento del valore intero da stampare da $t0 a $a0
    li $v0,1      #servizio di stampa di un intero
    syscall       #chiamata di sistemali

$v0,10
syscall
```

Fine

