



SAPIENZA
UNIVERSITÀ DI ROMA

COPROCESSORE MATEMATICO

Dott. Franco Liberati

Argomenti

01

Coprocessore Matematico

02

Istruzioni per i numeri in virgola
mobile

The image features a central, glowing blue microchip with a square shape and a grid-like pattern on its surface. It is set against a dark blue background that resembles a circuit board. Numerous glowing blue lines, representing circuit traces, extend from the chip and branch out across the frame. Some of these lines end in small, bright blue dots. The overall aesthetic is high-tech and futuristic, with a strong emphasis on blue light and geometric patterns.

Coprocessore Matematico

Coproprocessore Matematico

Registri

Il MIPS è dotato di un **coprocessore matematico** deputato a svolgere operazioni tra numeri reali rappresentati in virgola mobile a singola precisione (*float*) e doppia precisione (*double*) secondo lo standard IEEE 754

The screenshot shows the MARS 4.5 MIPS simulator interface. The main window is titled "MARS 4.5" and has a menu bar with "File", "Edit", "Run", "Settings", "Tools", and "Help". Below the menu bar is a toolbar with various icons. The main area is divided into two panes: "Edit" and "Execute". The "Execute" pane is currently active, showing a large blue area for the program execution. At the bottom of the window, there is a "Mars Messages" pane with a "Run I/O" button and a "Clear" button. The message area displays the text "-- program is finished running --".

On the right side of the window, there is a "Registers" panel. It contains a table for the Coprocessor 1 and Coprocessor 0 registers. The table has three columns: "Name", "Float", and "Double". The "Name" column lists registers from \$f0 to \$f31. The "Float" and "Double" columns show the values of the registers. All values are currently 0.0.

Name	Float	Double
\$f0	0.0	0.0
\$f1	0.0	0.0
\$f2	0.0	0.0
\$f3	0.0	0.0
\$f4	0.0	0.0
\$f5	0.0	0.0
\$f6	0.0	0.0
\$f7	0.0	0.0
\$f8	0.0	0.0
\$f9	0.0	0.0
\$f10	0.0	0.0
\$f11	0.0	0.0
\$f12	0.0	0.0
\$f13	0.0	0.0
\$f14	0.0	0.0
\$f15	0.0	0.0
\$f16	0.0	0.0
\$f17	0.0	0.0
\$f18	0.0	0.0
\$f19	0.0	0.0
\$f20	0.0	0.0
\$f21	0.0	0.0
\$f22	0.0	0.0
\$f23	0.0	0.0
\$f24	0.0	0.0
\$f25	0.0	0.0
\$f26	0.0	0.0
\$f27	0.0	0.0
\$f28	0.0	0.0
\$f29	0.0	0.0
\$f30	0.0	0.0
\$f31	0.0	0.0

Below the register table, there is a "Condition Flags" section with checkboxes for flags 0 through 7. All flags are currently unchecked.

Coprocessore Matematico

Il coprocessore matematico del MIPS è dotato di 32 registri, identificati come **\$f0...\$f31**, che sono usati rispettando alcune convenzioni

Registri del coprocessore matematico del MIPS	
Numero del registro	Uso
\$f0 - \$f3	Registri di scambio con la CPU
\$f4 - \$f10	Registri temporanei per numeri reali
\$f12 - \$f14	Argomenti per subroutine
\$f16 - \$f18	Registri temporanei per numeri reali (non preservanti)
\$f20 - \$f31	Registri temporanei per numeri reali (preservanti)



Coprocessore Matematico



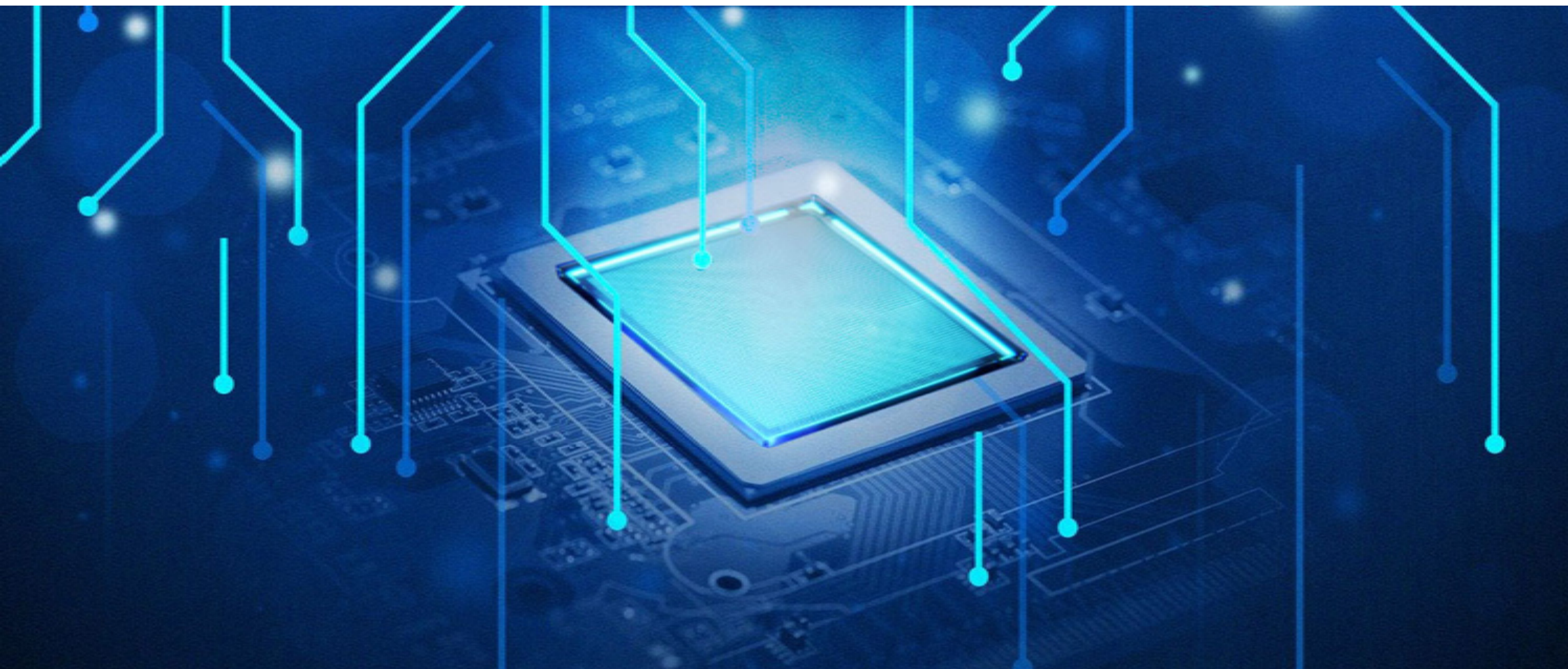
Definizione di operandi IEE754

La **definizione** di un numero reale in singola precisione in memoria si ottiene con la direttiva **.float**, o in doppia precisione **.double**, seguita dal valore espresso con un punto che divide la parte intera da quella decimale

Esempio

ValFloat: .float 3.5

ValDouble: .double -309380927329.7683



Istruzioni per i numeri in virgola mobile

Coprocessore Matematico ISA

Spostamento: Coprocessore Matematico ↔ Memoria

Le istruzioni di **spostamento** riguardano gli operandi contenuti nei registri all'interno del coprocessore matematico, in quelli residenti nell'Unità di Elaborazione e, infine, nelle locazioni della Memoria dei Dati.

lwc1 rfd,<label>	Preleva 32bit da una locazione di memoria <label> in un registro floating point rfd=<imm>
swc1 rfs,<label>	Archivia un numero reale di un registro floating point in memoria MEM=rfs
l.d rfd, <label>	Preleva 64bit da una locazione di memoria <label> in un registro floating point rfd=<label>
l.s rfd, <label>	Preleva 32bit da una locazione di memoria <label> in un registro floating point rfd=<label>
s.d rfs, <label>	Archivia il numero reale in doppia precisione sito in due registri floating point (rfs e rfs+1) in memoria MEM=rfs.rs1
s.s rfs, <label>	Archivia il numero reale in singola precisione sito in un registro floating point (rfs) in memoria MEM=rfs

Coprocessore Matematico ISA

Spostamento: Coprocessore Matematico ↔ Memoria (esempio)

Somma di due numeri reali
rappresentati in singola
precisione

```
.text
.globl main

main:
    lwc1 $f1, val_x    #prelievo primo operando singola precisione
                        #(analogo l.s $f1, val_x)
    lwc1 $f2, val_y    #prelievo secondo operando singola precisione
                        #(analogo l.s $f2, val_y)

    add.s $f0, $f1, $f2 #somma tra operandi in singola precisione

    li $v0, 10          #terminazione programma
    syscall

.data
val_x: .float 3.5
val_y: .float -23.7
```

Coprocessore Matematico ISA

Spostamento: Coprocessore Matematico ↔ Memoria (esempio)

l.d \$f2, reale64bit

Nei registri \$f2 e \$f3 è presente il valore in doppia precisione contenuto negli 8 byte contigui in memoria a partire dalla locazione di memoria rappresentato dall'etichetta *reale64bit*

s.d \$f4, reale64bit

Negli 8 byte contigui a partire dall'indirizzo di memoria rappresentato dall'etichetta *reale64bit* è presente il valore contenuto nei registri \$f4 e \$f5

Coprocessore Matematico ISA

Spostamento: Coprocessore Matematico ↔ Coprocessore matematico

mov.s rfd,rfs	Sposta il contenuto tra registri (singola precisione) rfd=rfs
mov.d rfd,rfs	Sposta il contenuto tra registri (doppia precisione) rfd=rfs

```
.text
.globl main

main:
    lwc1 $f1,x      #prelievo primo operando singola precisione
    lwc1 $f2,y      #prelievo secondo operando singola precisione

    add.s $f0,$f1,$f2 #somma tra operandi in singola precisione
    mov.s $f12,$f0   #pone il risultato in $f12
    li $v0,2         #richiede il servizio di stampa su videoterminale di stampa di un float
    syscall          #attivazione del servizio

    li $v0,10        #terminazione programma
    syscall

.data
x: .float 3.5
y: .float -23.7
```


Coprocessore Matematico ISA

Spostamento: Coprocessore Matematico ↔ CPU

mfcz rd,rfs	Sposta il contenuto del registro rfs del coprocessore z nel registro destinazione rd della CU
mfcz.d rd,rfs	<i>Se si sposta un double il contenuto al termine dell'operazione si trova in rd e rd+1</i>

ESEMPIO MARS

mfc1 \$t0,\$f0

#Sposta il valore float da \$f0 a \$t0

mfc1.d \$t0,\$f0

#Sposta il valore double di \$f0 e \$f1 in \$t0 e \$r1

mtcz rs, rfd	Sposta il contenuto del registro rs della CU nel registro destinazione rfd del coprocessore z
mtcz.d rs, rfd	Sposta il contenuto del registro rs della CU nel registro destinazione rfd del coprocessore z

ESEMPIO MARS

mtc1 \$t0,\$f0

#Sposta il valore da \$t0 a \$f0

Coprocessore Matematico ISA

Conversione

Lo spostamento tra registri con mfcz e mtcz avviene come una copia speculare. In altre parole trasferendo il contenuto da un registro ad uso generale a quello all'interno del coprocessore matematico (e viceversa) non c'è una conversione del valore da intero alla sua rappresentazione in virgola mobile (o da virgola mobile ad intero). La rappresentazione dal formato intero a standard IEEE 754 (o viceversa) avviene applicando, dopo lo spostamento, una istruzione di conversione

cvt.s.w rfd,rfs	Converte da intero (a 32bit) a numero in singola precisione rfd=(float) rfs
cvt.d.w rfd,rfs	Converte da intero (a 32bit) a numero in doppia precisione rfd=(double) rfs
cvt.w.s rfd,rfs	Converte (troncamento/approssimazione) da numero in singola precisione a intero rfd=(int) rfs
cvt.s.d rfd,rfs	Converte da numero in doppia precisione a singola precisione rfd=(float) rfs
cvt.d.s rfd,rfs	Converte da singola precisione a doppia precisione rfd=(double) rfs

Coprocessore Matematico ISA

Conversione (esempio)

Coprocessore Matematico ISA

Conversione (esempio)

.text	#Direttiva del Segmento Testo
.globl main	#Direttiva per indicare l'etichetta main come globale
main:	#Etichetta <i>main</i> : inizio del programma
#CONVERSIONE DA INTERO A VIRGOLA MOBILE	
lw \$t0,intvar	#Lettura dell'operando intero sito in memoria (\$t0=5)
mtc1 \$t0,\$f0	#Copia dal registro \$t0 in \$f0 (in \$f0 c'è la stringa binaria #intvar cioè \$f0=0000000000000000000000000000101 ovvero \$f0=7.E ⁻⁴⁵)
cvt.s.w \$f0,\$f0	#\$f0 ← (float)\$f0 cioè \$f0=5.0
li \$v0,10	
syscall	
.data	
intvar : .word 5	#Definizione di un numero intero a 32bit

Coprocessore Matematico ISA

Istruzioni aritmetiche

Le **istruzioni logiche-aritmetiche** avvengono sempre impiegando registri del coprocessore matematico. Oltre alle operazioni note ci sono funzioni specifiche per i numeri reali: il MIPS ha la radice quadrata definita dall'istruzione sqrt

SINGOLA PRECISIONE	DOPPIA PRECISIONE	SIGNIFICATO
abs.s rd,rs	abs.d rd,rs	$rd = rs $
add.s rd, rs, rt	add.d rd, rs, rt	$rd = rs + rt$
sub.s rd, rs, rt	sub.d rd, rs, rt	$rd = rs - rt$
div.s rd,rs,rt	div.d rd,rs,rt	$rd = rs / rt$
mul.s rd,rs,rt	mul.d rd,rs,rt	$rd = rs * rt$
sqrt.s rd,rs	sqrt.d rd,rs	$rd = \sqrt{rs}$

Coprocessore Matematico ISA

FLAG

Il coprocessore matematico del MIPS è dotato di otto flag utili per svolgere i salti condizionali

I flag sono enumerati <imm> da 0 a 7

Registers	Coproc 1	Coproc 0
Name	Float	Double
\$f0	0.0	0.0
\$f1	0.0	
\$f2	0.0	0.0
\$f3	0.0	
\$f4	0.0	0.0
\$f5	0.0	
\$f6	0.0	0.0
\$f7	0.0	
\$f8	0.0	0.0
\$f9	0.0	
\$f10	0.0	0.0
\$f11	0.0	
\$f12	0.0	0.0
\$f13	0.0	
\$f14	0.0	0.0
\$f15	0.0	
\$f16	0.0	0.0
\$f17	0.0	
\$f18	0.0	0.0
\$f19	0.0	
\$f20	0.0	0.0
\$f21	0.0	
\$f22	0.0	0.0
\$f23	0.0	
\$f24	0.0	0.0
\$f25	0.0	
\$f26	0.0	0.0
\$f27	0.0	
\$f28	0.0	0.0
\$f29	0.0	
\$f30	0.0	0.0
\$f31	0.0	

Condition Flags			
<input type="checkbox"/> 0	<input type="checkbox"/> 1	<input type="checkbox"/> 2	<input type="checkbox"/> 3
<input type="checkbox"/> 4	<input type="checkbox"/> 5	<input type="checkbox"/> 6	<input type="checkbox"/> 7

Coprocessore Matematico ISA

Istruzioni logiche

Le **istruzioni logiche-aritmetiche** relativa alla **comparazione** (*compare*) analizzano delle condizioni tra due operandi e settano dei bit (denominato flag) a 1 se la condizione è verificata; altrimenti lo setta a 0

SINGOLA PRECISIONE	DOPPIA PRECISIONE	SIGNIFICATO
c.eq.s r1,r2	c.eq.d r1,r2	Se $r1=r2$ setta il flag0 a vero (pone un check); se $r1 \neq r2$ il flag0 è falso (toglie il check)
c.eq.s <imm>,r1,r2	c.eq.d <imm>,r1,r2	Se $r1=r2$ setta il flag <i>imm</i> a vero (pone un check); se $r1 \neq r2$ flag <i>imm</i> è falso (toglie il check)
c.le.s r1,r2	c.le.d r1,r2	Se $r1 \leq r2$ setta il flag 0 a vero (pone un check); altrimenti il flag0 è falso (toglie il check)
c.lt.s r1,r2	c.lt.d r1,r2	Se $r1 < r2$ setta il flag 0 a vero (pone un check); altrimenti flag0 a falso (toglie il check)

Coprocessore Matematico ISA

SALTO CONDIZIONATO

Il coprocessore matematico del MIPS sfrutta l'istruzione BC1F e il controllo dei flag per effettuare un salto condizionato

bc1f imm, label	Salta a <i>label</i> se il codice di condizione indicato dal flag <i>imm</i> non ha il check (cioè si trova nella condizione falso)
bc1f label	Salta a <i>label</i> se il codice di condizione indicato dal flag0 non ha il check (cioè si trova nella condizione falso)

Coprocessore Matematico ISA

(esempio AREA CERCIO)

	.text	#Direttiva del Segmento Testo
	.globl main	#Direttiva per indicare l'etichetta main come globale
main:		#Etichetta <i>main</i> : inizio del programma
	l.s \$f5,zero	#Imposta il valore per il controllo del raggio (\$f5=0.0)
	li \$v0,6	#Impostazione del servizio di lettura di un valore reale singola precisione
	syscall	#Attivazione del servizio (in \$f0 è presente il valore reale letto da tastiera)
	c.lt.s \$f0, \$f5	#Setta a 1 il flag0 se il raggio è minore di zero
	bc1t raggio_negativo	#Salta se il flag0 è settato ad 1
	l.s \$f1,pigreco	#Lettura di π
	mul.s \$f2,\$f0,\$f0	#Calcolo raggio ² e copia in \$f4
	mul.s \$f2,\$f2, \$f1	#Calcolo $\pi \cdot \text{raggio}^2$ e copia in \$f4
	li \$v0,2	#Richiesta servizio stampa di un numero reale in singola precisione
	mov.s \$f12,\$f2	#Copia del valore reale in singola precisione da stampare
	syscall	#Attivazione del servizio
	j fine	
raggio_negativo:		
	li \$v0,4	#Richiesta del servizio di stampa di una stringa
	la \$a0,MsgErr	#Impostazione stringa da stampare
	syscall	#Attivazione del servizio
fine:	li \$v0,10	#Richiesta servizio di interruzione del programma
	syscall	#Attivazione del servizio

.data

pigreco: .float 3.141592

zero: .float 0.0

MsgErr:.asciiz "Errore: Raggio negativo!"

The background is a dark blue gradient with intricate white and yellow circuit board traces and components. The traces are thin lines that branch out and connect various components. The components are small, rectangular and circular shapes, some of which are highlighted in yellow. The overall design is technical and futuristic.

FINE