

## Linguaggio macchina e assembler

### 3 tipi di istruzioni macchina

- 1) **trasferimento** tra **RAM** e **registri di calcolo** della CPU
- 2) **operazioni aritmetiche**: somma, differenza, moltiplicazione e divisione
- 3) **operazioni di controllo**: confronto, salto e stop

### Istruzioni per **trasferimento** dati

- In realta', non e' un trasferimento ma una **copia**
- **Load**: da **memoria** a registro
- **Store**: da **registro** a memoria
  
- Anche trasferimento tra la **memoria RAM** e le unita' **input/output**

### Istruzioni **logico/aritmetiche**

- **Operazioni aritmetiche**: **somma**, ...
- **Operazioni logiche**: **and**, **or**, **xor**, anche **shift** e **rotate**

## Istruzioni di controllo

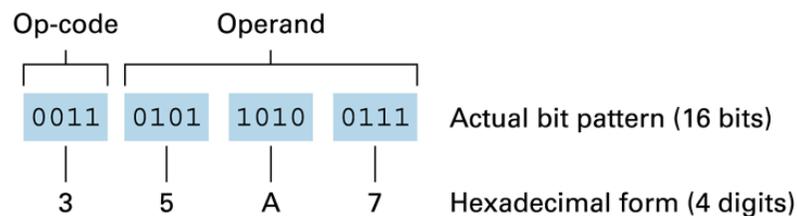
- Regolano l'esecuzione del programma
- Es.: **stop**
- Anche istruzioni di **salto**: se l'istruzione da eseguire non e' la successiva nella lista
- **Salto condizionato** o no
  - Es.: salta al passo 5, o salta al passo 5 se il valore ottenuto e' 0

## Divisione di due valori in memoria

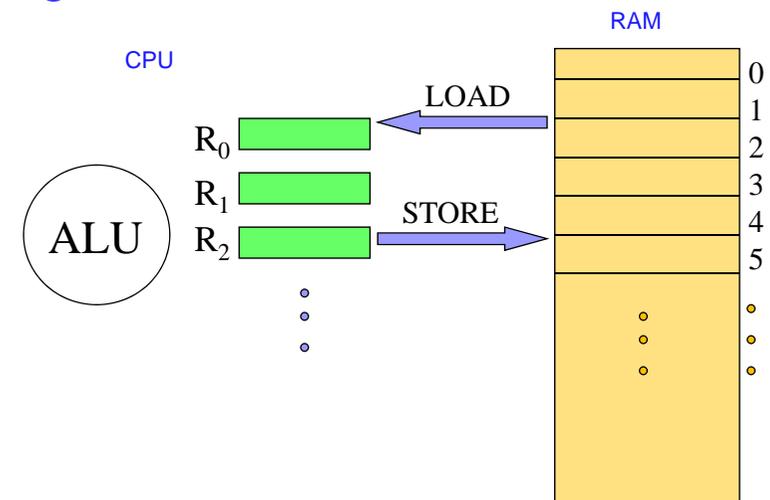
1. **Carica** in un registro un valore in memoria (**LOAD**)
2. **Carica** in un altro registro un altro valore in memoria (**LOAD**)
3. **Se** questo secondo valore e' 0, **salta** al passo 6 (**salto condizionato**)
4. **Dividi** il contenuto del primo registro per quello del secondo registro e metti il risultato in un terzo registro (**op. aritmetica**)
5. **Archivia** il contenuto del terzo registro in memoria (**STORE**)
6. **STOP**

## Istruzione macchina

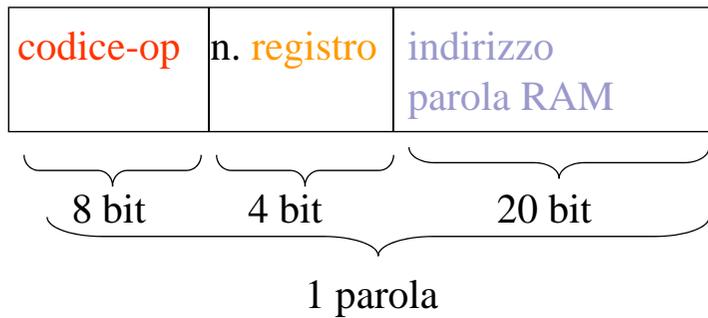
- **Due parti (campi):**
  - Campo **codice operativo**: quale operazione eseguire
  - Campo **operando**: diverso a seconda dell'operazione



## Istruzioni di trasferimento: registri $\leftrightarrow$ RAM



## Istruzioni di trasferimento in linguaggio macchina in binario!



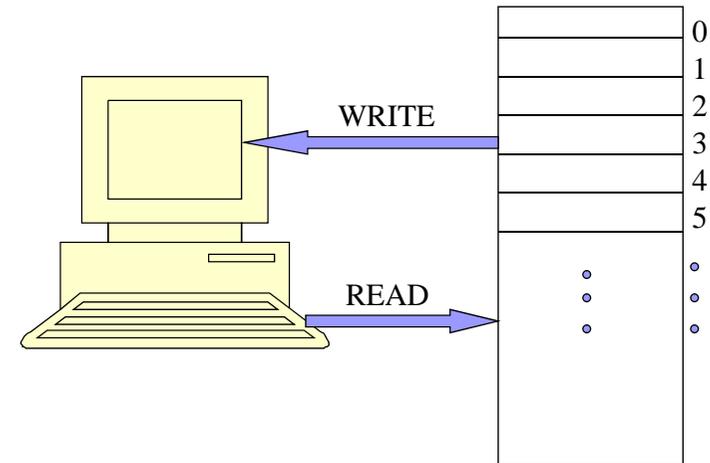
Codici:

LOAD	00000000
STORE	00000001

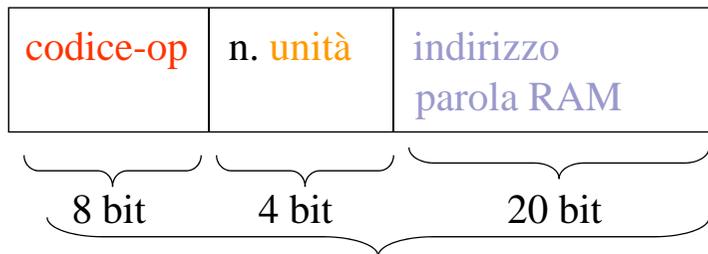
Esempio:

00000000	0000	0000000000000000000010
00000001	0010	00000000000000000000101

## Istruzioni di input/output: unità I/O $\Leftrightarrow$ RAM



## Istruzioni input e output in linguaggio macchina



Codici:

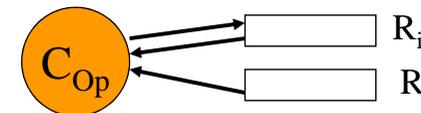
READ	00010000	Unità:	INP	0000 (tastiera)
WRITE	00010001		OUT	0001 (video)

Esempio:

00010000	0000	0000000000000000000010
00010001	0001	00000000000000000000101

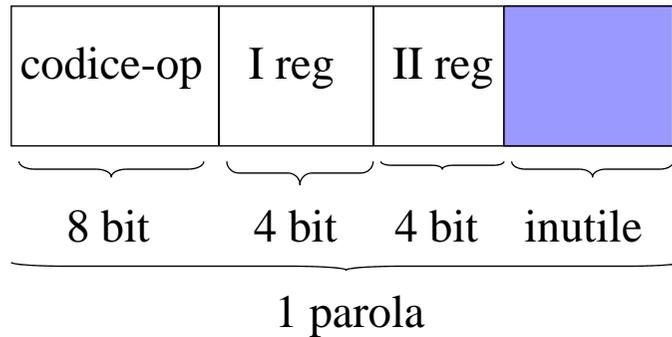
## ISTRUZIONI ARITMETICHE

eseguono somma, differenza, moltiplicazione e divisione **usando i registri come operandi**



ADD	00000010	FADD	00000011
SUB	00000100	FSUB	00000101
MULT	00000110	FMULT	00000111
DIV	00001000	FDIV	00001001
MOD	00001010		

### Istruzioni aritmetiche in linguaggio macchina:



Esempio: `0000010 ADD 0011 R3 0001 R1 xxxxxxxxxxxxxxxxxxxx`

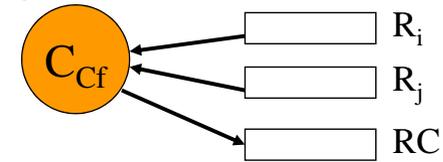
Notare che il risultato dell'operazione aritmetica viene messo nel primo registro

### Istruzioni di confronto

paragona il contenuto di 2 registri  $R_i$  ed  $R_j$  e:

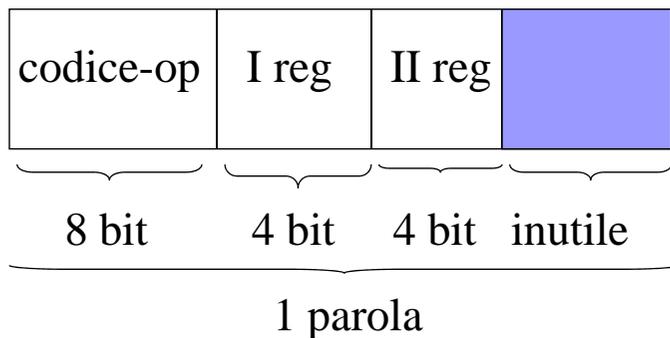
- se  $R_i < R_j$  mette **-1** nel registro RC
- se  $R_i = R_j$  mette **0** in RC
- se  $R_i > R_j$  mette **1** in RC

Registro di confronto



Codici: COMP 00100000

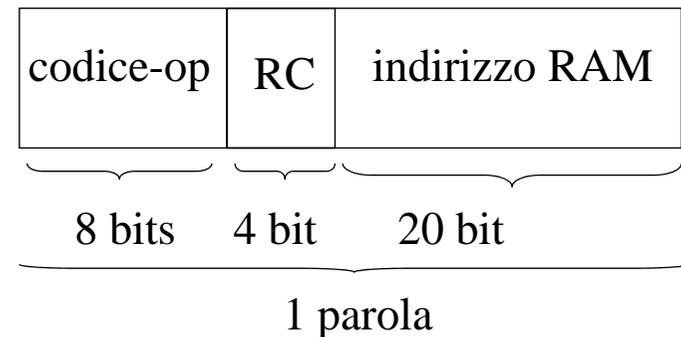
### Istruzioni di confronto in linguaggio macchina:



Codici: COMP 00100000

Esempio: `00100000 0010 0101 xxxxxxxxxxxxxxxxxxxx`

### Istruzioni di salto in linguaggio macchina:



Se il contenuto di RC soddisfa la condizione di salto aggiorna il PC all'indirizzo RAM indicato, Altrimenti incrementa il PC di 1

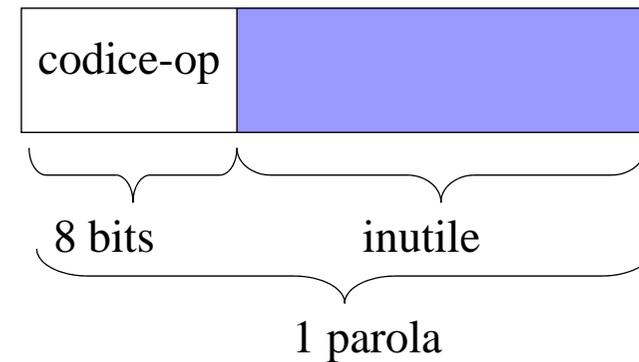
Esempio: `01000001 BRLT RC 00000000000000000001001 RAM Word 9`  
`10000000 BRANCH (RC) 000000000000000001010 RAM Word 10`

## Istruzione di STOP

termina il programma

Codice:    STOP    10000001

Istruzione di STOP in linguaggio macchina:



Esempio: 10000001 ~~xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx~~

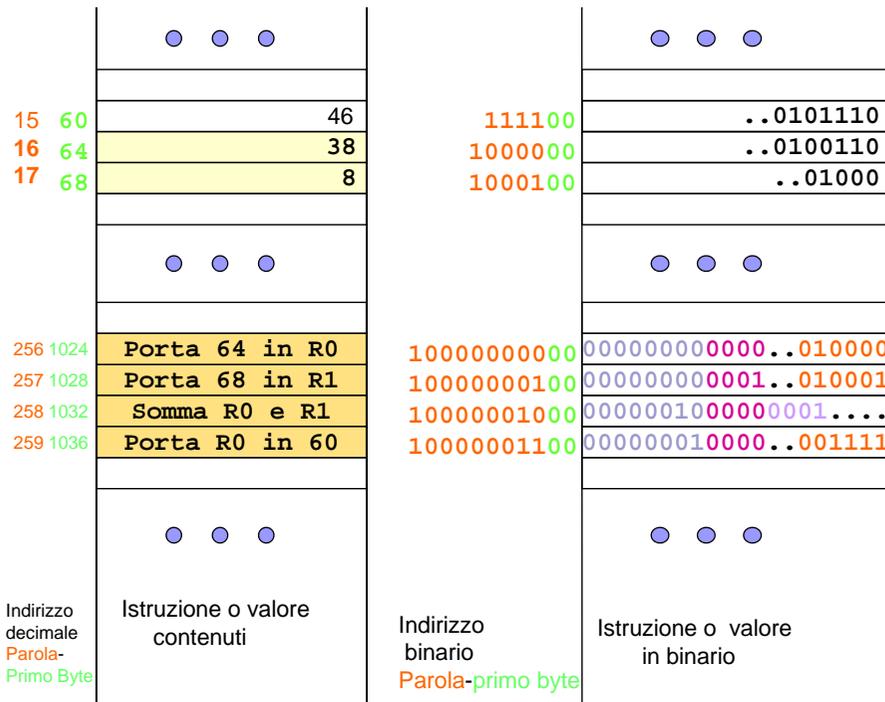
## Ciclo della CPU

- La CPU **esegue un programma memorizzato in RAM copiando** ad una ad una **le istruzioni nell'Instruction Register** dell'unita' di controllo
- **Ordine:** quello in cui sono memorizzate, a meno di istruzioni di salto
- **Registri speciali:**
  - contatore di programma (program counter)
  - Istruzione corrente (instruction register)

## Ciclo della CPU

- **Reperimento dell'istruzione:**
  - lettura della cella di RAM il cui indirizzo e' contenuto nel contatore di programma
  - caricamento del registro istruzione con l'istruzione
  - Incremento del contatore programma
- **Decodifica dell'istruzione:**
  - Trova gli operandi a seconda del codice operativo
  - Modifica contatore programma se istruzione di salto
- **Esecuzione dell'istruzione:**
  - Attiva i circuiti necessari





## Svantaggi del linguaggio macchina:

- programmi in binario sono **difficili** da scrivere, capire e cambiare
- il programmatore deve occuparsi di gestire la **RAM**: difficile ed inefficiente

primo passo → **Assembler**