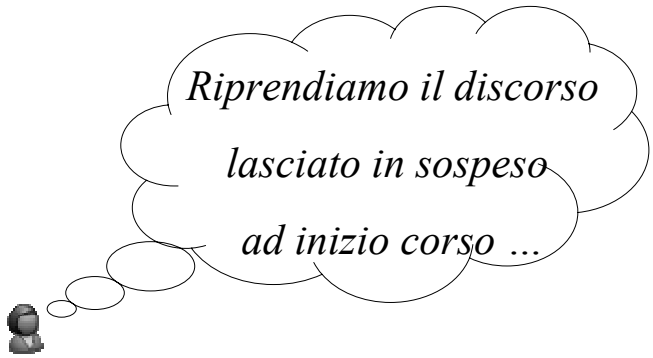
Architettura dell'elaboratore



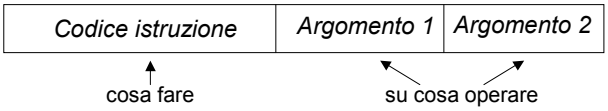
Riepilogando …

- I programmi e i dati **risiedono** nella memoria secondaria
- Per essere eseguiti (i programmi) e usati (i dati) vengono **copiati** nella memoria principale
- Il processore è in grado di **eseguire in modo sequenziale** le istruzioni di cui sono composti i programmi

Riepilogando …

- I programmi sono fatti di **istruzioni elementari**
- Le istruzioni possono avere **formati diversi**

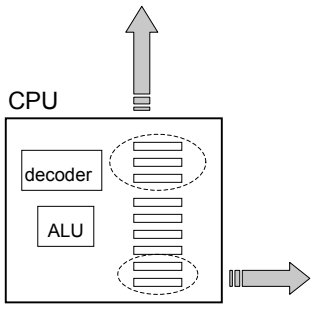
Es.

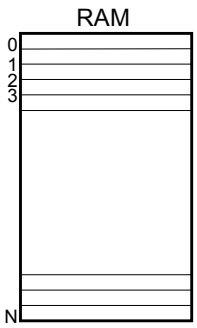


Riepilogando …

MAR = Memory Address Register
MDR = Memory Data Register

[RC = Registro di Controllo]





PC = Program Counter
IR = Instruction Register

Ad ogni ciclo di clock

1. Si legge dalla RAM la **prossima** istruzione da eseguire (**fetch**)
2. Si **decodifica** l'istruzione
3. Si **esegue** l'istruzione (**execute**)

Ad ogni ciclo di clock

1. **Fetch**
   i. si legge dalla RAM l'istruzione che si trova in RAM[PC] e si incrementa il PC
2. Si **decodifica** l'istruzione
3. Si **esegue** l'istruzione (**execute**)

Ad ogni ciclo di clock

1. **Fetch**
   i. MAR := PC
   ii. RC := “r”
   iii. MDR := RAM[MAR]
   iv. IR := MDR
   v. PC := PC + 1

2. Si **decodifica** l’istruzione

3. Si **esegue** l’istruzione (**execute**)

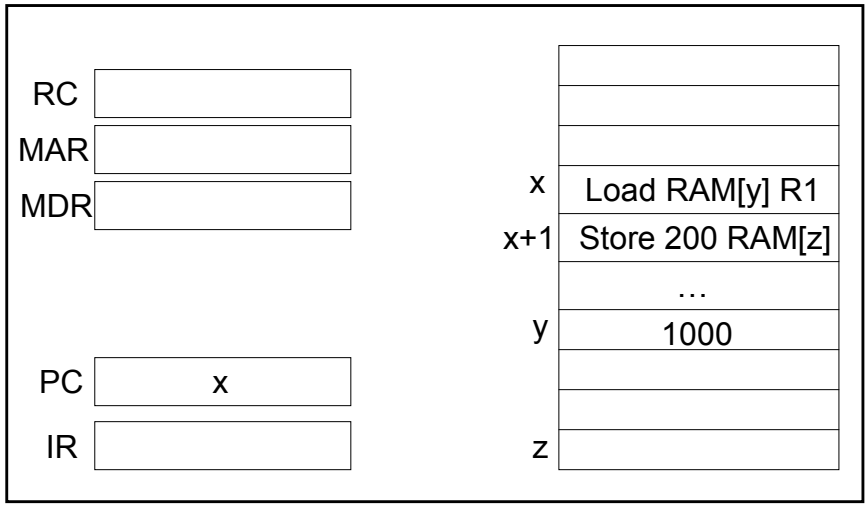
Ad ogni ciclo di clock

1. Fetch

2. Si **decodifica** l’istruzione
   - istruzioni di lettura in memoria
   - istruzioni di scrittura in memoria
   - istruzioni aritmetico / logiche
   - istruzioni di salto

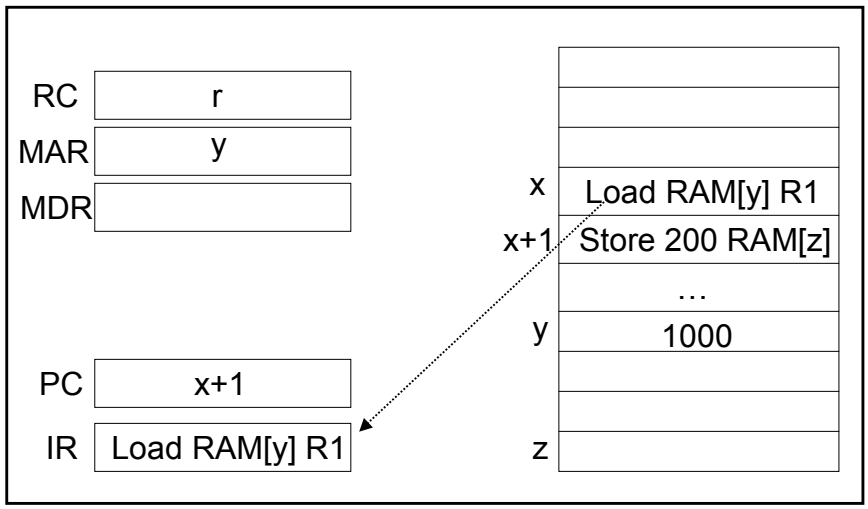
3. Si **esegue** l’istruzione
   - a seconda del tipo di istruzione verranno intraprese azioni diverse

Lettura dalla memoria (senza considerare il fetch dell'istruzione)



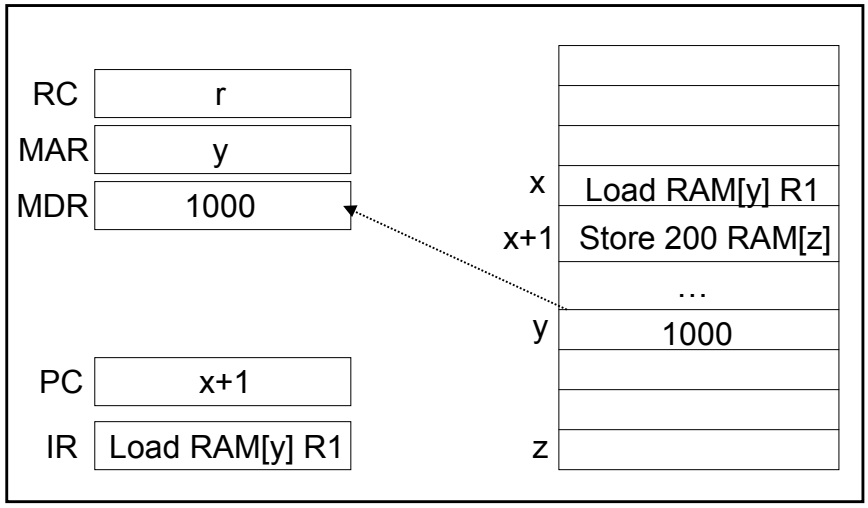
Descrizione ad alto livello, i dettagli al corso di Architetture …

Lettura dalla memoria (senza considerare il fetch dell'istruzione)

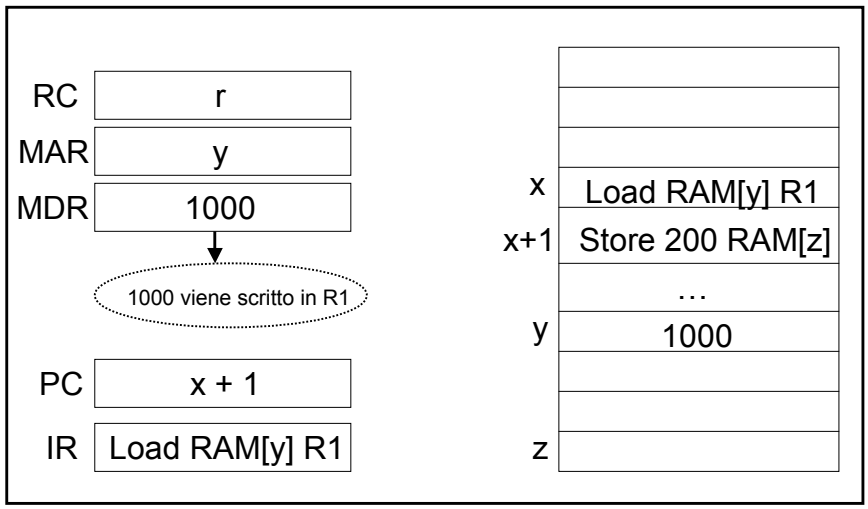


Descrizione ad alto livello, i dettagli al corso di Architetture …

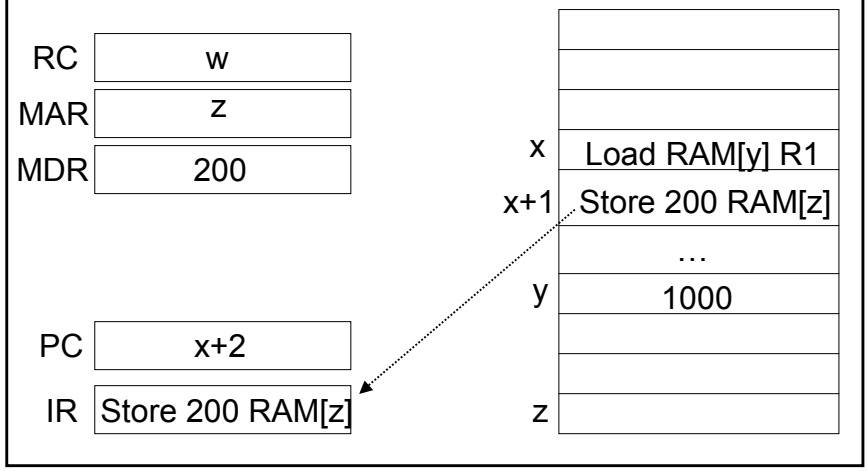
Lettura dalla memoria (senza considerare il fetch dell'istruzione)



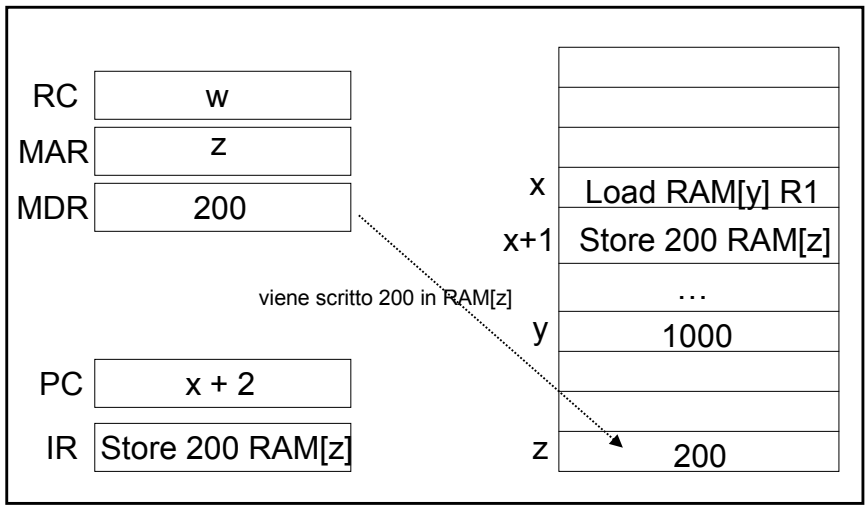
Lettura dalla memoria (senza considerare il fetch dell'istruzione)



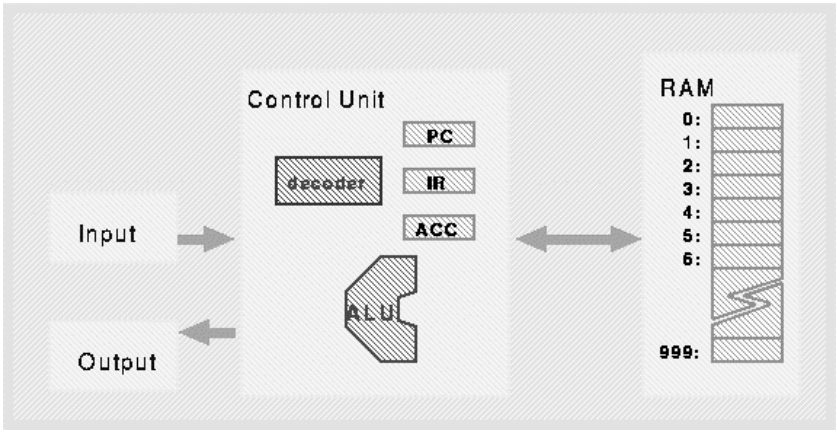
Scrittura in memoria (senza considerare il fetch dell'istruzione)



Scrittura in memoria (senza considerare il fetch dell'istruzione)



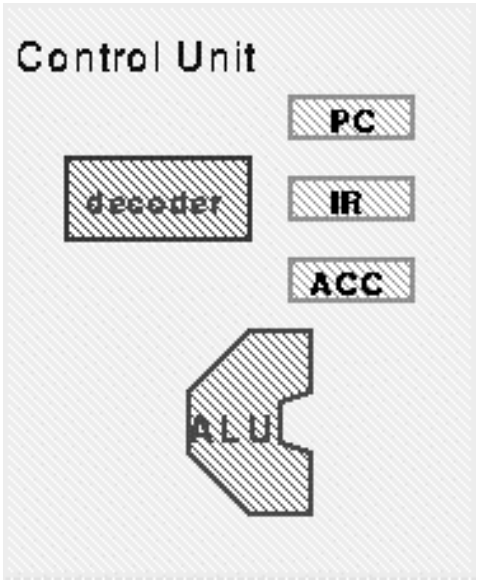
La macchina di Von Neumann



La macchina di Von Neumann

1. **RAM**: 1000 celle di memoria
2. **Input**: dispositivo per l'introduzione di valori interi
3. **Output**: dispositivo per l'output di valori interi
4. **Control Unit**: realizza il funzionamento della macchina

La macchina di Von Neumann



PC = Program Counter

IR = Instruction Register

ACC = Accumulatore

Istruzioni della macchina: somma

1. **codice**: 0 **parametro**: N
2. **descrizione**: somma tra due interi (il valore contenuto in ACC e il valore contenuto nella cella di memoria specificata dal parametro)
3. **esecuzione**: ACC := ACC + RAM[N]

:= indica l'operatore di assegnazione (= in C)

Istruzioni della macchina: differenza

1. **codice**: 1 **parametro**: N
2. **descrizione**: differenza tra due interi (il valore contenuto in ACC e il valore contenuto nella cella di memoria specificata dal parametro)
3. **esecuzione**: ACC := ACC - RAM[N]

Istruzioni della macchina: lettura

1. **codice**: 2 **parametro**: nessuno
2. **descrizione**: legge un valore numerico dal dispositivo di ingresso e lo copia in ACC
3. **esecuzione**: ACC := input

Istruzioni della macchina: scrittura

1. **codice**: 3 **parametro**: nessuno
2. **descrizione**: scrive il valore contenuto in ACC sul dispositivo di uscita
3. **esecuzione**: output := ACC

Istruzioni della macchina: memorizza

1. **codice**: 4 **parametro**: N
2. **descrizione**: memorizza il valore contenuto in ACC nella cella di memoria specificata dal parametro
3. **esecuzione**: RAM[N] := ACC

Istruzioni della macchina: carica

1. **codice**: 5 **parametro**: N
2. **descrizione**: carica in ACC il valore contenuto nella cella specificata dal parametro
3. **esecuzione**: ACC := RAM[N]

Istruzioni della macchina: salto

1. **codice**: 6 **parametro**: N
2. **descrizione**: copia il parametro nel registro PC
3. **esecuzione**: PC := N

Istruzioni della macchina: salto condizionato

1. **codice**: 7 **parametro**: N
2. **descrizione**: se il valore di ACC è zero, copia il parametro nel registro PC
3. **esecuzione**: if (ACC == 0) then PC := N

Istruzioni della macchina: end

1. **codice**: 8 **parametro**: nessuno
2. **descrizione**: ferma l'interpretazione del programma
3. **esecuzione**: ferma l'interpretazione del programma

Codifica delle istruzioni come numeri interi

1. **Moltiplico per 1000** il codice dell'istruzione
2. **Sommo** il valore dell'eventuale **parametro**

NB: il parametro dovrà essere compreso tra 0 e 999

Codifica delle istruzioni come numeri interi

Esempi

1. Voglio **scrivere nella cella 300**
   i. istruzione 4 — 4*1000 = 4000
   ii. parametro 300 — 4000+300=**4300**
2. Voglio **sommare al contenuto di ACC il valore della cella 167**
   i. istruzione 0 — 0*1000 = 0
   ii. parametro 167 — 0+167=**167**

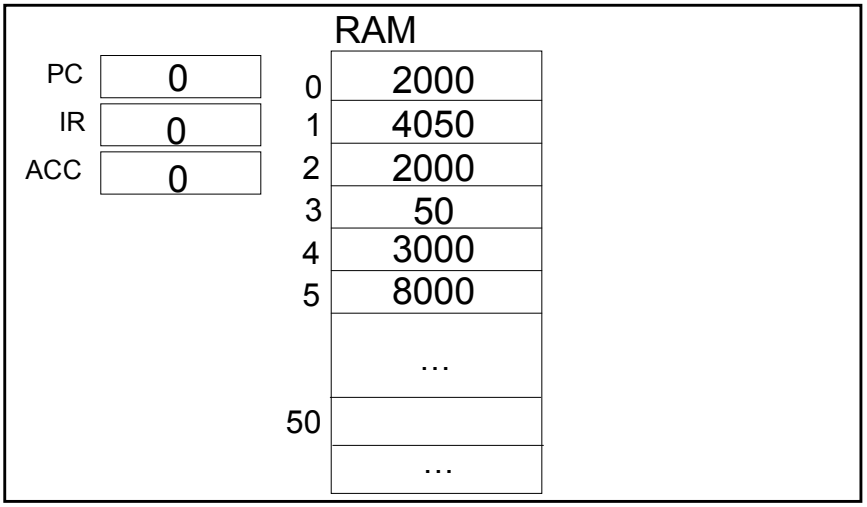
Codifica delle istruzioni come numeri interi

Esempi

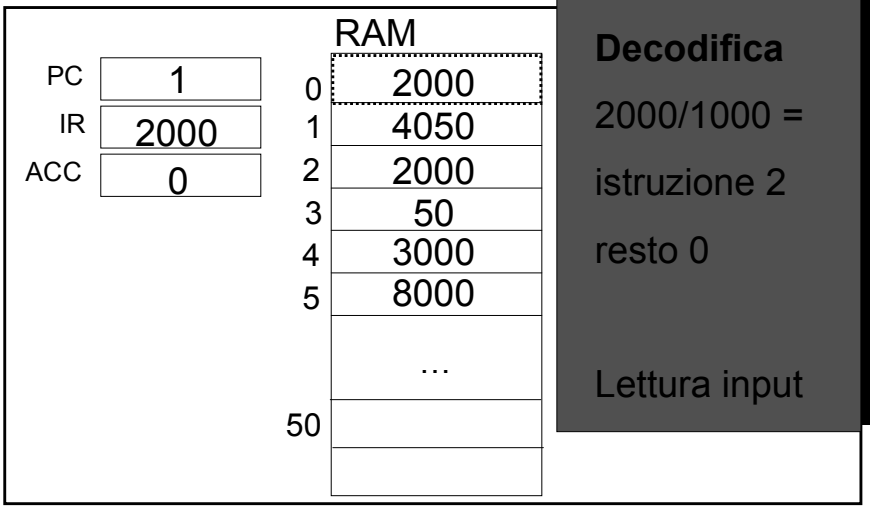
1. **6899**
   i. 6899/1000 = 6 + 899
   ii. istruzione 6 (salto)
   iii. parametro 899 (PC := 899)

2. **8000**
   i. 8000/1000 = 8
   ii. istruzione 8 (end)

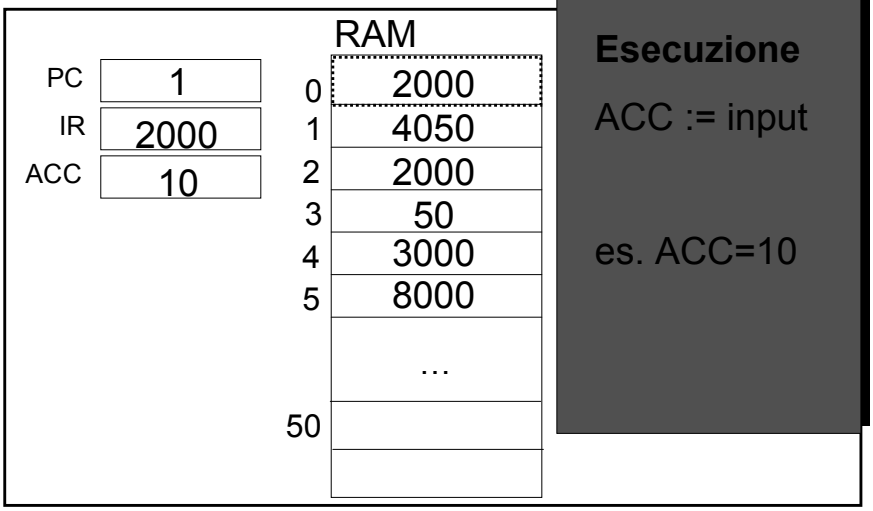
Un programma per la somma di due numeri



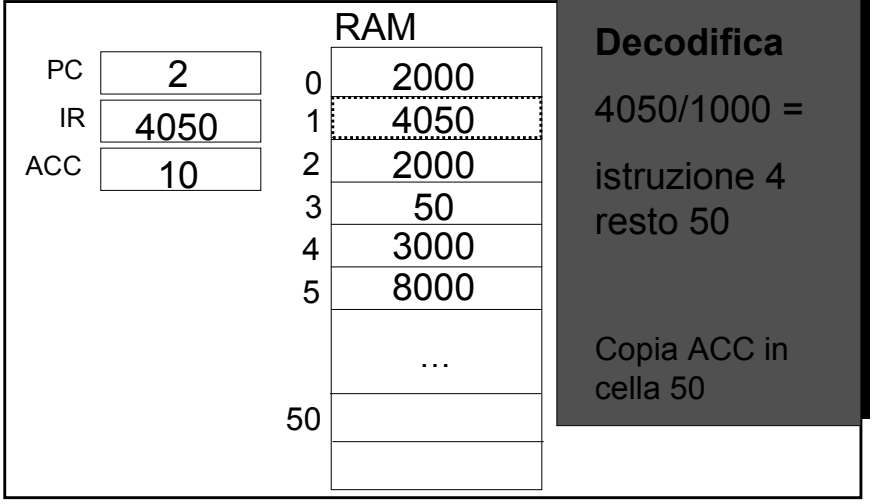
Un programma per la somma di due numeri



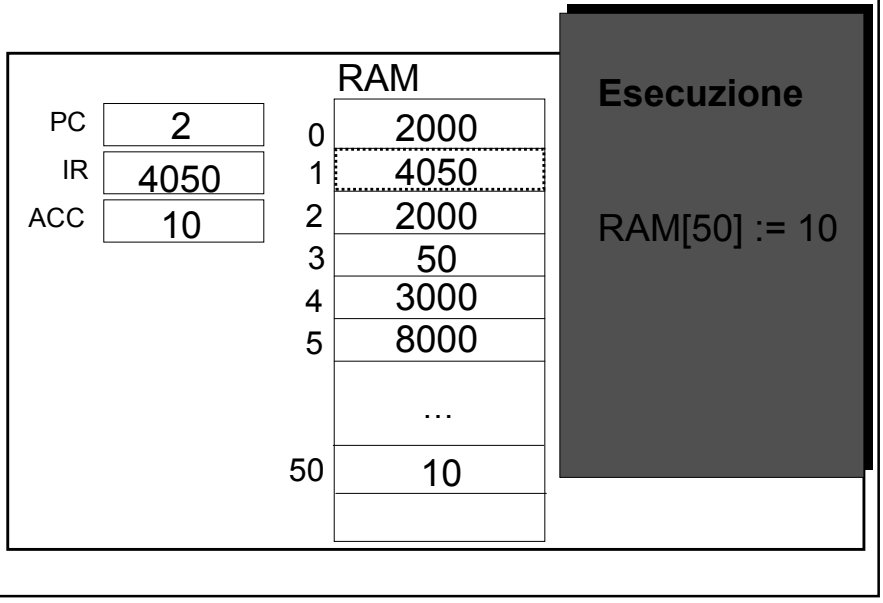
Un programma per la somma di due numeri



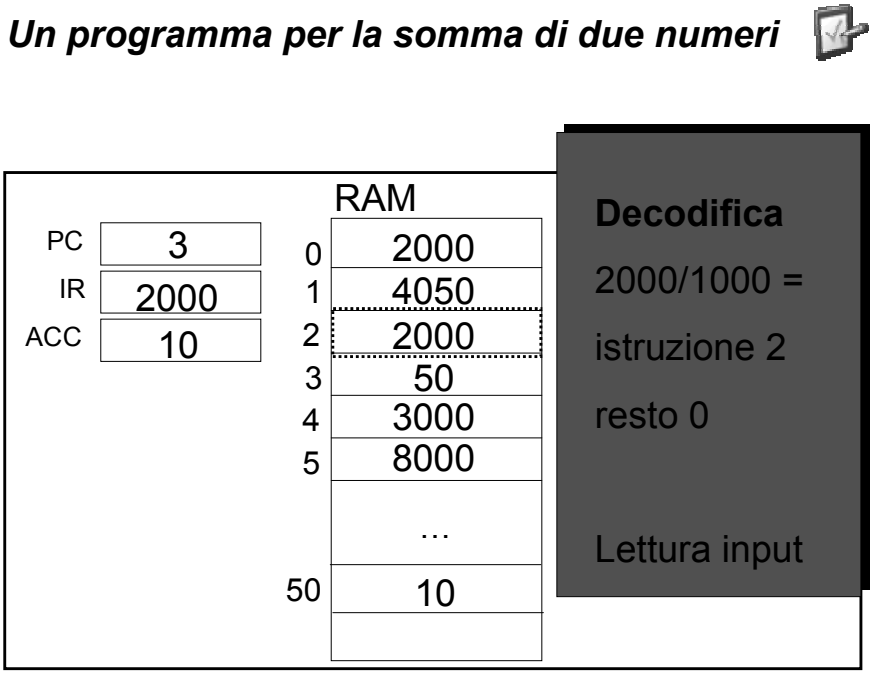
Un programma per la somma di due numeri



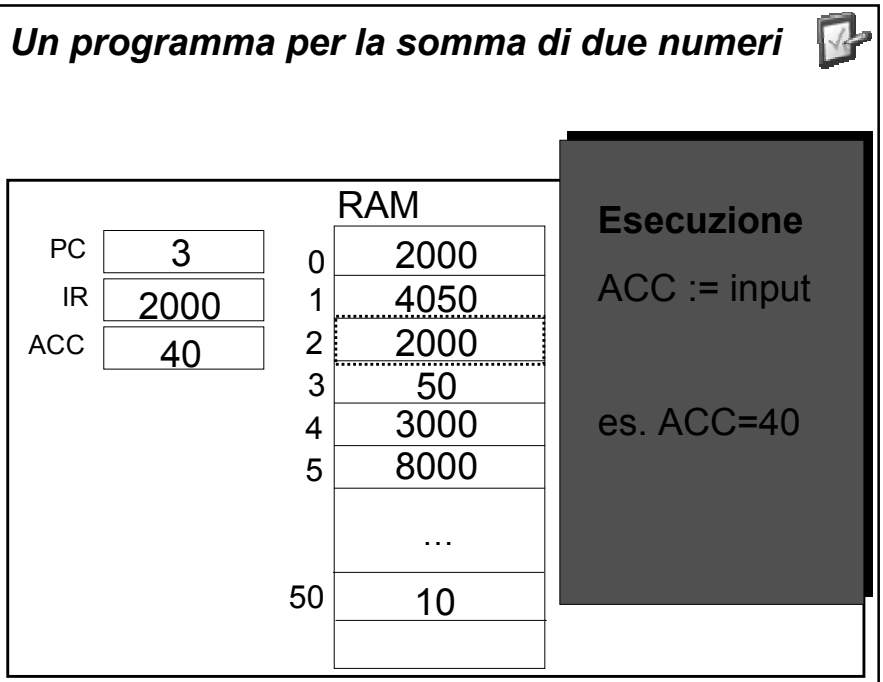
Un programma per la somma di due numeri



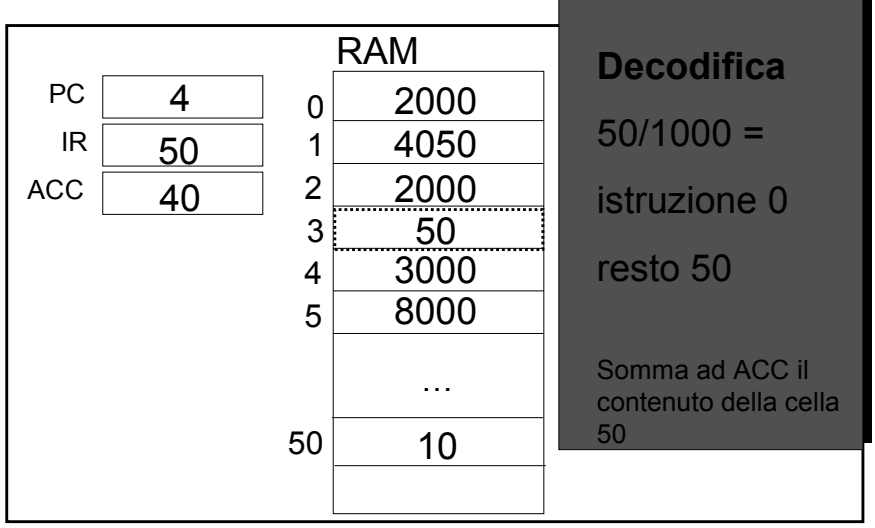
Un programma per la somma di due numeri



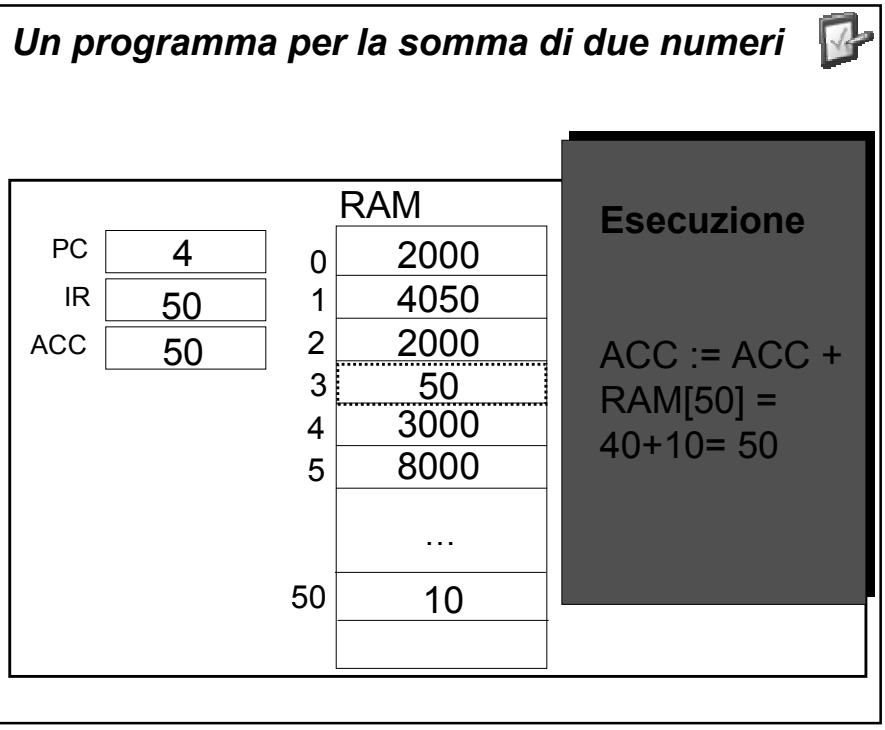
Un programma per la somma di due numeri



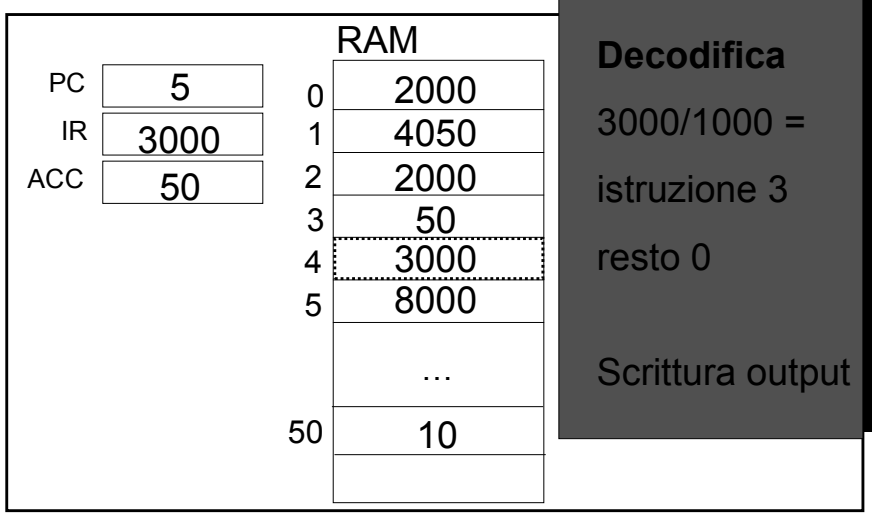
Un programma per la somma di due numeri



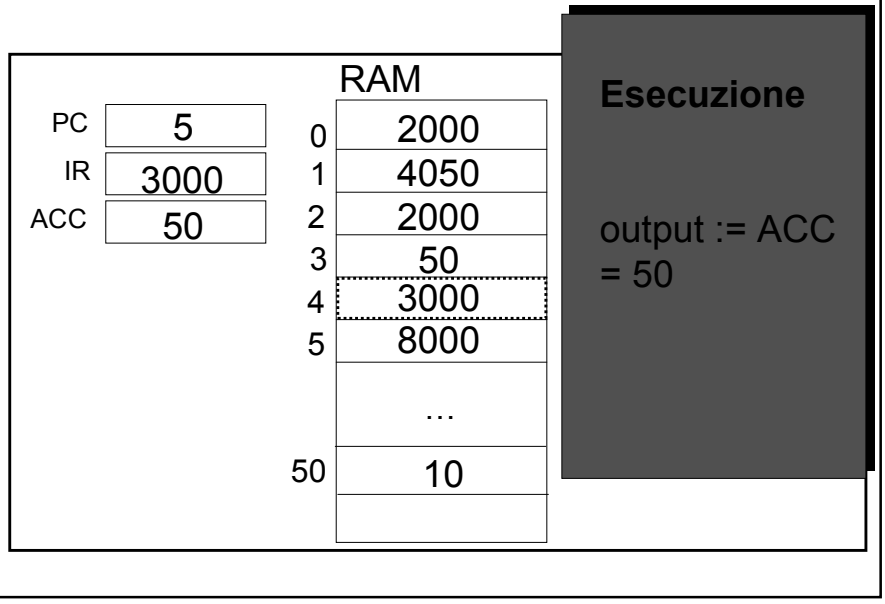
Un programma per la somma di due numeri



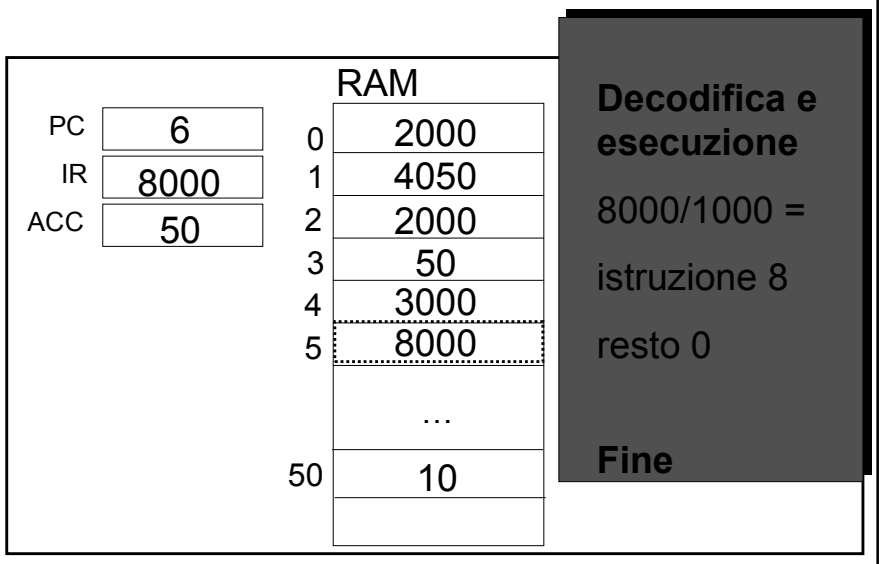
Un programma per la somma di due numeri



Un programma per la somma di due numeri



Un programma per la somma di due numeri



Un programma per ripetere N volte la somma di 2 numeri