Corso di Laurea in Informatica
Architettura degli elaboratori
a.a. 2025-2026

La macchina programmata

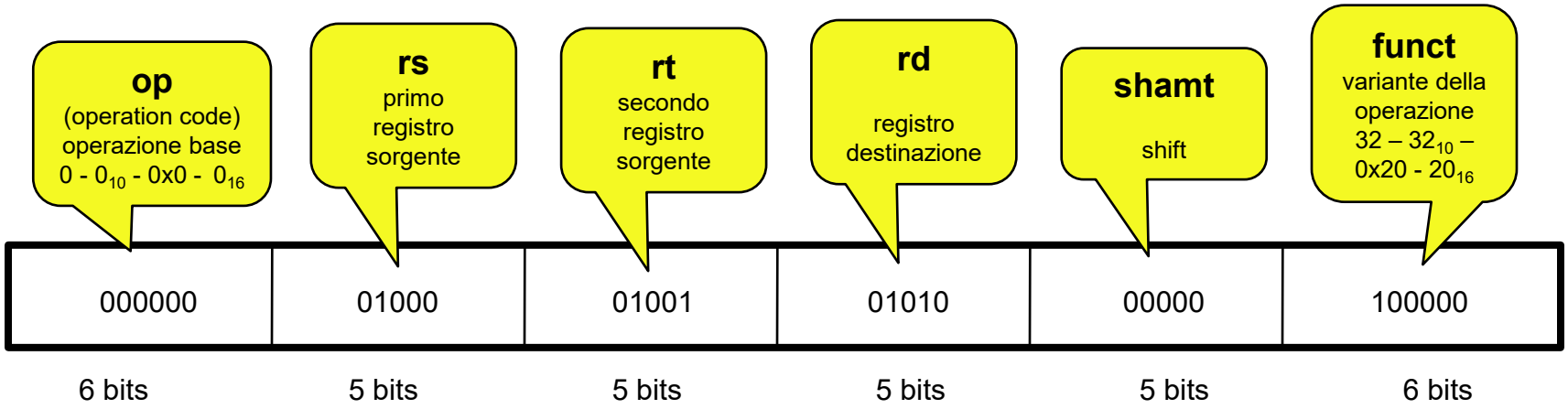
Istruzioni J-type
Istruzioni di salto

Claudia Raibulet
claudia.raibulet@unimib.it

Istruzione R-Type *(già vista)*

quello che si vuole fare (in italiano)
numeri decimali (per comodità)

"somma i contenuti del registro 8 e del registro 9 e metti il risultato nel registro 10"

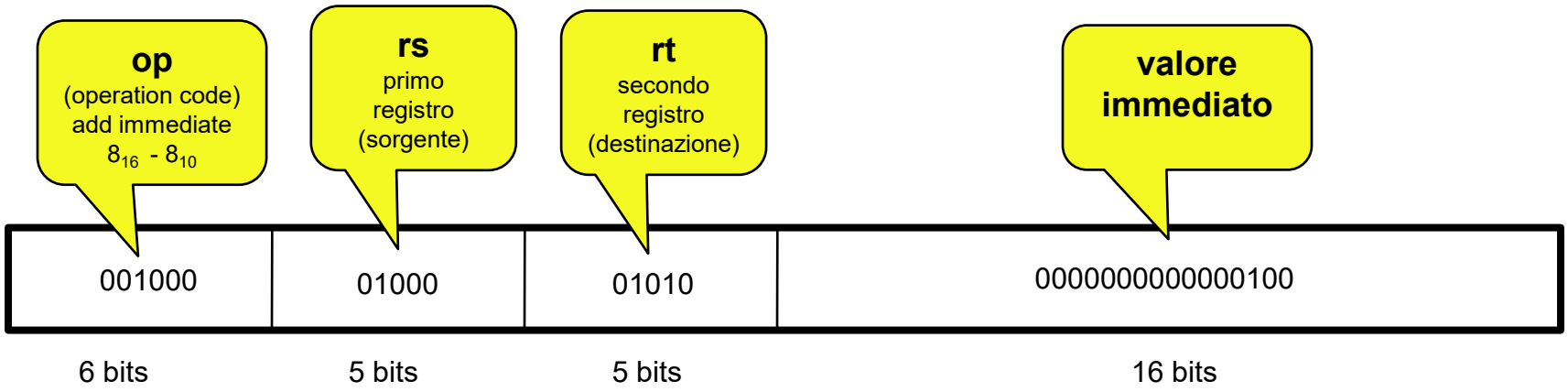


- **Se**
  - rs contiene inizialmente 64 (00000000000000000000000001000000)
  - rt contiene inizialmente 4 (00000000000000000000000000000100)
- **dopo l'esecuzione**
  - rd contiene 68 (00000000000000000000000001000100)
- **Formato Assembly**
  - add $t2, $t0, $t1
  - (attenzione all'ordine degli operandi)

Istruzione I-type *(add immediate, già vista)*

quello che si vuole fare (in italiano)
numeri decimali (per comodità)

"somma il valore 4 al registro 8 e metti il risultato nel registro 10"



- se rs contiene inizialmente 64 (00000000000000000000000001000000)
- dopo l'esecuzione rt contiene 68 (00000000000000000000000001000100)

Problema: qual è il range di valori immediati che si può esprimere con 16 bit in complemento a 2?
(-32768 <= valore <= 32767). Se serve un valore più grande, va gestito a livello programmativo.

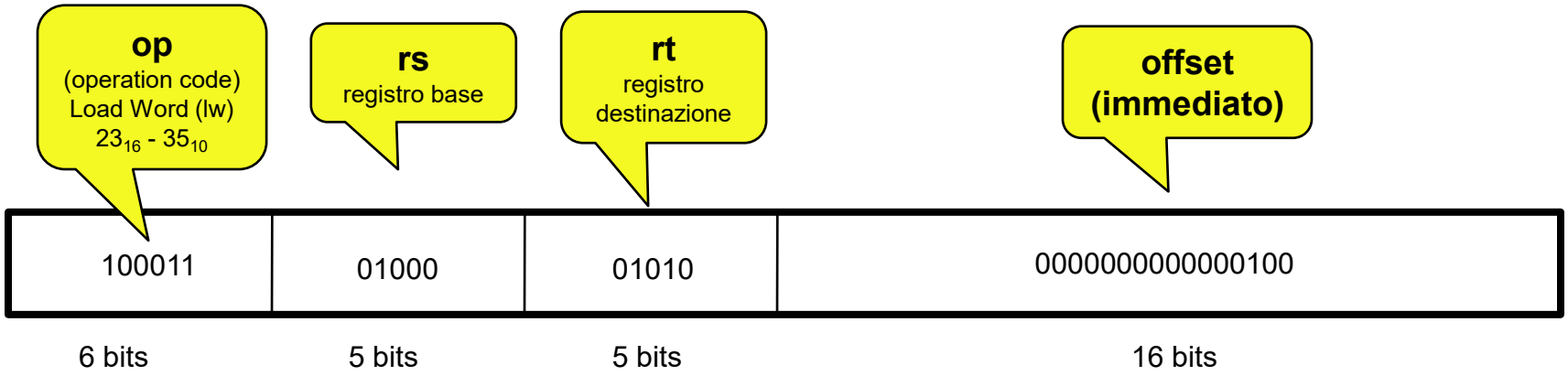
Formato Assembly:

addi $t10, $t8, 4

Istruzione I-type load word *(già vista)*

quello che si vuole fare (in italiano)
numeri decimali (per comodità)

"carica nel registro 10 il contenuto della parola (32 bit)
che è all'indirizzo di memoria ottenuto come somma del registro 8 e dell'offset immediato 4"

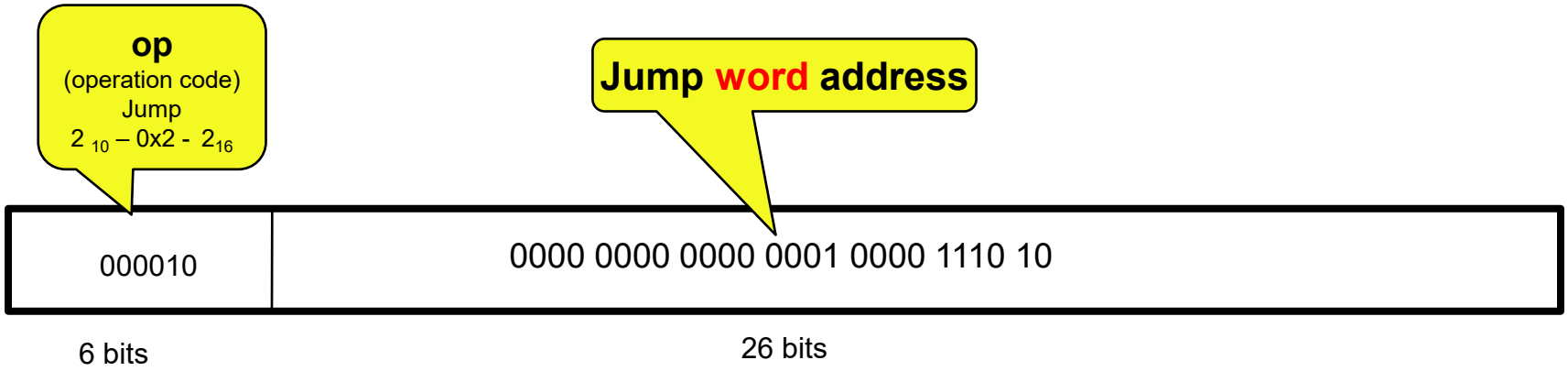


- se rs contiene inizialmente 64 (00000000000000000000000001000000)
- l'indirizzo in memoria della word da caricare è 68 (00000000000000000000000001000100)

- in rt vengono copiati 32 bit (1 word) a partire dall'indirizzo…
- …qualunque sia il significato di quei 32 bit

Istruzione J-Type (Jump)

"salta all'istruzione il cui indirizzo è $00010e8_{16}$"



Carica in PC 0000 0000 0000 0001 0000 1110 10**00**

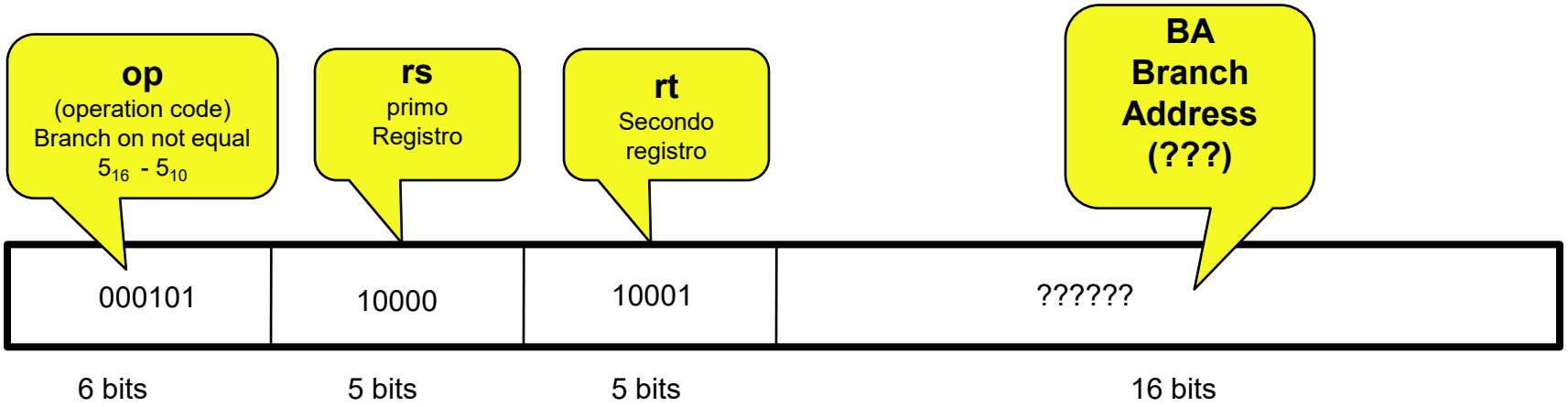
...poiché le istruzioni devono essere allineate al word
(responsabilità di chi scrive e carica i programmi in memoria;
in pratica, assembler e loader)

Quindi con 26 bit si indirizzano 2^{28} Byte di memoria programma oppure 2^{26} Word

I 4 bit più significativi sono i 4 bit più significativi di PC

Istruzione I-type (branch)

"salta a Branch Address se il contenuto di rs è diverso dal contenuto di rt"

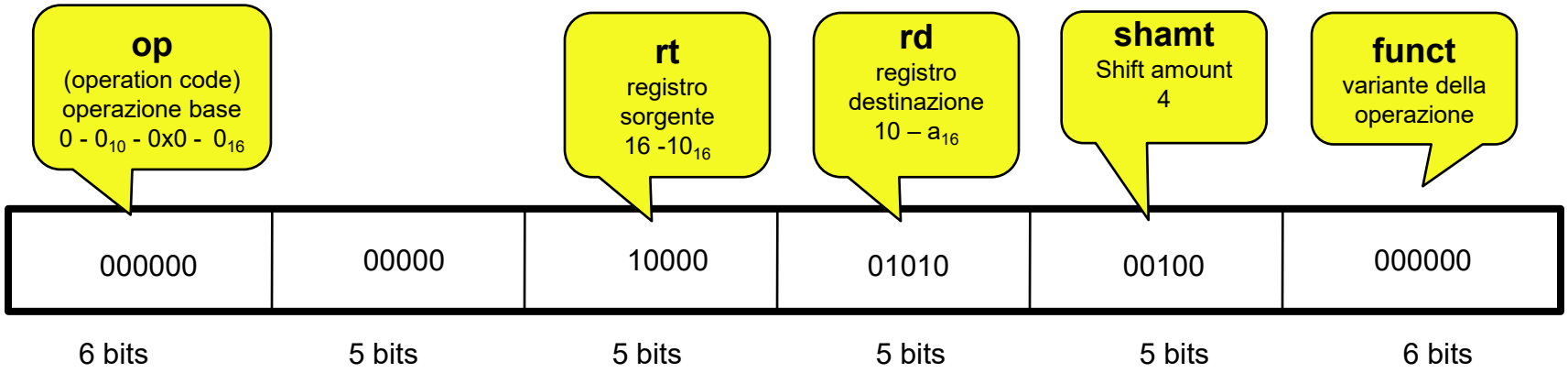


- Formato Assembly: bne $s0, $s1, Exit
- Exit è una etichetta che l'assembler traduce in uno spiazzamento
- Problema 1: "dove" si può saltare? (valore immediato – 2^{15} <= BA < 2^{15})
- Soluzione 1: usare un registro base R (indirizzo di salto = R + BA)
- Soluzione 2: Usare PC come registro base
- Se CA è l'indirizzo dell'istruzione corrente, si salta a CA + 4 + BranchAddr*4 (perche' ogni istruzione e' memorizzata su 4'byte; BranchAddr rappresenta il numero di word da saltare rispetto all'indirizzo memorizzato nel PC))
  - (PC è già stato incrementato, quindi PC contiene CA + 4)
  - Ampiezza del salto: +- 2^{15} rispetto a PC
  - Adeguata in moltissimi casi (loop, if…)

Istruzione R-Type (shift left)

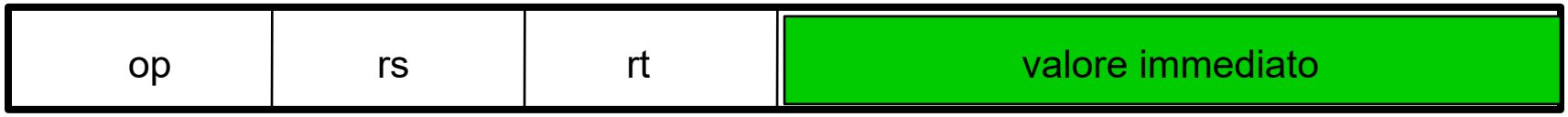
quello che si vuole fare (in italiano)
numeri decimali (per comodità)

"shift di 4 bit a sinistra i contenuti del registro 16 e metti il risultato nel registro 10"

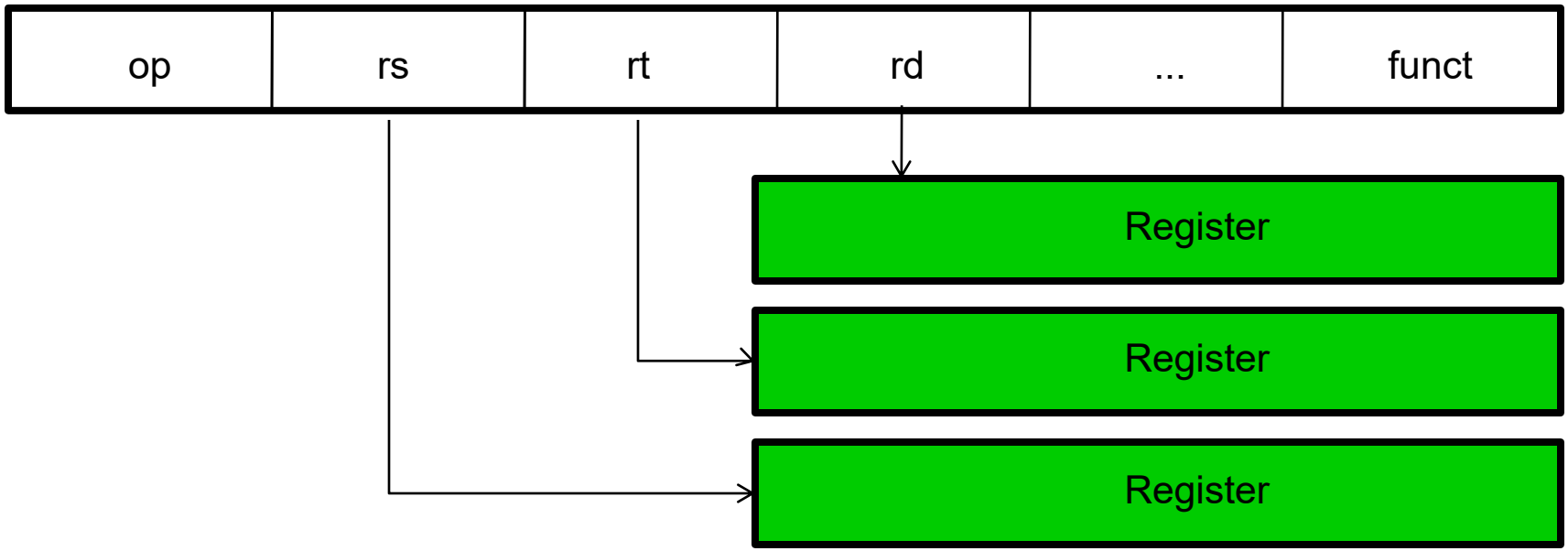


- **Se**
  - rt contiene inizialmente 9 (0000 0000 0000 0000 0000 0000 1001) (= 9_{16})
- **dopo l'esecuzione**
  - rd contiene 144 (0000 0000 0000 0000 0000 0000 1001 0000) (= 90_{16})

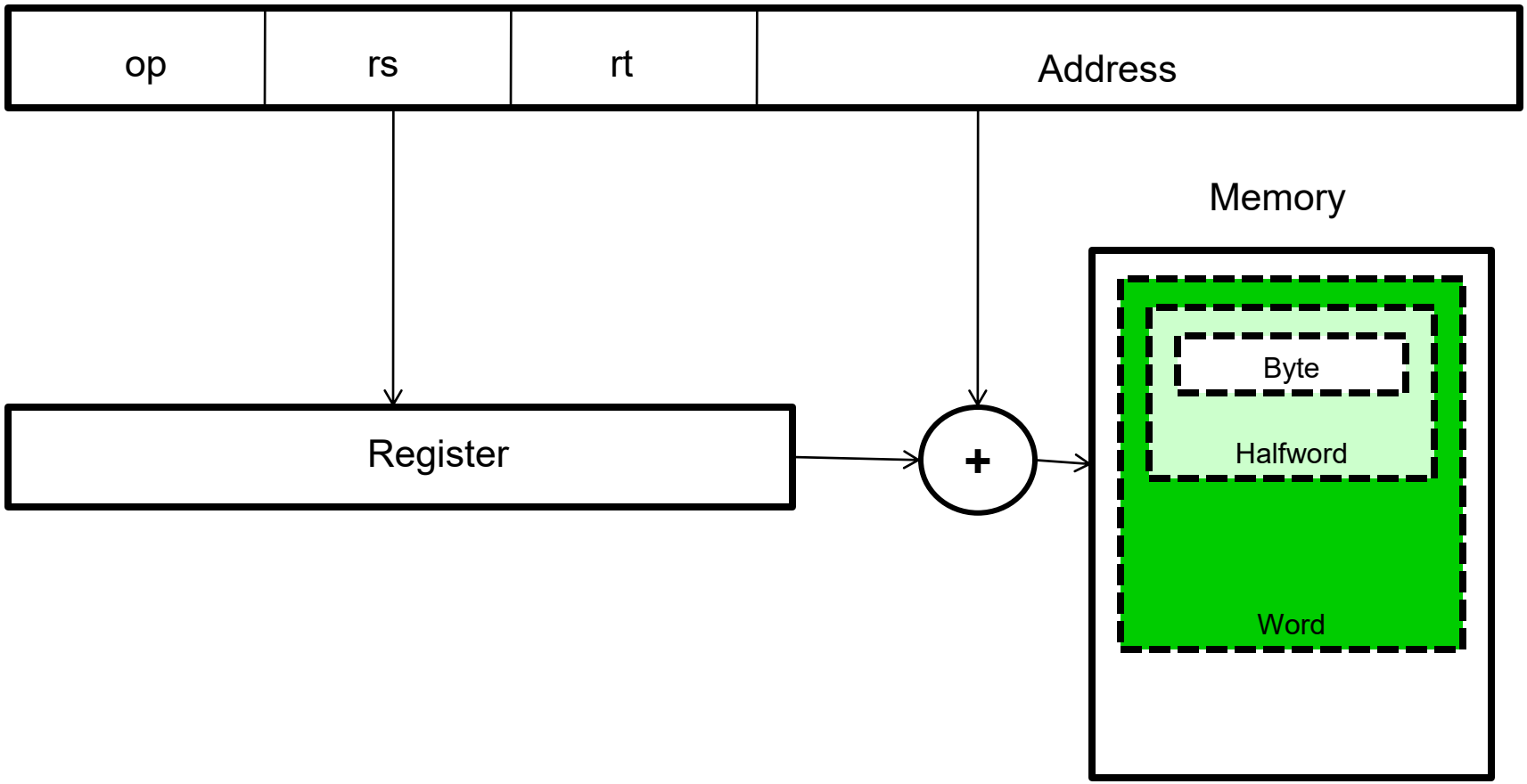
Immediate addressing



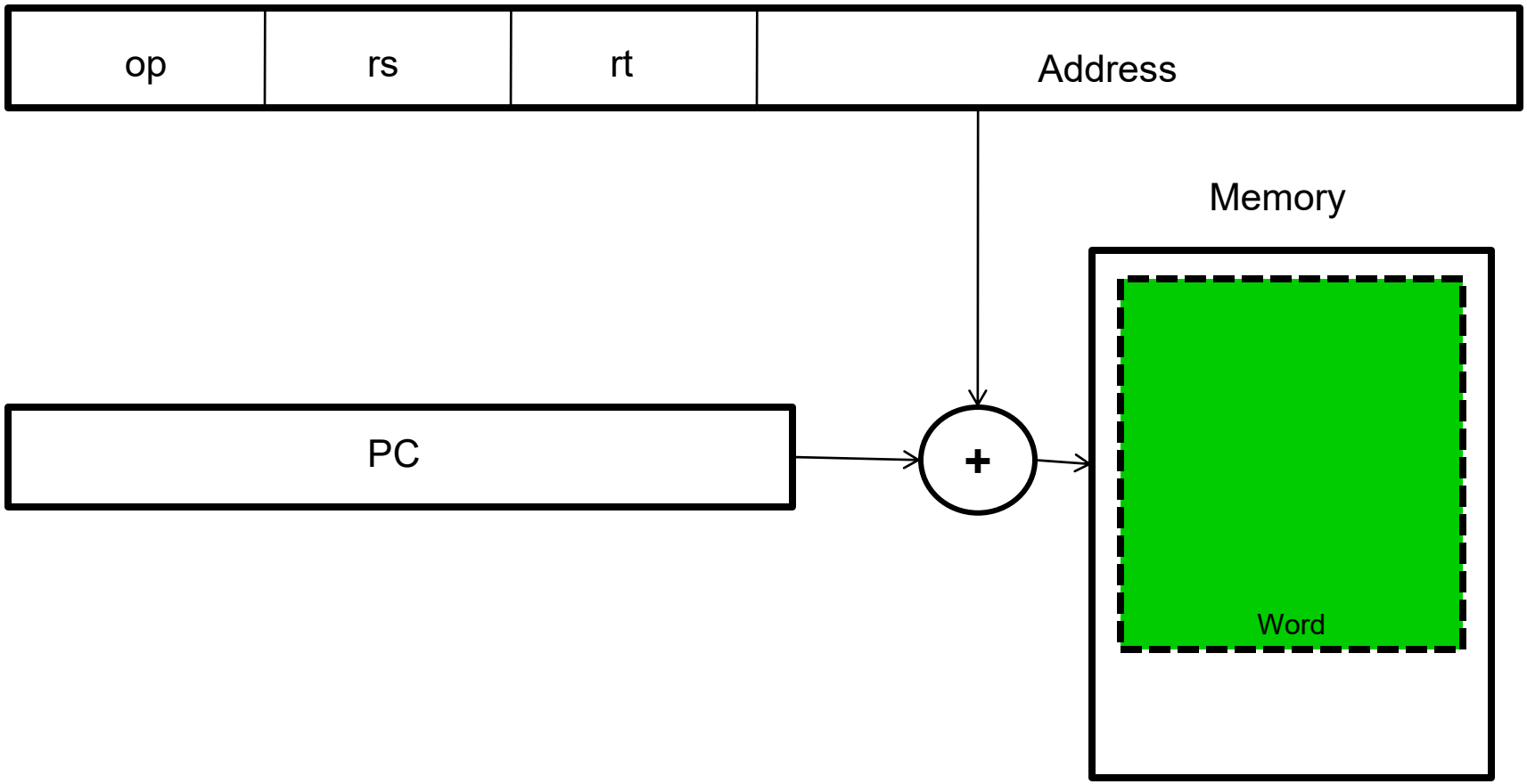
Register addressing



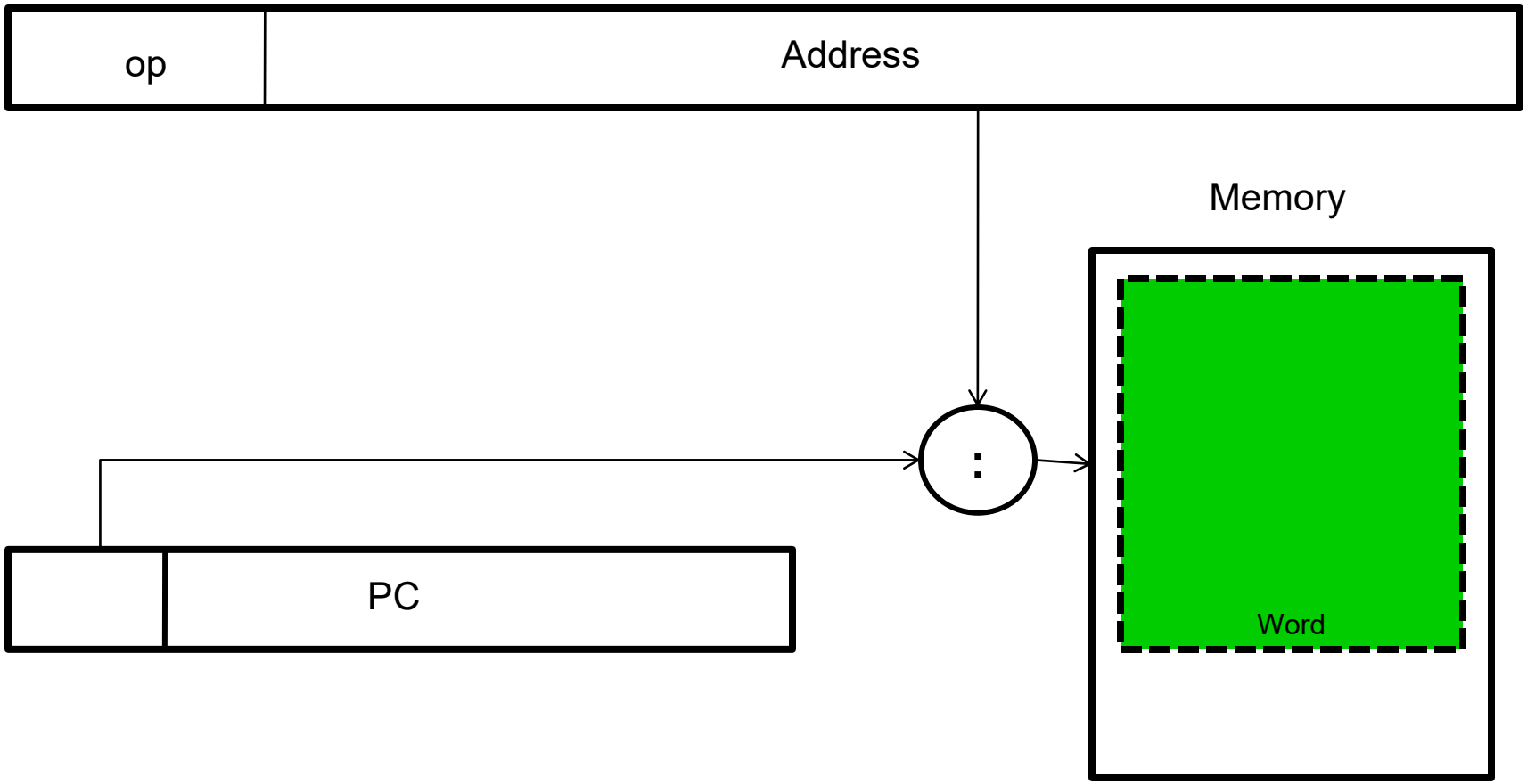
Base addressing



PC-relative addressing



Pseudodirect addressing



Esercizio

Si chiede di scrivere le istruzioni MIPS per il seguente codice C o Java:

```
if (i==j)
        f=g+h;
else
        f=g-h;
```

Considerando che le 5 variabili sono memorizzate nei registri: $s0 - $s4.
L'indirizzo della prima istruzione è 80000.

Soluzione (tutto in base 10):

80000		bne $s3, $s4, ELSE	5	19	20		2	
80004		add $s0, $s1, $s2	0	17	18	16	0	32
80008		j EXIT	2			20004		
80012	ELSE:	sub $s0, $s1, $s2	0	17	18	16	0	34
80016	EXIT:	...						