Lezione 18

CPU a singolo ciclo

Proff. A. Borghese, F. Pedersini

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

Sommario

- La CPU
- Sintesi di una CPU per le istruzioni di tipo R
- Sintesi di una CPU per le istruzioni di tipo I (memoria)
- Sintesi di una CPU per le istruzioni di tipo I (salti)
- CPU che gestisce istruzioni: lw/sw e branch

Obiettivo

- Obiettivo: costruzione di una CPU completa
  - In grado di eseguire:
  - Accesso alla memoria in lettura (**`lw`**) o scrittura (**`sw`**).
  - Istruzioni logico-matematiche (e.g. **`add, sub, and`**....).
  - Istruzioni di salto condizionato (**`branch`**) o incondizionato (**`jump`**)
- Per la sintesi dei circuiti è necessario definire:
  - Formati di istruzione: **R, I, J**
  - Ciclo esecuzione istruzioni

Architettura delle istruzioni di tipo: *R* e *I*

- Definizione dei campi istruzione:
  - **`op (opcode):`** identifica il tipo di istruzione
  - **`rs`**: registro contenente il **primo** operando **sorgente**
  - **`rt`**: registro contenente il **secondo** operando **sorgente**
  - **`rd`**: registro **destinazione** contenente il risultato
  - **`shamt`**: shift amount (scorrimento)
  - **`funct`**: indica la variante specifica dell'operazione
  - **`Indirizzo`**: offset

R

op	rs	rt	rd	shamt	funct
6 bit	5 bit	5 bit	5 bit	5 bit	6 bit

I

op	rs	rt	Indirizzo
6 bit	5 bit	5 bit	16 bit

Ciclo di esecuzione di istruzioni MIPS

1. **FETCH**: preleva l'istruzione dalla memoria e la inserisce in IR

2. **DECODIFICA**: Capisce di che tipo di istruzione si tratta
   - usa l'istruzione per decidere cosa fare
   - recdupera gli operandi, contenuti nei registri del Register File

3. **Da qui le istruzioni si differenziano:**
   - **Calcolo**: utilizzo dell'ALU dopo aver letto i registri:
     - per calcolare l'indirizzo in memoria
     - per eseguire un'operazione logico-aritmetica
     - per effettuare un test.
   - **Accesso** alla memoria
   - **Scrittura** del risultato nel register file

Rapporto UC – Dati

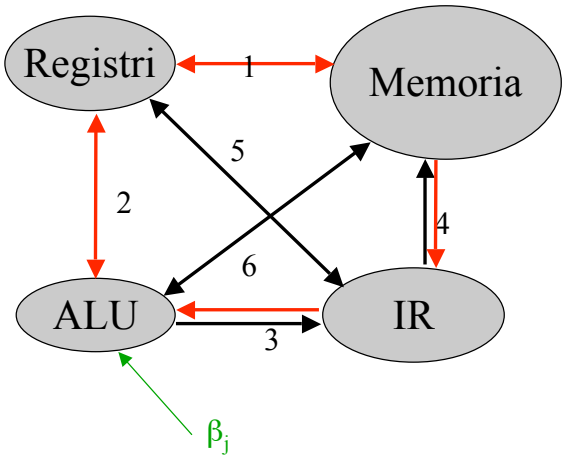
- CPU: approccio alla progettazione di tipo **Controllore – Data-path**
  - **Fasi comuni:** (nel ciclo di esecuzione)
    Fetch, Decodifica (generazione dei segnali di controllo)
  - **Fasi specifiche:**
    Esecuzione, Accesso memoria, WriteBack

I componenti di un'architettura – Datapath

- **CPU:**
  - **Banco di registri (Register File)** in cui memorizzare i dati di utilizzo più frequente. Tempo di accesso 10 volte più veloce della memoria principale.
  - Registro **Program counter (PC)** Contiene l'indirizzo dell'istruzione corrente. Da aggiornare durante l'evoluzione del programma.
  - Registro **Instruction Register (IR)** Contiene l'istruzione in corso di esecuzione. Verrà utilizzato più avanti nelle architetture multi-ciclo.
  - **Arithmetic Logic Unit (ALU):** Unità per l'esecuzione delle operazioni aritmetico-logiche. I dati possono provenire da registri oppure direttamente dalla memoria, a seconda delle modalità di indirizzamento previste.
  - **Unità di controllo (UC):** controlla il flusso e determina le operazioni di ciascun blocco.
- **MEMORIA:**
  - Memoria Istruzioni
  - Memoria Dati

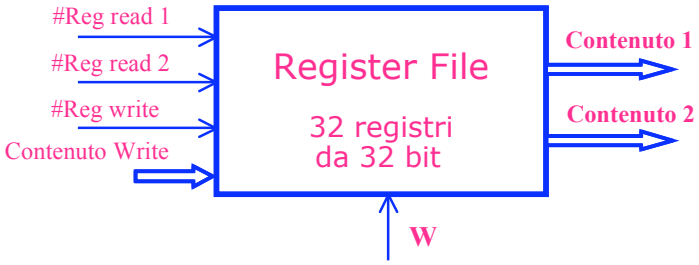
L'unità di controllo

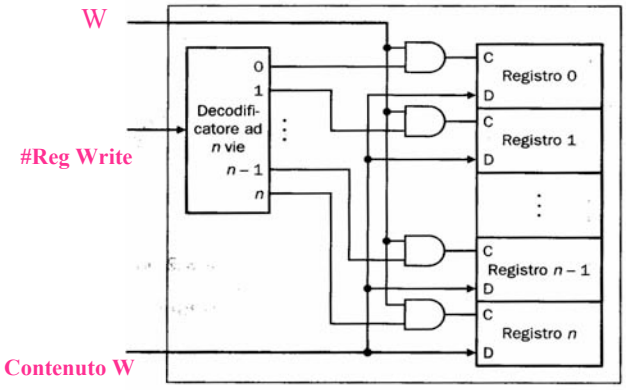
- L' Unità di controllo (UC) coordina i flussi di informazione (è il "cervello" della CPU):
  - selezionando l'operazione opportuna delle ALU.
  - abilitando le vie di comunicazione opportune a seconda dell'istruzione in corso di esecuzione.



Register file

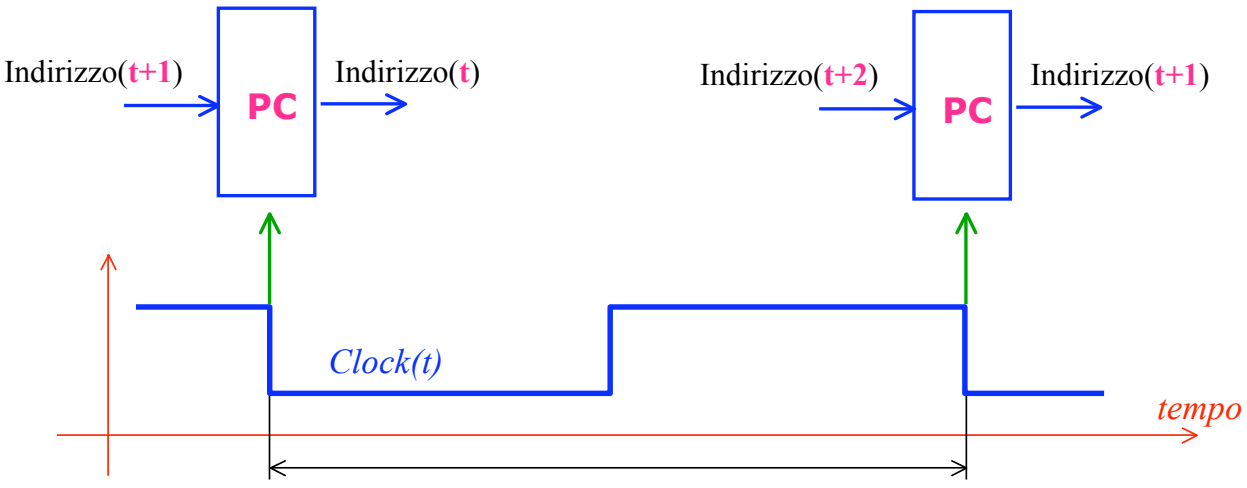
- Doppia porta di Lettura
  - il dato in lettura è sempre disponibile
  - non serve un comando dalla UC
- Scrittura:
  - Comando esplicito (**W – write**) dall'unità di controllo.





CPU singolo ciclo: temporizzazione

CPU singolo ciclo = 1 istruzione in 1 ciclo di clock



T_C > tempo necessario per eseguire il cammino critico

Sommario

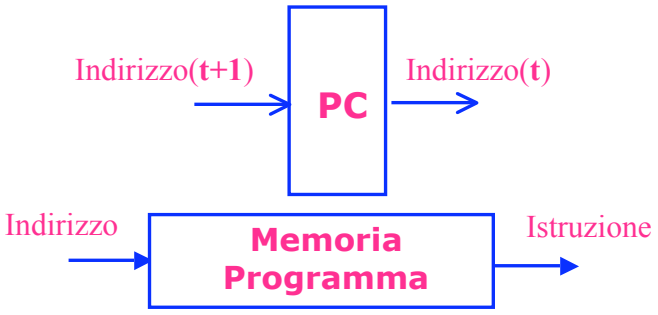
- La CPU
- Sintesi di una CPU per le istruzioni di tipo R
- Sintesi di una CPU per le istruzioni di tipo I (memoria)
- Sintesi di una CPU per le istruzioni di tipo I (salti)
- CPU che gestisce istruzioni: lw/sw e branch

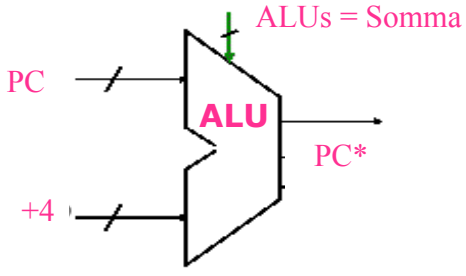
Fase di *FETCH*

- **FETCH**:
  - Memorizzare l'indirizzo dell'istruzione nel Program Counter.
  - Leggere l'istruzione dalla memoria di programma.
  - Aggiornare l'indirizzo in modo che in PC sia contenuto l'indirizzo dell'istruzione successiva:

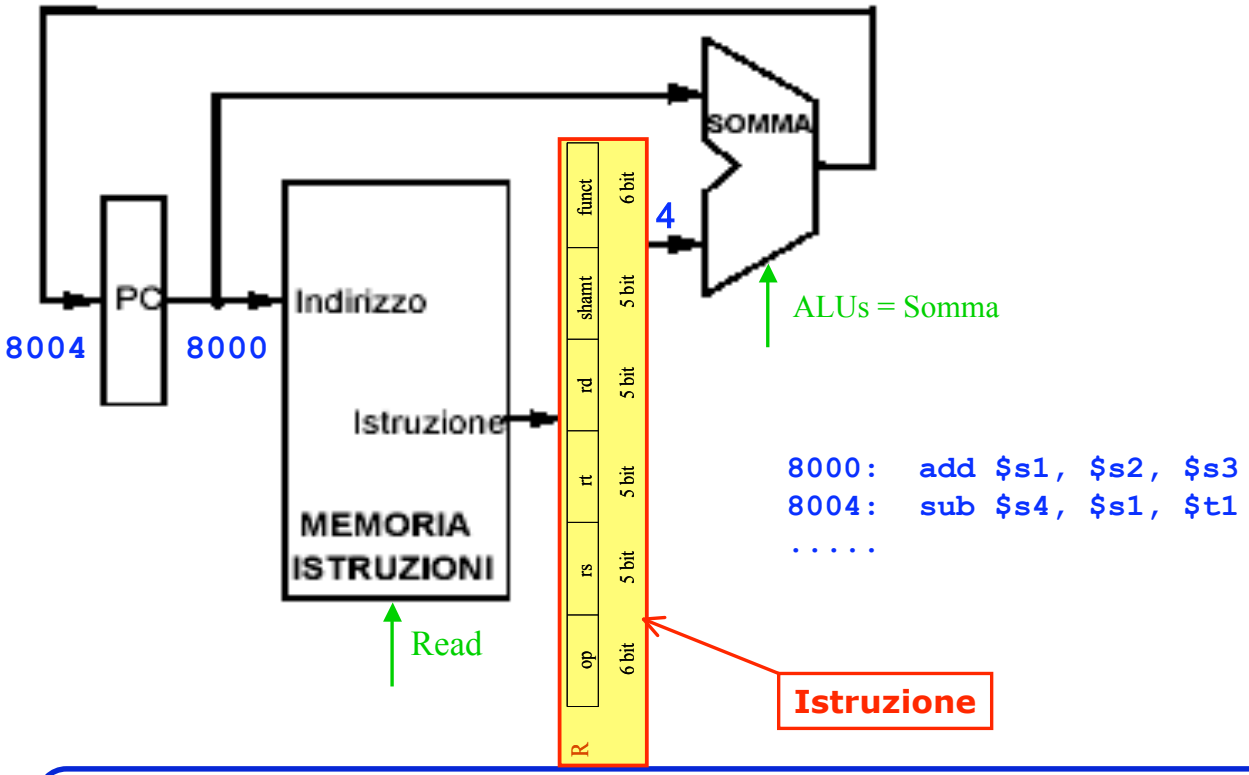
$$PC \leftarrow PC + 4$$

  - Dispositivi coinvolti:

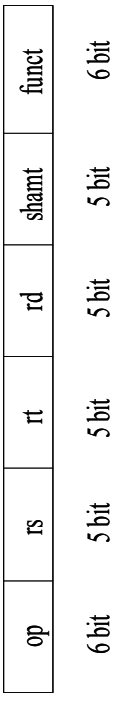




Circuito della fase di fetch

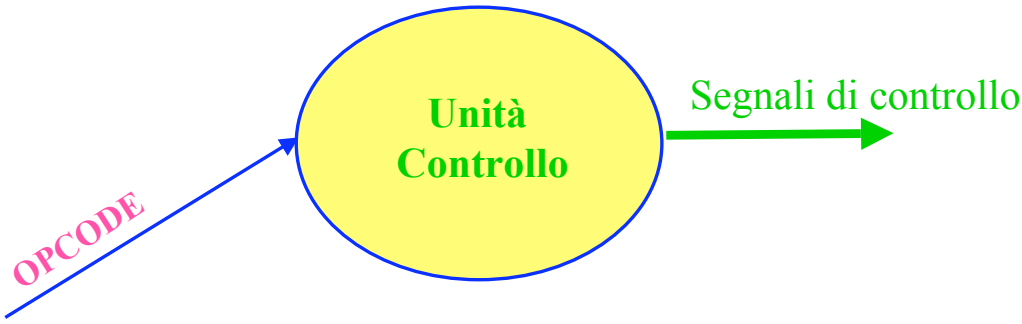


Fase di **decodifica**

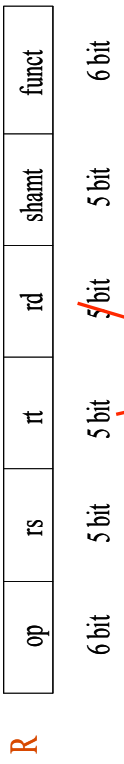


DECODIFICA:

- Leggo l'istruzione e genero i segnali di controllo opportuni.
- Leggo il contenuto dei registri.



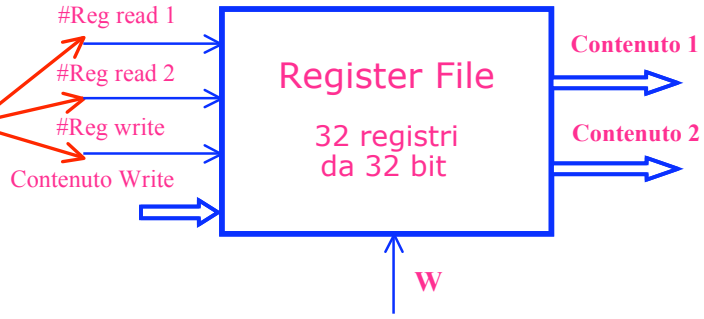
Lettura dei registri (istr. tipo R)



ESECUZIONE:

1) Leggo l'istruzione e genero i segnali di controllo opportuni.

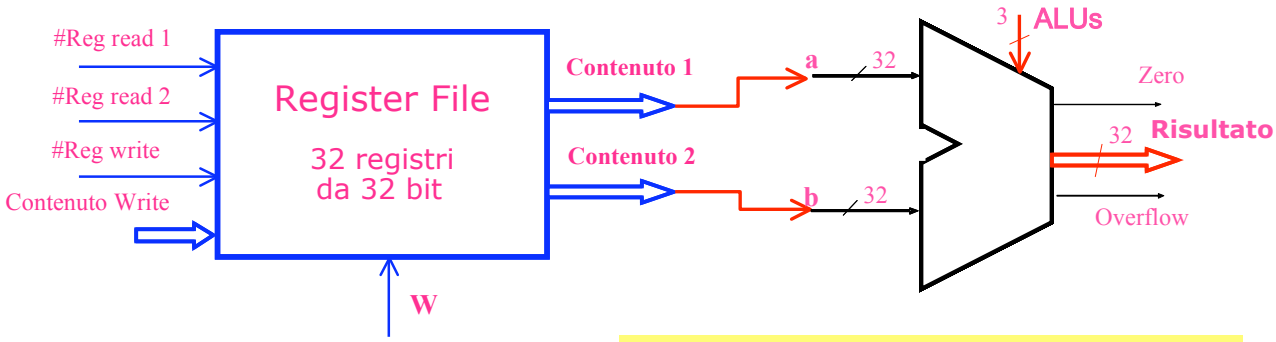
2) Leggo il contenuto dei registri.




```
8000    add $s1, $s2, $s3
8004    sub $s4, $s1, $t1
.....
```


Fase di **Esecuzione** (tipo R)

- I dati vengono letti dal Register File e immessi nella ALU
  - Necessità della doppia porta di uscita del Register File
- La ALU viene comandata in modo opportuno (**ALU$_S$**)

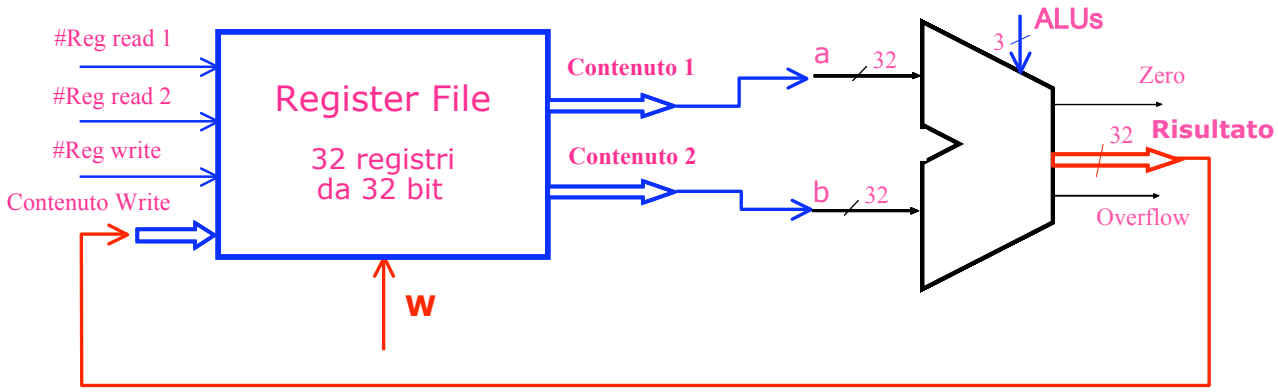



```
8000    add $s1, $s2, $s3
8004    sub $s4, $s1, $t1
.....
```

Fase di **Write back** (tipo R)

Write-back:

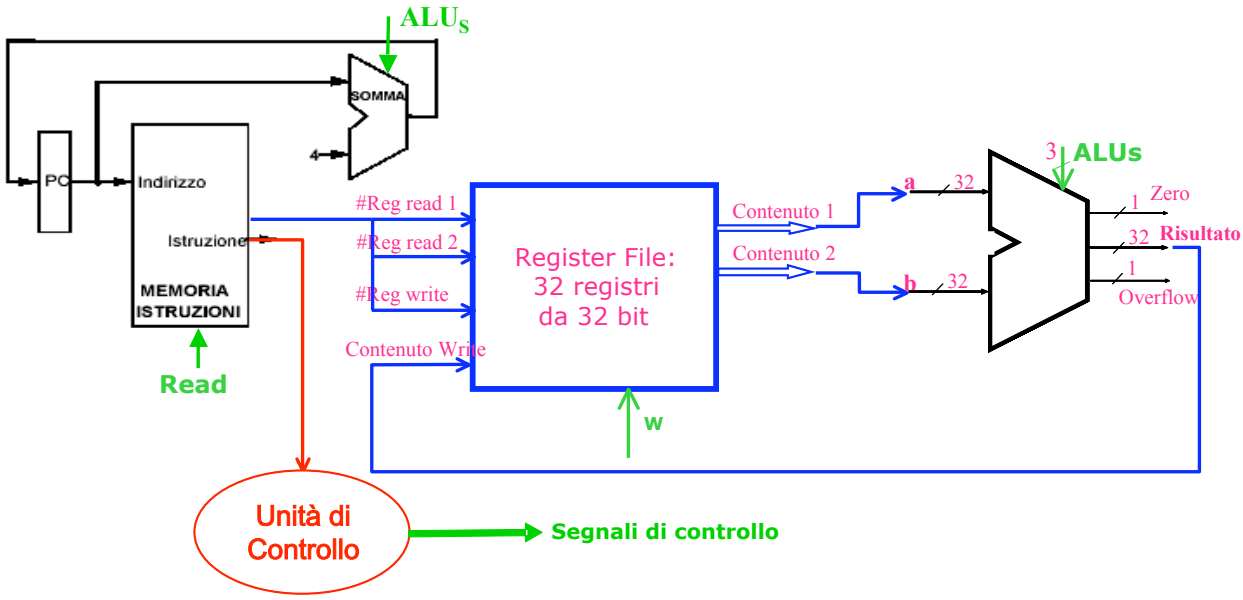
- La ALU ha eseguito l'operazione (ALU_S)
- Il risultato viene memorizzato nel register file
  - Comandi: Write (**W**), #reg_write



CPU per l'esecuzione di un'istruzione di tipo **R**

Tipo R

op	rs	rt	rd	shamt	funct
6 bit	5 bit	5 bit	5 bit	5 bit	6 bit

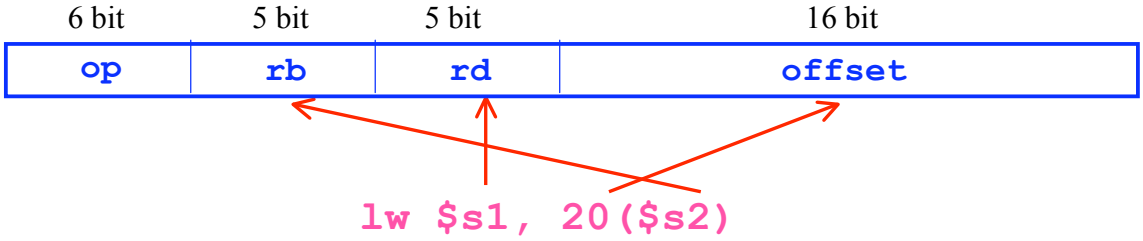


Sommario

- La CPU
- Sintesi di una CPU per le istruzioni di tipo R
- Sintesi di una CPU per le istruzioni di tipo I (memoria)
- Sintesi di una CPU per le istruzioni di tipo I (salti)
- CPU che gestisce istruzioni: lw/sw e branch

Istruzioni di tipo I – memoria: `lw/sw`

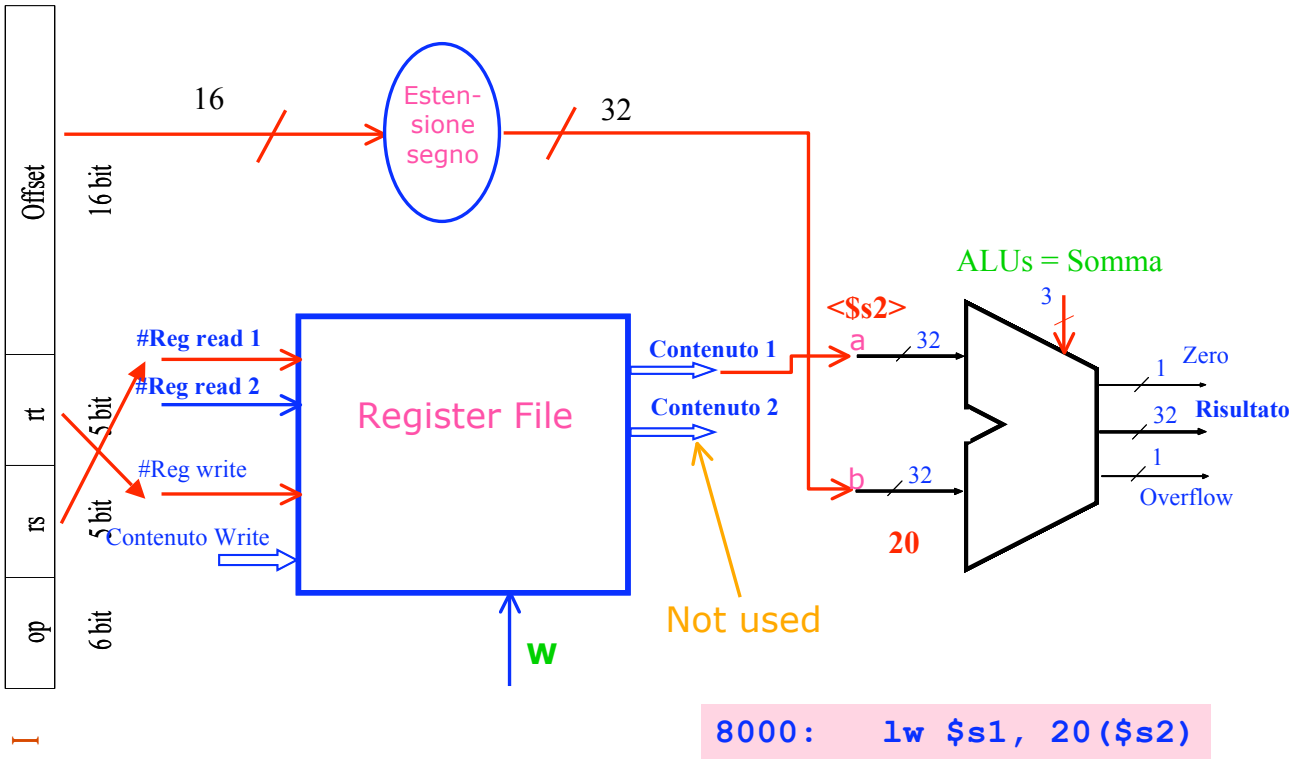
Tipo (formato) I



L'indirizzo di memoria sarà:

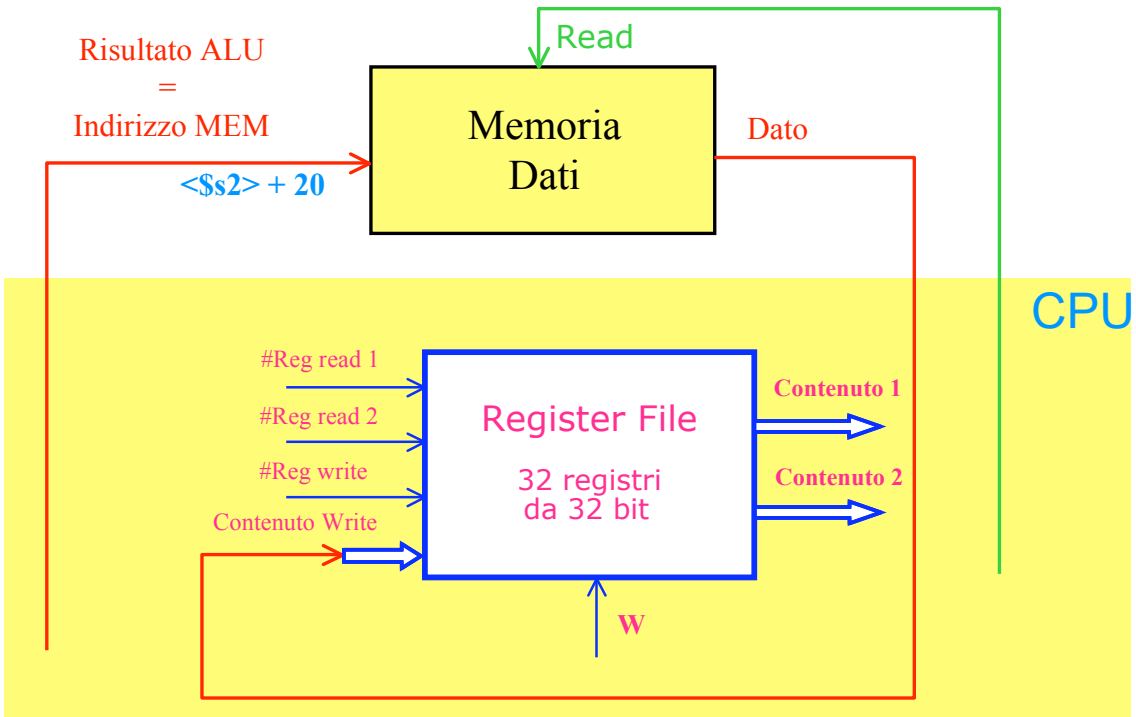
Base ($s2)	0100 1000 0011 0001 1011 1011 1011 1011 +
+ Offset	0000 0000 0001 0100 +
= indirizzo:	0100 1000 0011 0001 1011 1011 1100 1111

Fase di esecuzione (tipo I: `lw`)

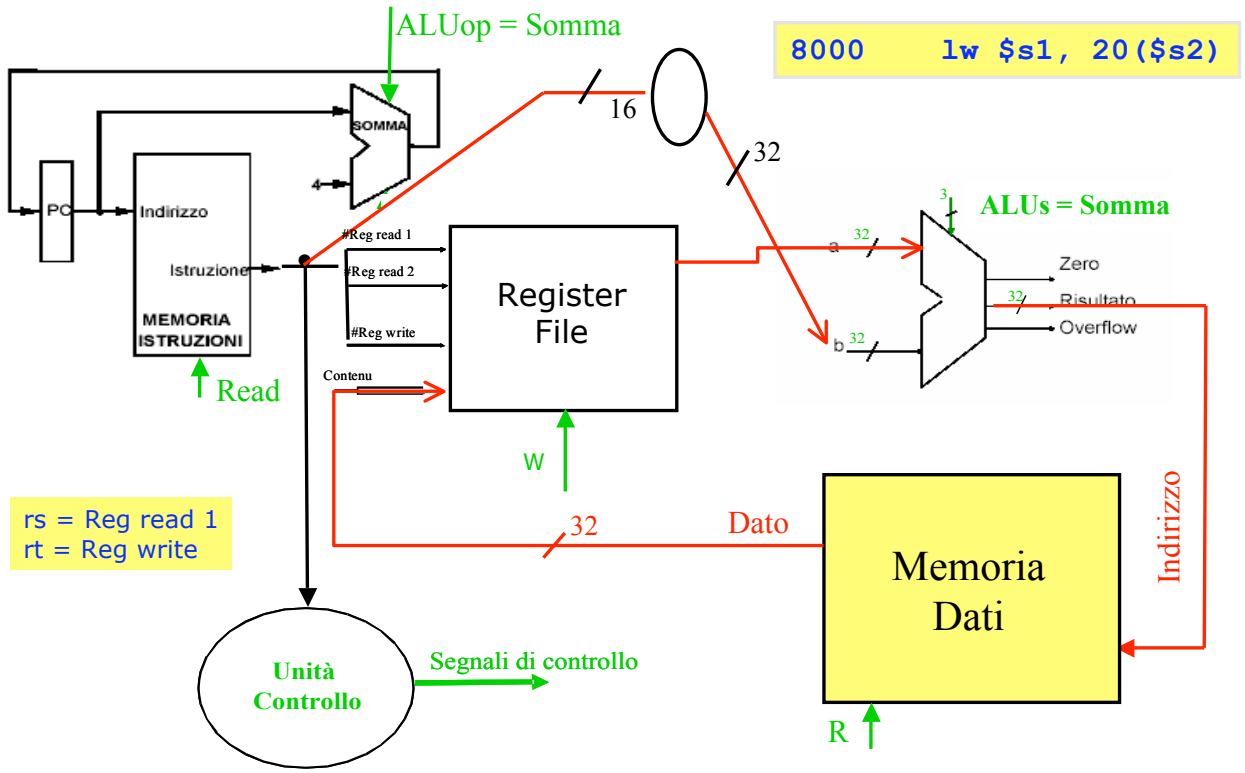



```
8000:   lw $s1, 20($s2)
```

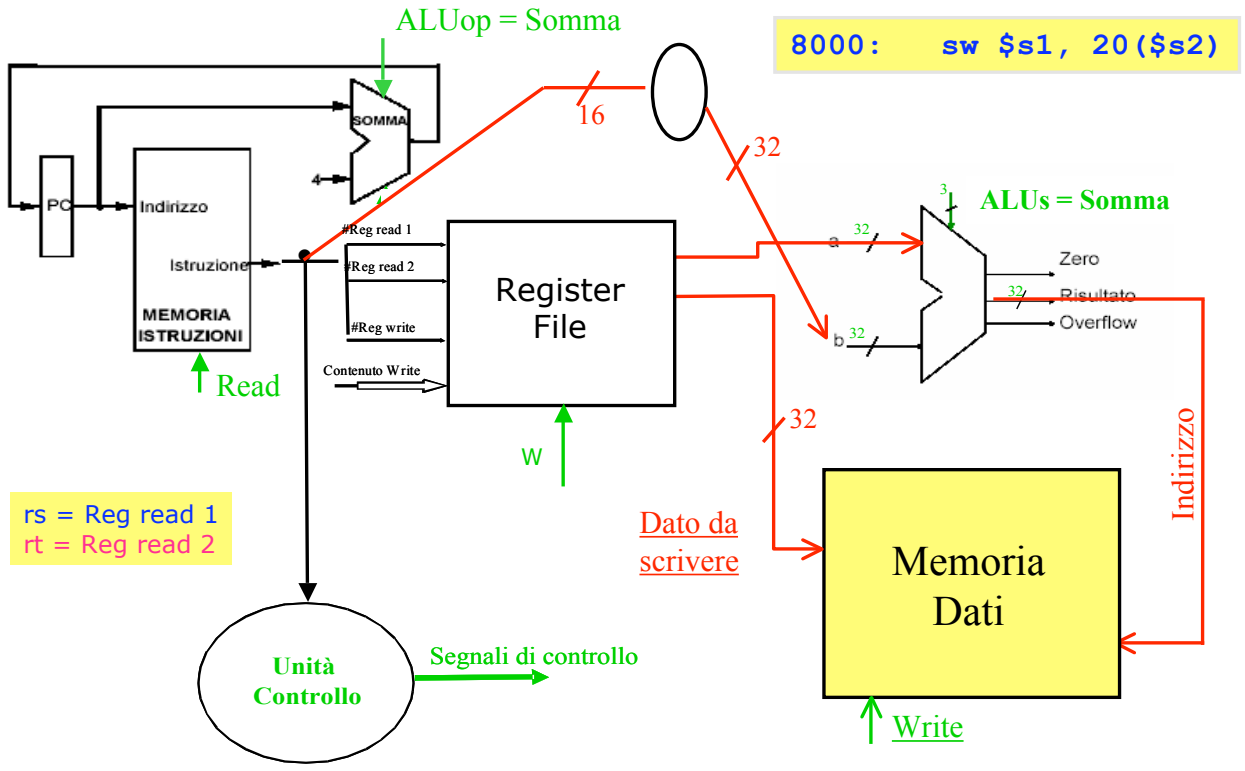

Lettura della memoria: write-back



CPU per l'esecuzione di una `lw`



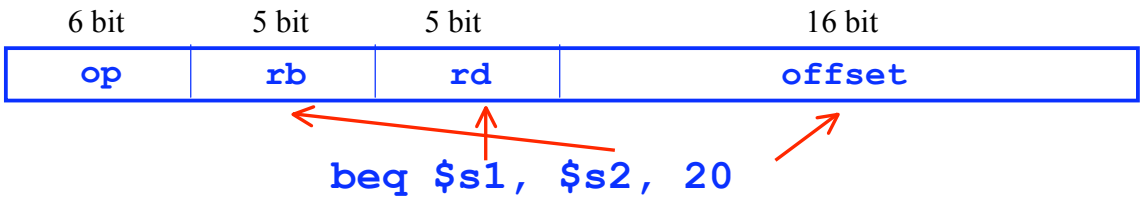
CPU per l'esecuzione di una `sw`



Sommario

- La CPU
- Sintesi di una CPU per le istruzioni di tipo R
- Sintesi di una CPU per le istruzioni di tipo I (memoria)
- Sintesi di una CPU per le istruzioni di tipo I (salti)
- CPU che gestisce istruzioni: R, lw/sw, branch

Istruzioni di tipo I: **branch**



L'indirizzo di salto sarà determinato in due passi:

A) Calcolo dello spiazzamento in byte: Offset * 4 20: 10100 → 1010000

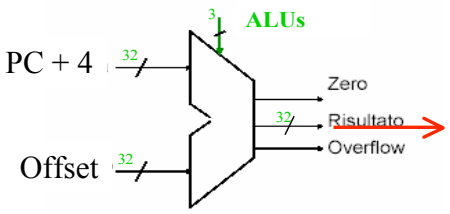
B) Deteminazione dell'indirizzo di salto come:

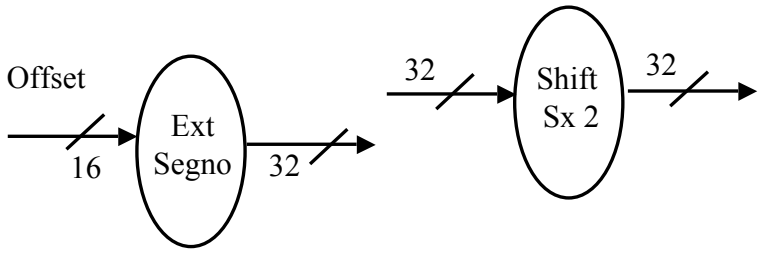
Base (PC)	0100 1000 0011 0001	1011 1011 1011 1011 +
Offset		0000 0000 0101 0000 +
Indirizzo salto	0100 1000 0011 0001	1011 1011 1100 1011

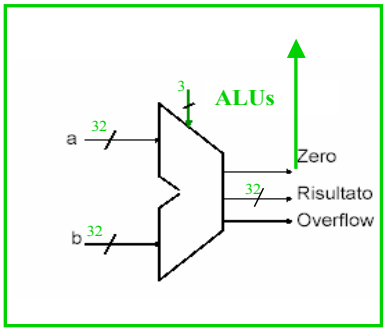
1) **BRANCH**: Determinare se l'uguaglianza è vera
2) Calcolare l'indirizzo di salto.

Fase di esecuzione della beq

1) Estensione dell'offset su 32 bit.
2) Moltiplicazione per 4 dell'offset.
3) Somma del PC con l'estensione del segno.
4) Controllo se il contenuto dei registri è uguale.

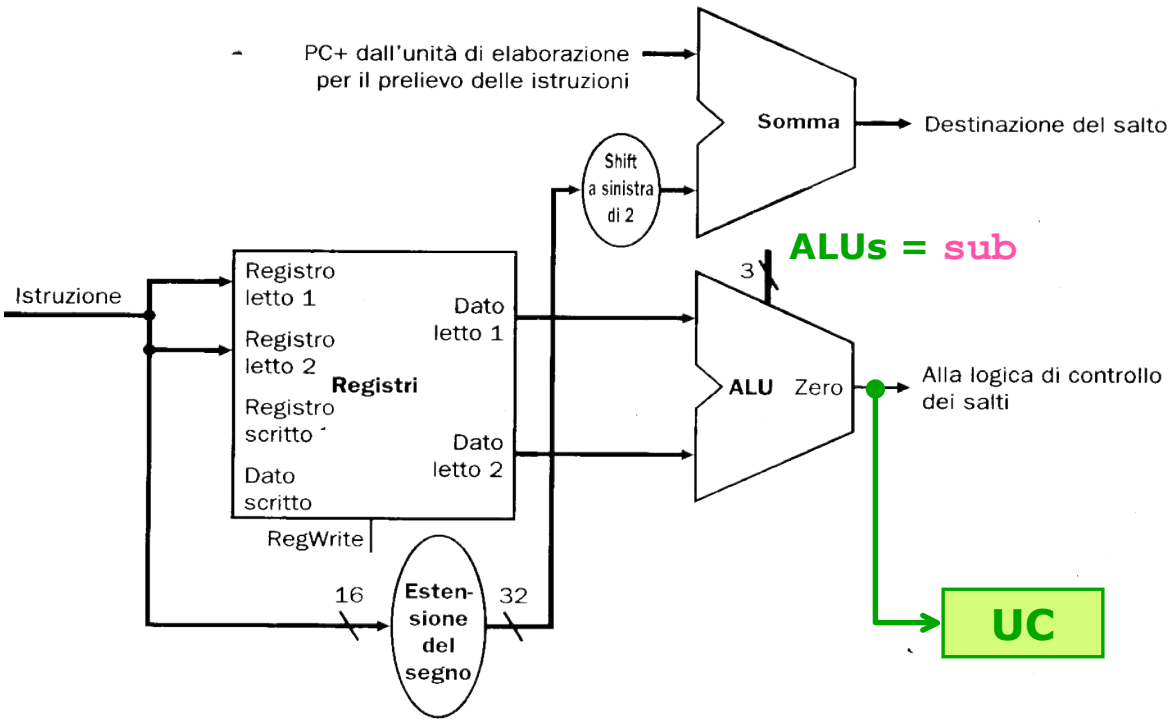






000100	10001	10010	0000 0000 0000 0101
OpCode	**rs**	**rt**	**Offset**

Circuito di esecuzione: beq



Sommario

- La CPU
- Sintesi di una CPU per le istruzioni di tipo R
- Sintesi di una CPU per le istruzioni di tipo I (memoria)
- Sintesi di una CPU per le istruzioni di tipo I (salti)
- CPU che gestisce istruzioni: tipo R, lw/sw, branch

ISA MIPS – tipi di istruzione: R, I (`lw/sw,branch`)

- I tre formati delle istruzioni sono:

31-26	25-21	20-16	15-11	10-6	5-0
op	rs	rt	rd	shamt	funct

31-26	25-21	20-16	15-0
35/43	rs	rt	offset

31-26	25-21	20-16	15-0
4	rs	rt	indirizzo

- Campo op sempre contenuto nei primi 6 bit (in seguito, Op[5-0])
- Registri da leggere sempre rs e rt, di posizione 25-21 e 20-16
- Registro base per lettura/scrittura sempre rs, di posizione 25-21
- Offset sempre in posizione 15-0
- Registro destinazione rd (15-11) per istruzioni di tipo R, rt (20-16) per istruzioni lettura (necessario multiplexer)

CPU singolo ciclo: caratteristiche

- **CPU a singolo ciclo:** il ciclo di esecuzione di un'istruzione si compie in un unico ciclo di clock.
- tutte le unità funzionali coinvolte in un'istruzione vengono utilizzate contemporaneamente
- Ogni unità funzionale può essere utilizzata 1 sola volta durante un'istruzione.
- Duplicazione Memoria: Memoria dati e memoria istruzioni.
- Triplicazione ALU: 2 sommatori + 1 general purpose.
- Introduzione di multiplexers.

Schema generale

- Esegue: accesso a memoria: ***lw/sw,***
  aritmetico-logiche ***tipo R,***
  branch: ***beq***