


Architettura degli elaboratori lezione 11


 Appunti di Davide Scarlata 2024/2025

 **Prof:** Claudio Schifanella

 **Mail:** claudio.schifanella@unito.it

 **Corso:** C

 [Moodle Unito](#)

 **Data:** 15/04/2025

b

Buffer (non) invertente

Circuito semplice che ha input, un output, un input di controllo e tre stati, ovvero :

- Quando il segnale di controllo vale 1, il circuito si chiude e l'input scorre fino all'output (valendo quindi 0 o 1).
- Quando il segnale di controllo vale 0, il circuito si apre e l'output (a prescindere dal valore dell'input), non ha un segnale (terzo stato).

Può essere utile nel caso in cui abbiamo vari un registro (formato da vari flip-flop D) e vogliamo disconnetterli (o connetterli)

Example

| In verde l'enable comune che permette di far passare o meno un dato

- Read register number 1 : che si occupa di leggere il primo registro selezionato. Legge 5 bit (sempre per il discorso che abbiamo 32 registri e quindi bastano 5 bit per specificare un numero da 0 a 31).
- Read register number 2 : che si occupa di leggere il secondo registro selezionato. Uguale al 1 in pratica.
- Read data 1 : contiene il valore del registro 1 letto. In un'architettura a 32 bit (quella che stiamo vedendo), restituisce 32 bit.
- Read data 2 : uguale a read data 1, ma contiene il valore del registro 2 letto.
- Write register : serve per specificare l'indirizzo dove voglio andare a scrivere l'eventuale output della ALU. Legge sempre 5 bit.
- Write data : serve per scrivere qual è il valore che voglio scrivere nel registro selezionato. Riceve 32 bit.
- Write (segnale di controllo) : serve per dire che vogliamo scrivere un dato in un registro. È solo 1 bit. Ci serve perché non tutte le istruzioni modificano un valore nei registri (ad esempio una sw, non salva nulla in alcun registro, però abbiamo comunque bisogno di accedere ai registri).

Example

Quindi, per chiarire, prendiamo questo esempio :

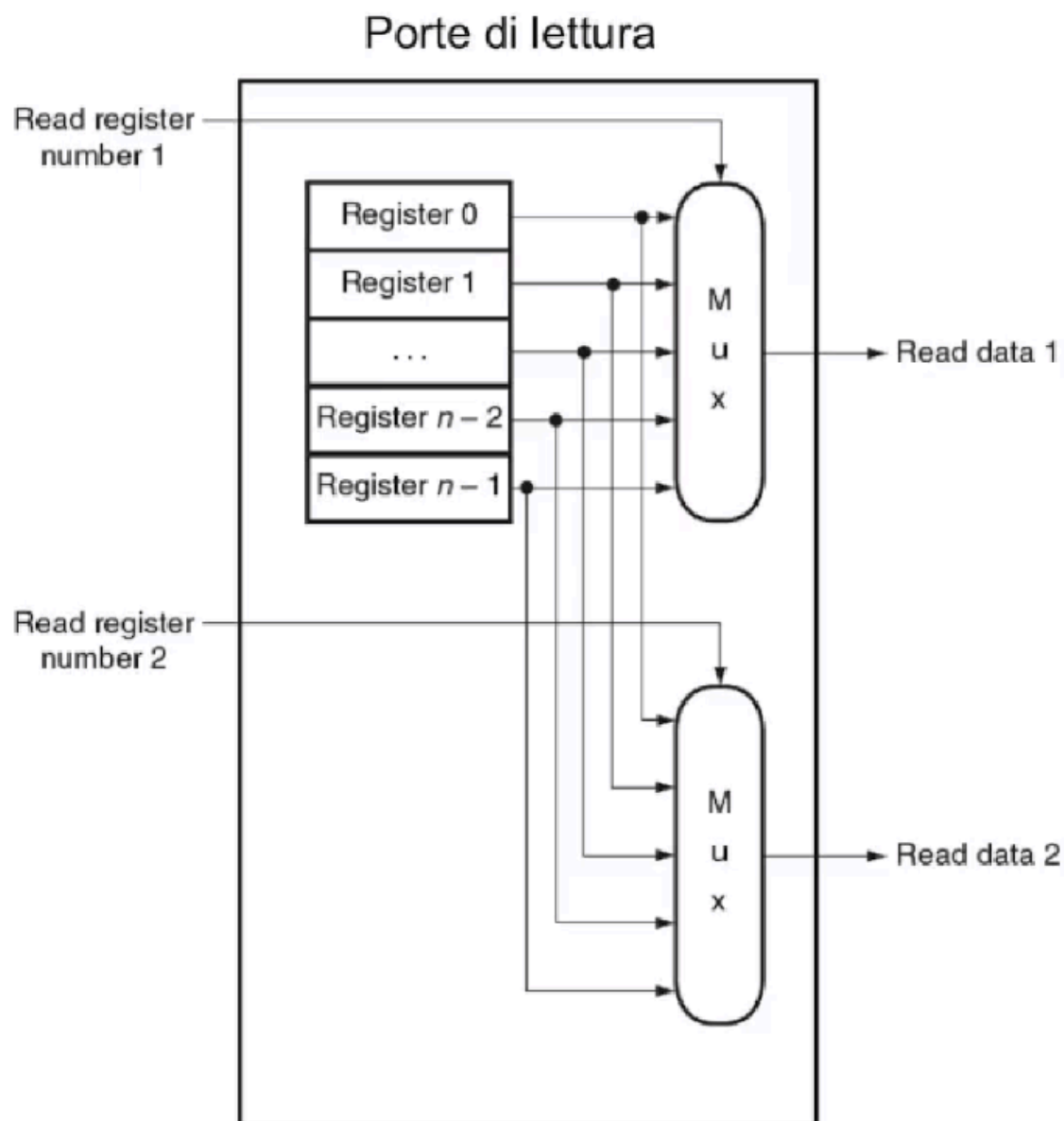
```
add x2, x4, x5
```

Gli step sono :

- Per prima cosa leggiamo tramite Read register number 1 e 2 i due registri (x4 e x5)
- Passiamo i loro valori, tramite "read data" 1 e 2, all'ALU che li somma
- L'ALU ritorna a "write data" il valore sommato
- Viene passato "x2" a "write register" per sapere dove dobbiamo scrivere il risultato della somma
- Impostiamo "write" ad 1
- Salviamo il valore ottenuto dalla ALU in x2.

Selezionamento dei registri

Abbiamo detto sopra che Read register number 1/2 selezionano dei registri, ma non come. Per selezionare dei registri quindi, si usa un multiplexer (lezione 9, un circuito combinatorio che dati 2^n input, permette di selezionare in uscita 1 solo di questi input avendo n ingressi di controllo)

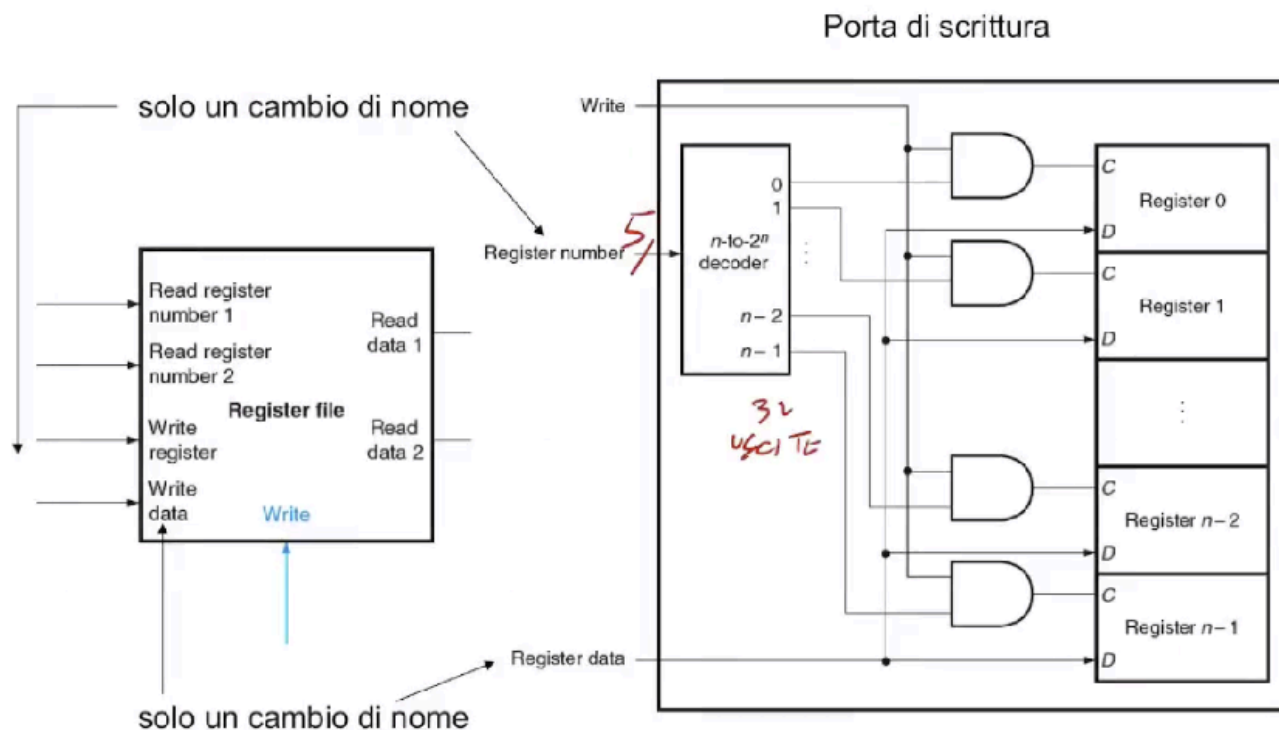


Se passiamo quindi al multiplexer 1, il valore "1" questo farà uscire da "read data 1" il valore contenuto in "register 1".

Scrittura di un registro

Abbiamo detto sopra che "write register" selezionano dei registri, ma non come. Per selezionare dei registri quindi, si usa un decoder (lezione 9, un circuito combinatorio che permette di scegliere uno di 2^n output, avendo n input). Al decoder dobbiamo aggiungere, come già detto, write. Se mettiamo in AND il segnale restituito dal decoder con write (per ogni

registro), riusciamo a scrivere solamente nel registro selezionato (se non mettessimo il decoder, scriveremmo su tutti i registri).



Tipi di memoria RAM

- SRAM : RAM statiche (flip-flop tipo D), estremamente veloci, utilizzate per realizzare le cache
- DRAM : RAM dinamiche (transistor con condensatore), vanno rinfrescati, offrono grandi capacità ma più lente