Architettura degli Elaboratori

Informatica per la Comunicazione Digitale

Università degli Studi di Milano

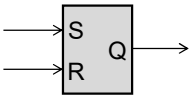
Lezione 9:

# Registri

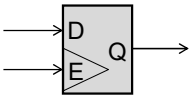
Marco Tarini

1

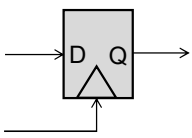
## Blocchi sequenziali per memorizzare un bit: riassunto



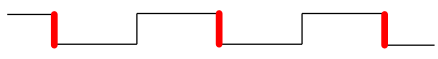

**SR-Latch** (**Bistabile SR**)
Asincrono




**Gated D-Latch** (**Bistabile D**)
Sincrono sul livello
Trasparente (per metà del ciclo)




**Flip-Flop**
Sincrono sul fronte
Sempre opaco





2

## Registri

- Chiaramente, l'informazione è fatta di dati non rappresentabili con 1 bit solo
  - Eccetto forse dati di tipo booleano (vero / falso)
- **Registro** = blocco funzionale
  - costituito da insieme di *n* componenti da un bit
  - memorizza una «*parola*» (o «*word*») di *n* bit
- Esistono tipi diversi di registro. Ne vediamo due:
  - Registro a scorrimento
  - Registro parallelo
- Caratteristiche comuni a tutti i registri:
  - **Scrittura**: sincronizzato sul fronte
    cioè, il registro assume un nuovo
    valore *solo all'inizio/fine di ogni ciclo di clock*
  - **Lettura**: asincrona, e costante.
    Cioè, per tutto il ciclo di clock, il registro presenta in uscita
    il word attualmente memorizzato

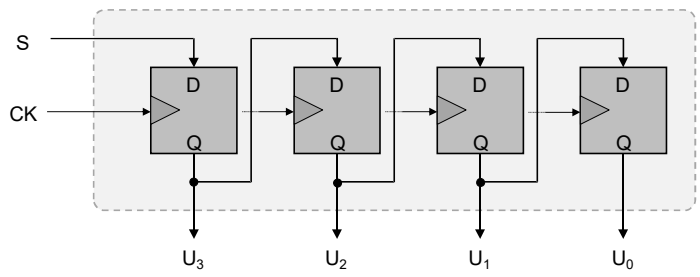
il tipo più comune; spesso, si omette di specificare «parallelo»

per noi: quello di discesa (scelta arbitraria)

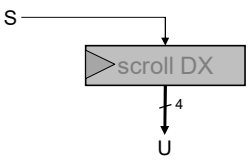


3

## Esempio: registro a scorrimento a 4 bit




implementazione




simbolo funzionale



4

## Registro a scorrimento (a n bit)

- Il registro a scorrimento a n bit ha:
  - un ingresso S (1 bit)
  - n uscite parallele $U_{n-1}$ , … , $U_0$
  - oltre a naturalmente l'ingresso di clock (per la sincronizzazione)
- Ad ogni ciclo di clock,
  fa scorrere di un bit verso destra la parola memorizzata (right shift),
  e il bit più a sinistra (il MSB) assume il valore di S
- Uso in scrittura: comporre il word da memorizzare un bit alla volta
  - Analogia: come in un dettato, o nello spelling di una parola
  - Per comporre un word servono *n* cicli di clock
- Uso in lettura (come per qualsiasi registro):
  - l'intera parola U è costantemente presentata in uscita
- Considerazioni:
  - Vantaggio: un solo canale di input, circuiti più semplici
  - Svantaggio: lento in scrittura



5

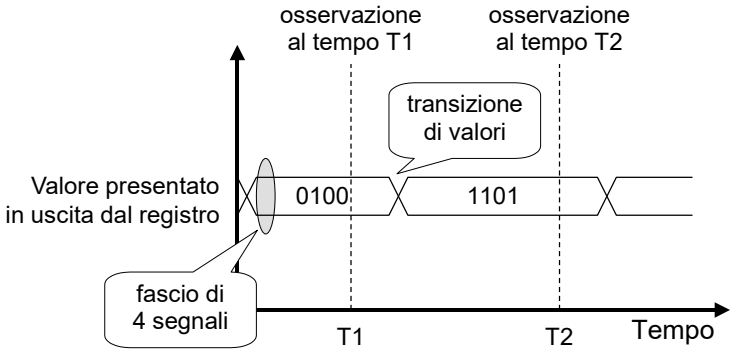
## Registro a scorrimento (a n bit) implementazione

- Costituito da n flip-flop collegati in cascata
  - Input del primo = il singolo segnale in ingresso S
  - input di tutti gli altri = output del precedente
- Osserva l'implementazione:
  se si usassero dei bistabili D (sincronizzati sul livello)
  invece di flip-flop (sincronizzati sul fronte),
  durante il livello alto del clock il bit S si propagherebbe per tutta la catena
  fino a dove arriva allo scadere del ciclo (che è difficile capire quando avviene)
  … disastro!
  - La sincronizzazione sul fronte è sempre necessaria!



6

## Diagramma temporale: come rappresentare l'uscita di un registro (o, in generale, di **un fascio** di segnali)

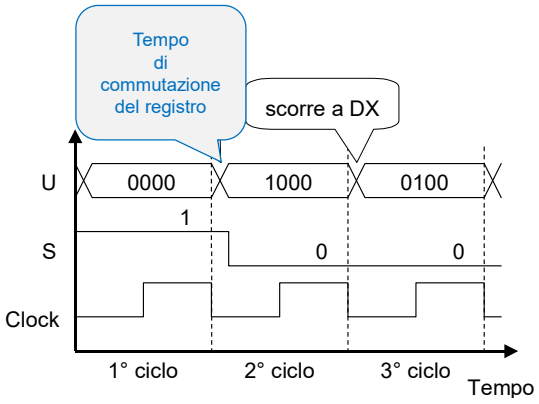



- Al tempo T1 i 4 segnali valgono 0100
  (se la parola nel registro rappresenta un intero, quel registro vale «4»)
- Al tempo T2 i 4 segnali valgono 1101
  (la parola memorizzata nel registro vale «13», oppure «-2» se è in CP2)



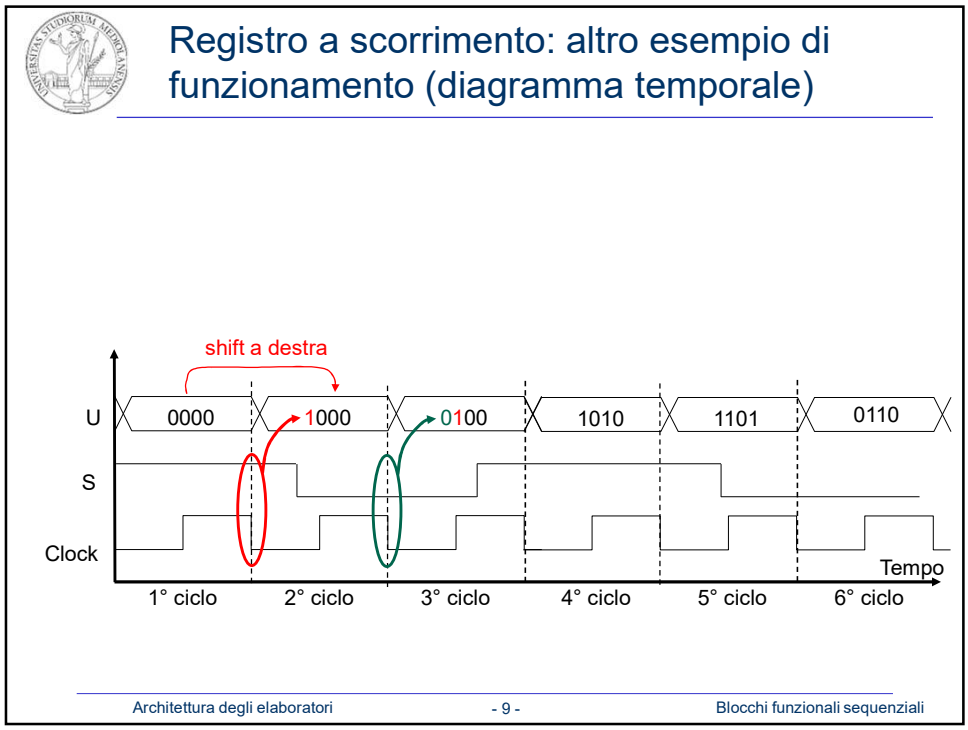
7

## Registro a scorrimento: esempio di funzionamento (attraverso il diagramma temporale)

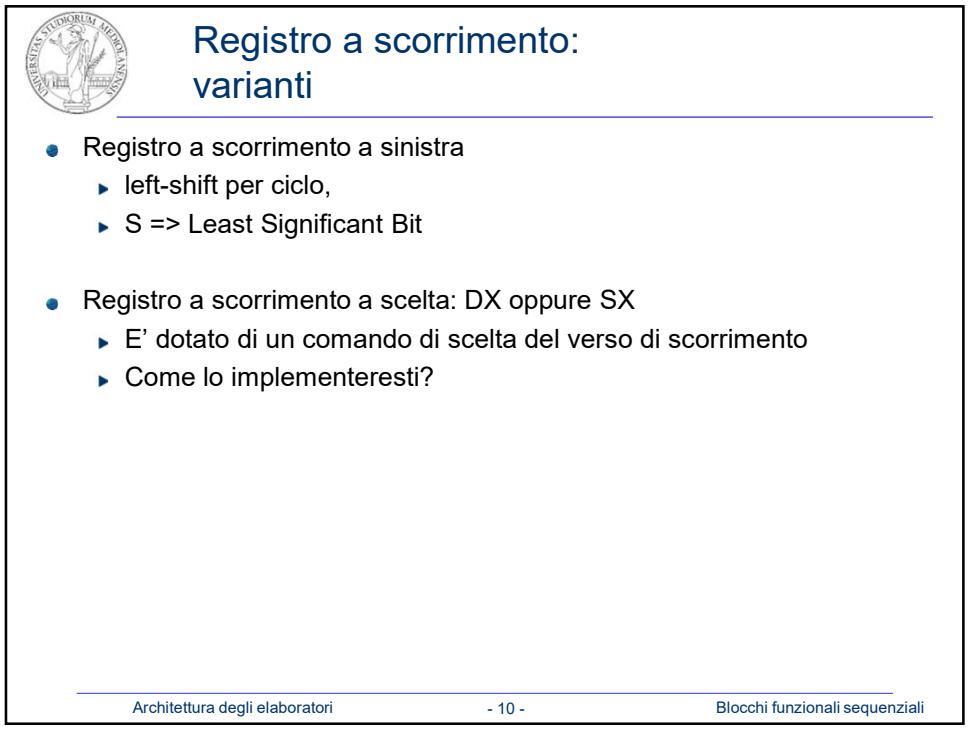





8


Registro a scorrimento: altro esempio di funzionamento (diagramma temporale)
shift a destra
U
S
Clock
0000
1000
0100
1010
1101
0110
Tempo
1° ciclo
2° ciclo
3° ciclo
4° ciclo
5° ciclo
6° ciclo
Architettura degli elaboratori
- 9 -
Blocchi funzionali sequenziali

9


Registro a scorrimento: varianti
Registro a scorrimento a sinistra
left-shift per ciclo,
S => Least Significant Bit
Registro a scorrimento a scelta: DX oppure SX
E' dotato di un comando di scelta del verso di scorrimento
Come lo implementeresti?
Architettura degli elaboratori
- 10 -
Blocchi funzionali sequenziali

10

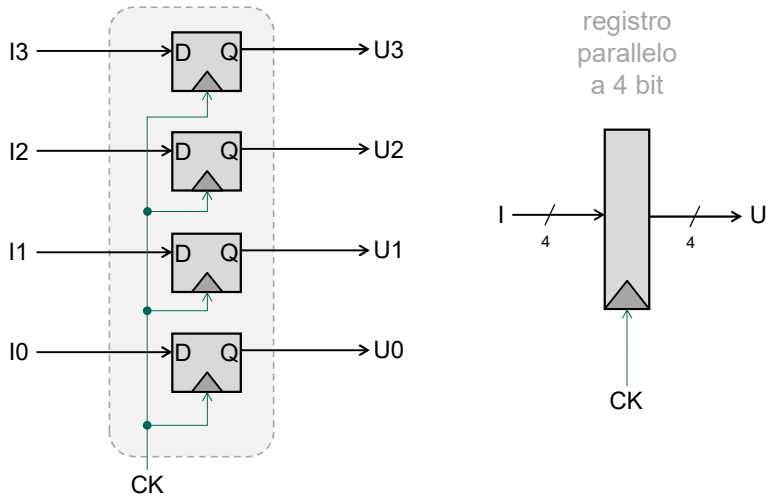
## Registro parallelo (il tipo standard di registro)

- Anche il registro parallelo è costituito da $n \geq 1$ flip-flop
- Ha:
  - $n$ ingressi $I_0, \ldots, I_{n-1}$
  - $n$ uscite $U_0, \ldots, U_{n-1}$
  - ingresso di clock CK (per sincronizzarsi naturalmente!)
- All'inizio di ogni ciclo di clock, il registro legge e memorizza nel suo stato la parola di $n$ bit presente in ingresso
  - tutta insieme, non un bit alla volta come nel caso del reg. a scorrimento
- Come per tutti i registri:
  - In ogni momento, il registro espone sulle $n$ uscite il valore attuale della parola di $n$ bit memorizzata all'ultimo fronte di discesa del clock
- Anche in questo caso, è necessaria la **sincronia sul fronte** (di discesa)
  - Nota: se si usassero dei bistabili D sincronizzati sul livello, allora durante il livello alto del clock il registro sarebbe esso stesso trasparente.



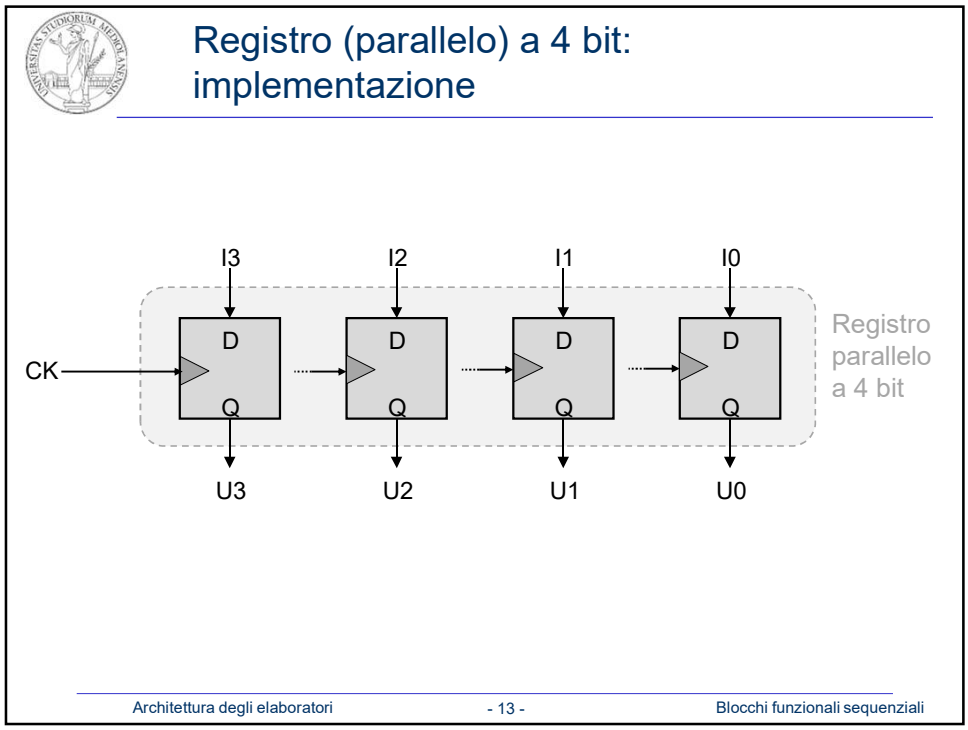
11

## Registro (parallelo) a 4 bit: implementazione






12


Registro (parallelo) a 4 bit: implementazione
I3
I2
I1
I0
D
Q
CK
U3
U2
U1
U0
Registro parallelo a 4 bit
Architettura degli elaboratori
- 13 -
Blocchi funzionali sequenziali

13

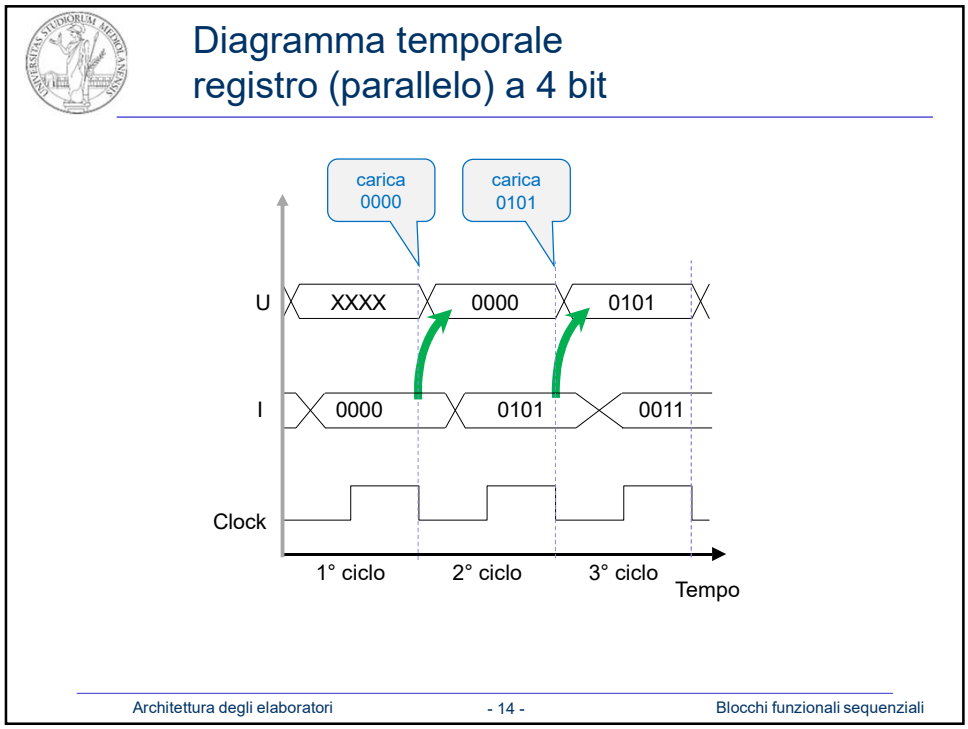


Diagramma temporale registro (parallelo) a 4 bit
carica 0000
carica 0101
U
XXXX
0000
0101
I
0000
0101
0011
Clock
1° ciclo
2° ciclo
3° ciclo
Tempo
Architettura degli elaboratori
- 14 -
Blocchi funzionali sequenziali

14

## Dimensione del registro (note)

- Definizione: in una data architettura (e in un istruction set), una «parola» («word») è un valore memorizzato in *n* bit, dove *n* è la dimensione dei registri usati in una data architettura
- Questo numero *n* (la lunghezza di una parola) è una scelta fondamentale che tipicamente coinvolge tutta l'architettura.
- Ad esempio:
  - **E' la dimensione dei registri**
  - E' la dimensione degli operandi, e del risultato, della ALU
  - E' la dimensione della lunghezza dei comandi dell'Instruction Set (anche se, in alcuni IS, alcuni comandi sono scritti su più parole)
  - E' la dimensione degli *indirizzi di memoria* (ma se non sempre) (vedi lezioni successive)
- Per esempio, quando diciamo (correttamente) che *«MIPS è un architettura a 32 bits»* stiamo dicendo che *«i registri di una macchina MIPS sono composti di 32 Flip-Flop»*, etc



15

## Dimensione del registro (note)

La dimensione dei registri, e quindi delle parole (word), è una scelta fondamentale nella progettazione di un'architettura

Scelte comuni:

- Architetture "a 32 bits"
  - MIPS
  - Apple Mac "Core Duo"
  - Intel x86 (i286, i386, i486…)
- Architetture "a 64 bits"
  - Apple Core 2 Duo, Xeon, i3, i5, i7, …
  - Intel x64, AMD 64 …
- Architetture "a 8 bits"
  - Intel 8080, Zilog Z80, NES, … e altre vecchie glorie dei videogames anni '80 (e oltre)



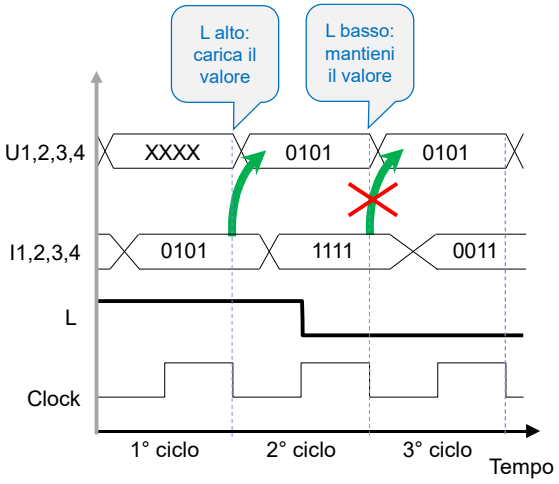
16

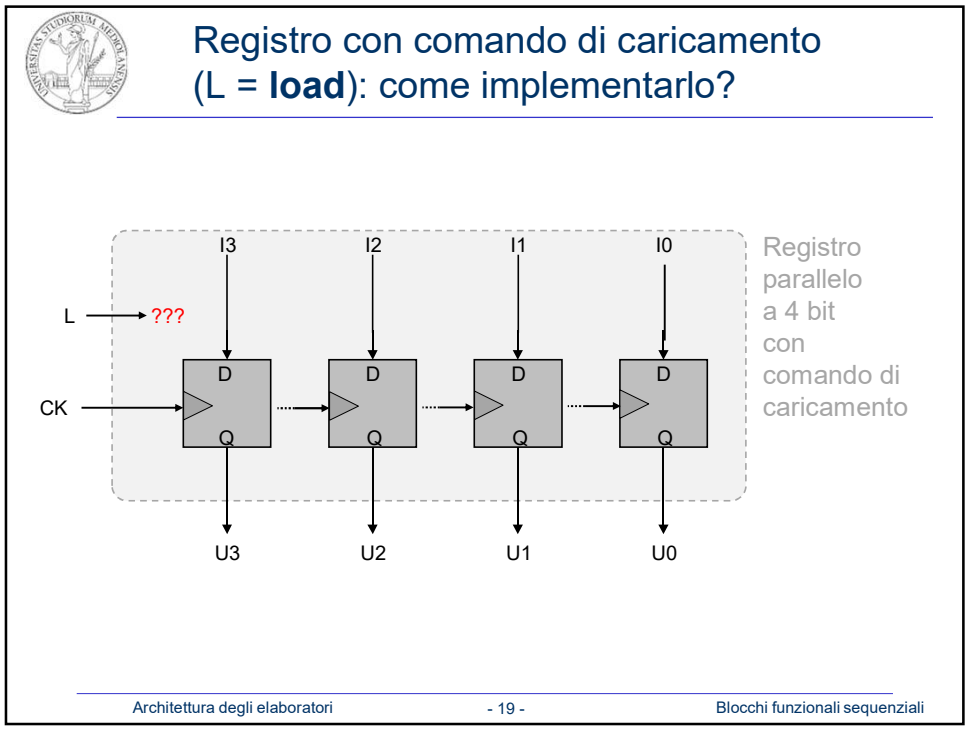
## Registro parallelo con comando di caricamento

- Opzionalmente, possiamo aggiungere in ingresso un comando di **Caricamento** (o **Load**) di un bit:
- controlla se il registro è attivo in scrittura.
  - Quando **L** è attivo (**L** = 1): la parola in ingresso al registro viene memorizzata come normale nel registro, allo scattare del ciclo di clock
  - Altrimenti (**L** = 0), il registro mantiene il suo valore memorizzato corrente anche allo scoccare del ciclo di clock, «non ascoltando» l'input

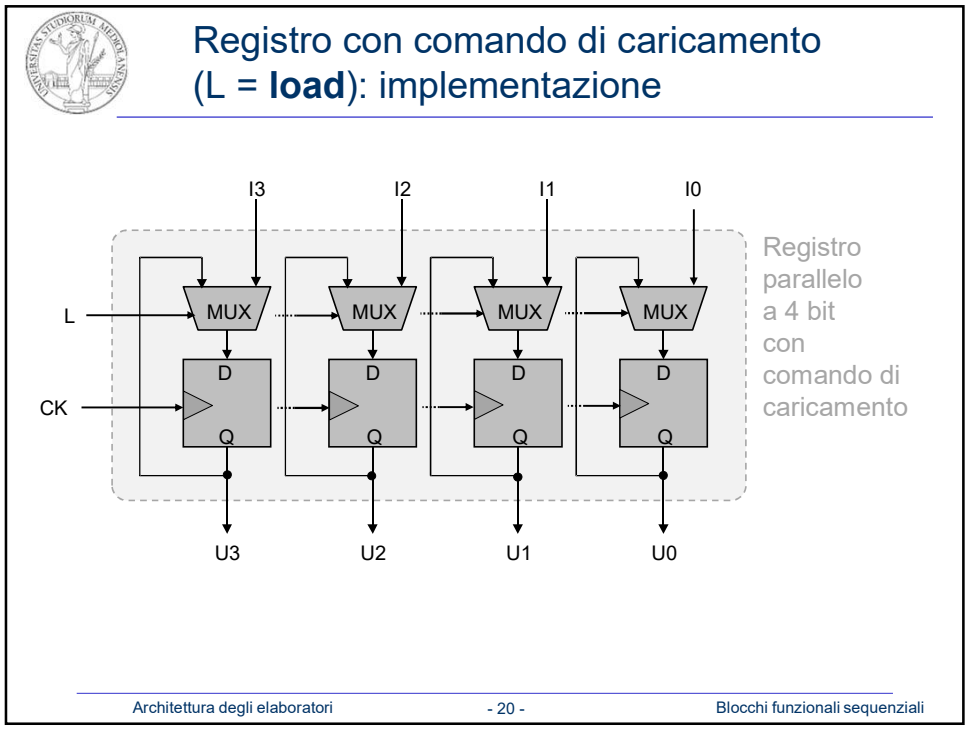


## Registro con comando di caricamento: diagramma temporale




Registro con comando di caricamento (L = **load**): come implementarlo?
I3
I2
I1
I0
L
???
CK
D
Q
U3
U2
U1
U0
Registro parallelo a 4 bit con comando di caricamento
Architettura degli elaboratori
- 19 -
Blocchi funzionali sequenziali

19


Registro con comando di caricamento (L = **load**): implementazione
I3
I2
I1
I0
L
MUX
CK
D
Q
U3
U2
U1
U0
Registro parallelo a 4 bit con comando di caricamento
Architettura degli elaboratori
- 20 -
Blocchi funzionali sequenziali

20

## Altri comandi in ingresso opzionali

- Ogni comando è un ulteriore bit di ingresso
  - Comando di **Riprisinto** (o **Clear**, o **Reset**)
    se attivato, azzera il registro, mettendolo a tutti 0.
  - Comando di **Precarica** (o **Preset**)
    se attivato, mette tutto il registro a 1
- Questi comandi
  - Possono essere aggiunti a qualsiasi registro (anche a scorrimento)
  - Utilizzo: inizializzare il registro!
  - Possono essere implementati come sincroni o asincroni
  - Se **asincorni**: appena il segnale è attivato (messo a 1),
    il registro viene sovrascritto (senza aspettare il clock)
  - Se **sincroni**: il segnale ha effetto sul fronte del clock, come al solito.
  - Quando reset o preset sono attivi, il registro non memorizza il suo input
    (come se Load fosse 0)
- Come li implementeresti?



21

## Registro con comando di caricamento (L = **load**): come implementarlo? Note

- Nella soluzione vista, ogni flip-flop riceve in input:
  - Il valore I presentato in ingresso, quando L = 1.
  - Il valore già memorizzato in quello stesso flip-flop, quando L = 0.
    In questo caso, il nuovo valore coincide col vecchio, come corretto.

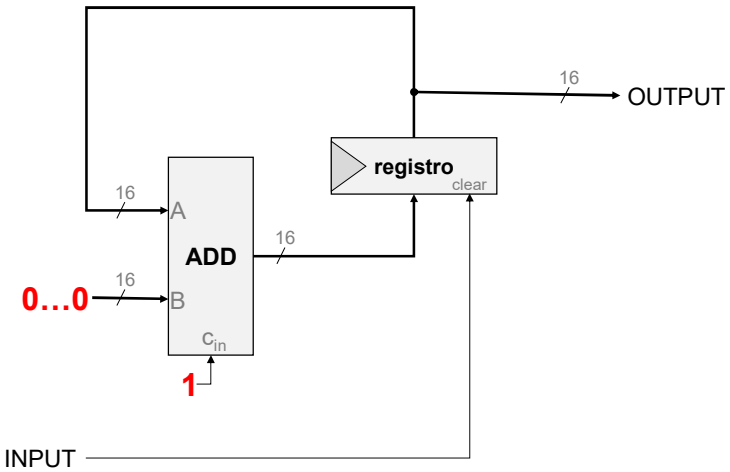
Soluzioni apparentemente valide, ma che non funzionano:

- Mettere L in AND con il segnale di Clock,
  cioè immettere in ogni flip-flop (L ∧ Clock) all'ingresso △:
  Anche se questo apparentemente annulla il segnale di clock,
  impedendo al flip-flop di cambiare valore, in realtà
  l'atto stesso di commutare L da 1 a 0 genera un fronte di discesa
  del segnale «L ∧ Clock» (durante l'intervallo alto del clock),
  facendo assumere al Flip-Flop il nuovo valore in D!
- Mettere L in AND con il segnale di data,
  cioè immettere in ogni flip-flop (L ∧ Input) all'ingresso D:
  In questo modo, L avrebbe solo l'effetto di annullare il dato in ingresso.
  L  quindi funziona come comando di RESET (sincrono), non di Load!



22

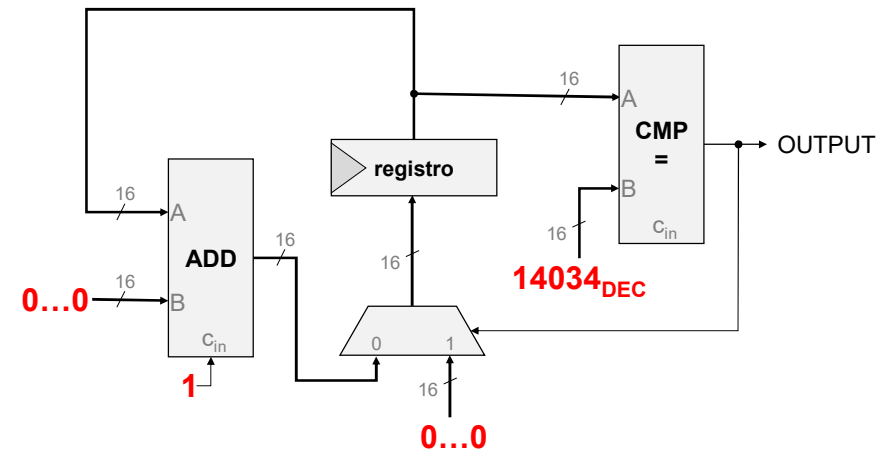
## Esempio: cosa fa questo circuito?






23

## Esempio: cosa fa questo circuito?






24

## Esercizi di riepilogo

- Prova a implementare il segnale di Set e di Reset in un registro parallelo

Comprendi nel dettaglio il funzionamento dei due circuiti proposti qui sopra

- Nel primo di essi:
  - Cosa leggo, nel tempo, dall'output, mentre l'input viene tenuto a 0?
  - Cosa leggo , nel tempo, dall'output, mentre l'input viene tenuto a 1?
  - Disegna il diagramma temporale le funzionamento, se input vale 0, ed ad un certo punto, nel corso del 2 ciclo di clock, l'input passa a 1
- Nel secondo di essi:
  - cosa succede se l'output viene mandato ad un campanello, che emette un suono quando riceve un impulso ad 1 (anche breve)?
  - Quanto deve essere lungo il ciclo di clock? Vedi lucido successivo



25

## Esercizi di riepilogo: calcolo del periodo di clock

- Ipotizziamo che (nel secondo schema)
  - Il tempo di commutazione di ADD sia 13ns (significa che, entro 13ns da quando si presentano in *tutti* gli input i valori corretti, AND produca l'output corretto per quei valori)
  - Il tempo di commutazione di CMP sia 8ns (idem)
  - Il tempo di commutazione del MUX sia 3ns (idem)
  - Il tempo di commutazione del registro sia 1ns (cioè, da quando scatta il clock – fronte di discesa, passa 1ns prima che l'input del registro sia presentato in uscita)
  - Il segnale si propaghi nei cavi istantaneamente (questa è sempre solo una approssimazione)
- Quanto deve essere lungo il periodo di clock, affinché, allo scattare del fronte di discesa, al registro sia presentato l'input corretto da memorizzare?
  - Cosa succede se il clock ha un periodo minore di questo? (cioè, una frequenza maggiore)



26