

# Circuiti Sequenziali

Corrado Santoro

Dipartimento di Matematica e Informatica

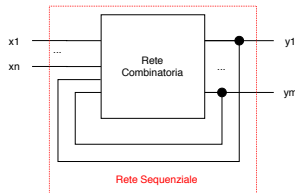
santoro@dmi.unict.it



Corso di Architettura degli Elaboratori

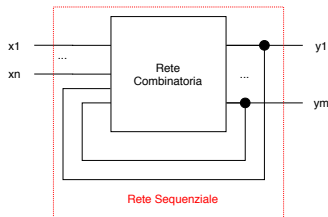
## Circuiti Sequenziali

- I **Circuiti Sequenziali** sono reti di porte logiche le cui uscite dipendono non solo dagli **ingressi** ma anche dal **tempo**
- In particolare, le uscite all'istante di tempo  $t^*$  dipendono:
  - Dal valore degli **ingressi all'istante  $t^*$**
  - Da valore delle **uscite stesse agli istanti precedenti**, cioè al tempo  $t < t^*$
- Le uscite **dipendono** dunque dagli **ingressi** e dalle **uscite stesse**
- Ciò viene ottenuto aggiungendo ad un rete combinatoria un meccanismo di **feedback**



## Circuiti Sequenziali

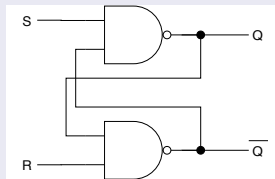
- Le **tabelle di verità** dei circuiti sequenziali includono dunque:
  - I valori degli **ingressi**
  - I valori delle **uscite all'istante precedente**
  - I **nuovi valori delle uscite all'istante attuale**



$x_1$	...	$x_n$	$y_1(t-1)$	...	$y_m(t-1)$	$y_1(t)$	...	$y_m(t)$
0	...	0	0	...	0	0	...	1
0	...	0	0	...	1	0	...	1
1	...	0	0	...	1	1	...	0
1	...	0	1	...	0	1	...	0

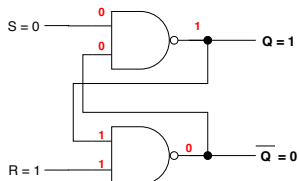
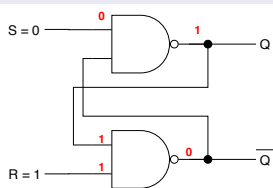
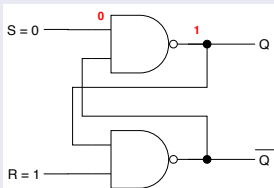
## Il Flip-Flop Set-Reset

- Il **Flip-Flop Set-Reset** è un tipico circuito sequenziale, ed è la base da cui vengono derivati tutti i circuiti sequenziali usati in un calcolatore
- Il nome “**flip-flop**” indica che si tratta di un circuito in grado di commutare tra due stati: **flip** ⇔ **flop**
- L'aggiunta “**set-reset**” indica i nomi dei due ingressi che permettono appunto di effettuare l'operazione di **set** e **reset** (dettagli più avanti)



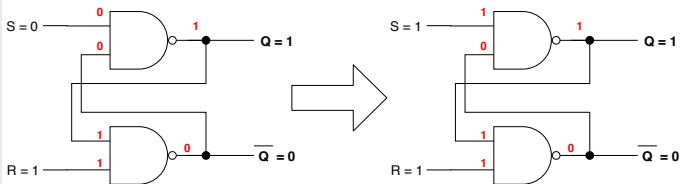
# Analisi del Flip-Flop Set-Reset

Consideriamo gli ingressi  $S = 0$ ,  $R = 1$



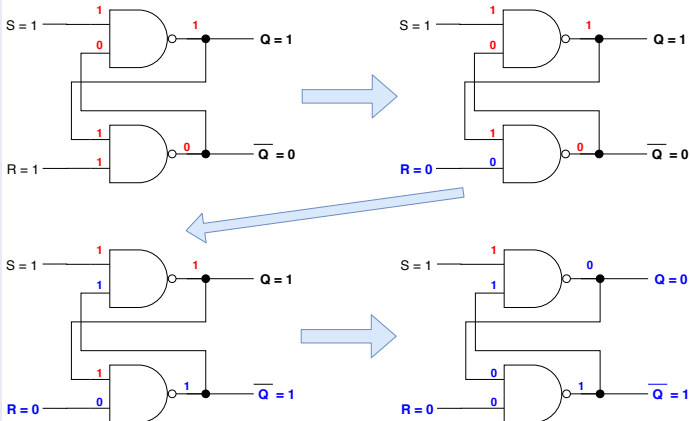
# Analisi del Flip-Flop Set-Reset

Passaggio degli ingressi da  $\{S = 0, R = 1\}$  a  $\{S = 1, R = 1\}$



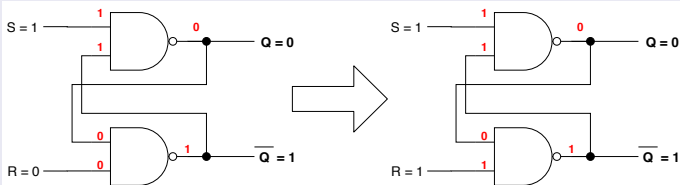
# Analisi del Flip-Flop Set-Reset

Passaggio degli ingressi da  $\{S = 1, R = 1\}$  a  $\{S = 1, R = 0\}$



# Analisi del Flip-Flop Set-Reset

Passaggio degli ingressi da  $\{S = 1, R = 0\}$  a  $\{S = 1, R = 1\}$



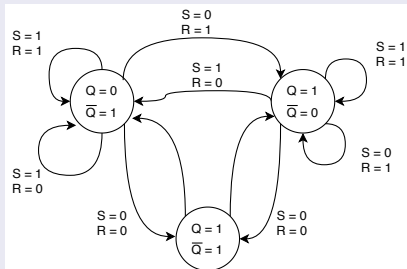


# Flip-Flop Set-Reset

## Tabella Verità FF-SR

$S$	$R$	$Q(t)$	$\overline{Q}(t)$	
0	1	1	0	Stato di "Set"
1	0	0	1	Stato di "Reset"
1	1	$Q(t-1)$	$\overline{Q}(t-1)$	Stato mantenuto
0	0	1	1	instabile!

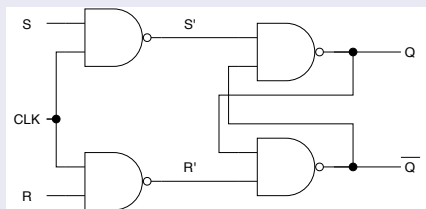
## Automa a Stati Finiti FF-SR



# Il Flip-Flop Set-Reset "Gated"

## Il Flip-Flop Set-Reset "Gated"

- Modifichiamo il FF-SR introducendo altre due porte NAND che vincolano gli ingressi ad un ulteriore ingresso che chiamiamo **Clock**
- Quando il Clock vale **0**, avremo  $S' = 1$  e  $R' = 1$ , pertanto qualunque variazione sugli ingressi  $S$  e  $R$  **non avrà alcun effetto sullo stato del flip-flop**
- Quando il Clock vale **1**, gli ingressi  $S$  e  $R$  saranno propagati negati su  $S'$  e  $R'$ , cioè  $S' = \bar{S}$  e  $R' = \bar{R}$ , il flip-flop **potrà cambiare stato**

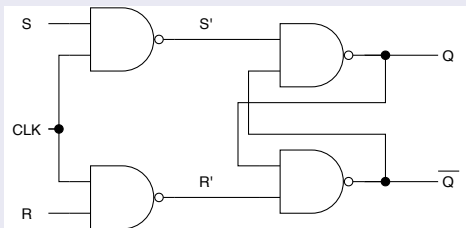


# Gated Flip-Flop Set-Reset

## Tabella Verità Gated FF-SR

$S$	$R$	$Clk$	$S'$	$R'$	$Q(t)$	$\bar{Q}(t)$	
$X$	$X$	0	1	1	$Q(t-1)$	$\bar{Q}(t-1)$	Stato mantenuto
1	0	1	0	1	1	0	Stato di "Set"
0	1	1	1	0	0	1	Stato di "Reset"
0	0	1	1	1	$Q(t-1)$	$\bar{Q}(t-1)$	Stato mantenuto
1	1	1	0	0	1	1	instabile!

## Gated FF-SR



# D-Type Flip-Flop

Modifichiamo il circuito come segue:

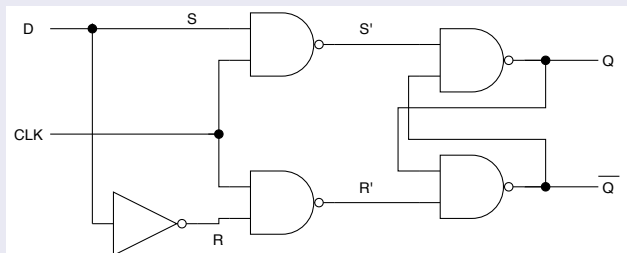
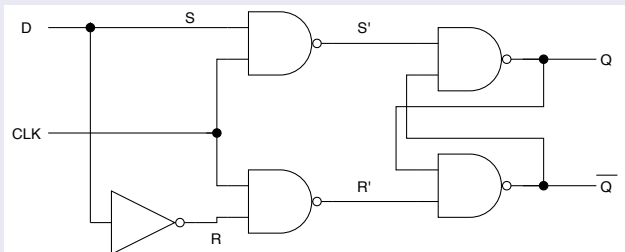


Tabella Verità D-Type FF

$D$	$Clk$	$S$	$R$	$S'$	$R'$	$Q(t)$	$\overline{Q}(t)$	
$X$	0	$D$	$\overline{D}$	1	1	$Q(t-1)$	$\overline{Q}(t-1)$	Stato mantenuto
0	1	0	1	1	0	0	1	Stato di "Reset"
1	1	1	0	0	1	1	0	Stato di "Set"

# D-Type Flip-Flop



## Tabella Verità D-Type FF

Consideriamo solo gli ingressi  $D$ ,  $CLK$  e l'uscita  $Q$ :

$D$	$Clk$	$Q(t)$	
$X$	0	$Q(t-1)$	L'uscita mantiene il valore
0	1	0	L'uscita "copia" l'ingresso $D$
1	1	1	L'uscita "copia" l'ingresso $D$

## D-Type FF e Memoria a 1 Bit

Consideriamo solo gli ingressi  $D$ ,  $CLK$  e l'uscita  $Q$ :

$D$	$Clk$	$Q(t)$	
$X$	0	$Q(t-1)$	L'uscita mantiene il valore
0	1	0	L'uscita "copia" l'ingresso $D$
1	1	1	L'uscita "copia" l'ingresso $D$

- Lo stato (uscita) del FF cambia **solo quando Clock = 1**
- Quando **Clock = 1**, l'uscita **ricopia** il valore dell'ingresso  $D$
- Quando **Clock = 0**, l'uscita **manitiene** il vecchio valore
- Il circuito opera da **memoria a un bit**

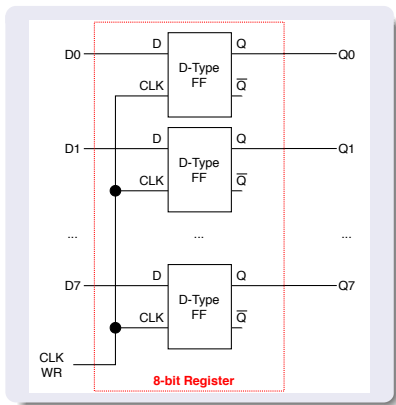
# Registro a 8 bit

## 8 Bit Register

Collegando insieme 8 flip-flop D-type si ottiene un **elemento di memoria a 8 bit** denominato **registro (a 8 bit)**

I Clock sono collegati tutti insieme in modo da avere un unico segnale per la "scrittura" (write) del dato in ingresso

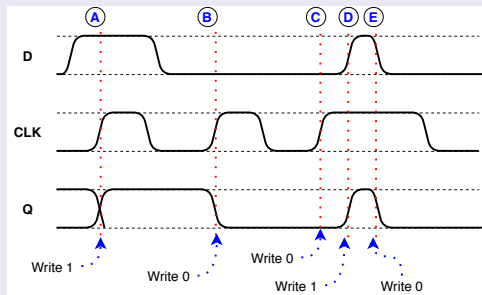
$D_n$	$WR$	$Q_n(t)$
X	0	$Q_n(t-1)$
X	1	$D_n$



# Flip-Flop e Edge Triggering

## D-Type e Clock

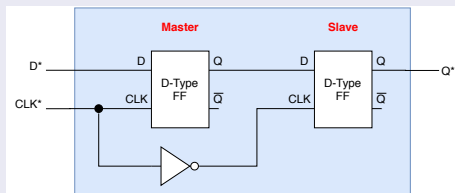
- Il FF-D-Type “memorizza” il bit su D quando **Clock = 1**
- Tuttavia, se, quando **Clock = 1**, l'ingresso D dovesse cambiare, allora l'uscita cambierà di conseguenza, come mostrato in figura negli istanti **C, D** ed **E**
- Questo comportamento costringe a mettere vincoli sul timing dei segnali D e CLK





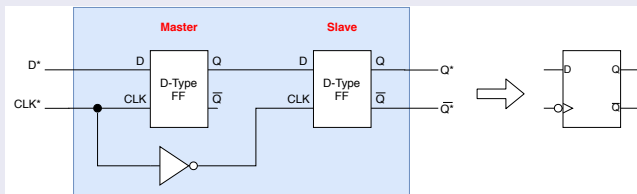
## D-Type Master-Slave

- Se tuttavia colleghiamo **in serie** due FF-D-Type come in figura otteniamo che:
  - Quando  $CLK^* = 1$  il segnale su  $D^*$  si propaga sull'uscita del primo FF (**Master**)
  - Quando  $CLK^* = 0$  l'uscita del primo FF (**Master**) si propaga sull'uscita del secondo FF (**Slave**) e quindi su  $Q^*$
- Il circuito complessivo **memorizza il dato** ( $D^* \rightarrow Q^*$ ) quando il segnale  $CLK^*$  passa **dallo stato 1 allo stato 0**, cioè su un **fronte di discesa** del segnale



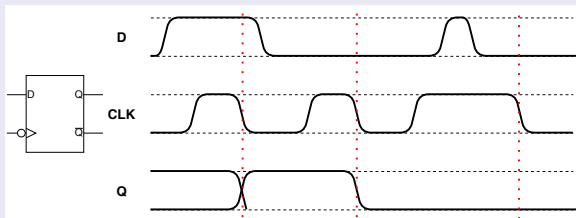
## Edge Triggered D-Type

- Il simbolo sulla destra è quello che si adotta per i FF di tipo **“edge-triggered”**, cioè il cui Clock agisce su un fronte del segnale piuttosto che sul livello
- Il **“triangolo”** indica il **fronte di salita**  $CLK\ 0 \rightarrow 1$
- Il **“triangolo con pallino”** indica il **fronte di discesa**  $CLK\ 1 \rightarrow 0$



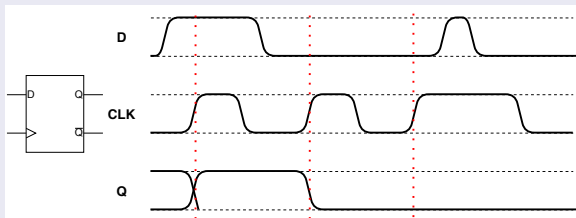
## Falling Edge Triggered D-Type

- Il FF “memorizza” il bit su D quando il **Clock** passa dallo stato 1 allo stato 0 (**falling edge**)

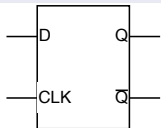


## Rising Edge Triggered D-Type

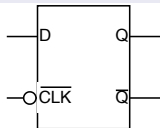
- Il FF “memorizza” il bit su D quando il **Clock** passa dallo stato 0 allo stato 1 (**rising edge**)



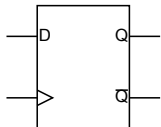
## Simboli dei vari FF D-Type



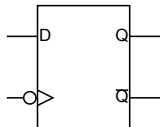
Level-Trigger, on CLK = 1



Level-Trigger, on CLK = 0



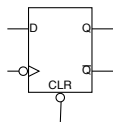
Edge-Trigger, on rising edge



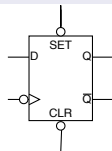
Edge-Trigger, on falling edge

# D-Type Flip-Flop with Clear and Preset

## Varianti dei FF D-Type



Edge-Trigger, falling edge  
with (async) clear

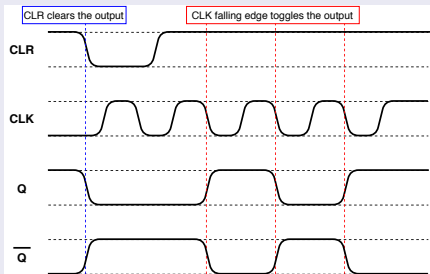
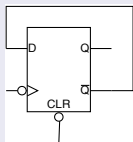


Edge-Trigger, falling edge  
with (async) clear and preset

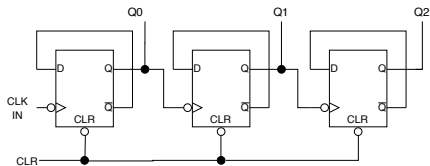
A volte i FF possono presentare degli ingressi di *CLEAR* o di *PRESET* che permettono di forzare le uscite indipendentemente dagli ingressi di *D* e *CLOCK*

<i>D</i>	<i>CLK</i>	<i>CLR</i>	<i>SET</i>	<i>Q(t)</i>	$\bar{Q}(t)$	
X	X	0	1	0	1	output clear
X	X	1	0	1	0	output set
0	↓	1	1	0	1	D copied to output
1	↓	1	1	1	0	D copied to output
X	X	1	1	$Q(t-1)$	$\bar{Q}(t-1)$	unchanged

## FF T-Type

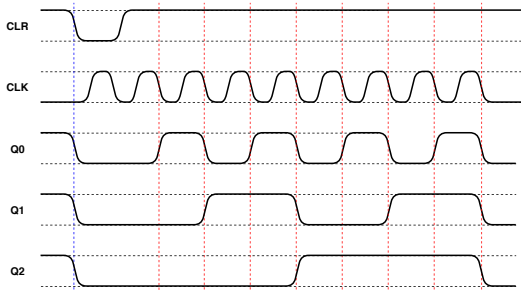


- Se, in un D-type, colleghiamo l'ingresso  $D$  all'uscita  $\bar{Q}$  otteniamo un flip-flop denominato **T-type (toggle)**
- Ad ogni "colpo di clock", l'uscita  $\bar{Q}$  viene "copiata" su  $Q$
- Otteniamo cioè che (ad ogni colpo di clock)  $Q(t) = \bar{Q}(t - 1)$
- In altri termini, ad ogni **colpo di clock** le uscite **cambiano stato (toggle)**



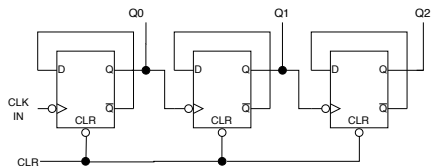
## Serie di FF T-Type

- Realizziamo un circuito in cui mettiamo 4 FF T-type "in serie", ovvero l'uscita del FF  $n$  diventa il clock del FF  $n + 1$
- Analizziamo la sequenza temporale



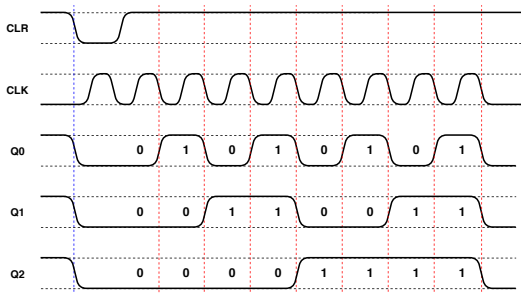


# Circuiti costruiti sui Flip-Flop

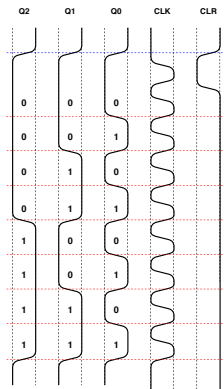
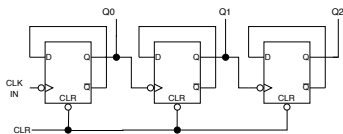


## Serie di FF T-Type

- Etichettiamo con 0 e 1 i vari stati logici delle uscite Q0, Q1 e Q2



# Il Contatore a 3 bit

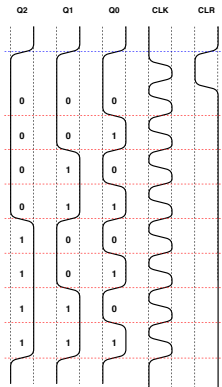
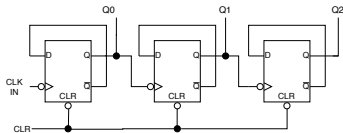


## Contatore a 3 bit

- Ruotiamo la figura di 90 gradi ed analizziamo le **sequenze** prodotte nel tempo da  $Q_0$ ,  $Q_1$  e  $Q_2$
- Esse costituiscono le **rappresentazioni binarie** della sequenza di numeri  $0, 1, 2, \dots, 7$

$Q_2$	$Q_1$	$Q_0$	Base 10
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

# Il Contatore a 3 bit

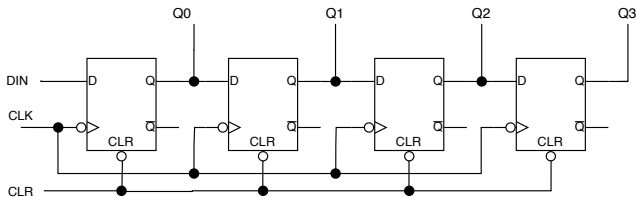


## Contatore a 3 bit

- Abbiamo dunque realizzato un circuito **contatore binario a 3 bit**

Q2	Q1	Q0	Base 10
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7
0	0	0	0
...	...	...	...

# Circuiti costruiti sui Flip-Flop

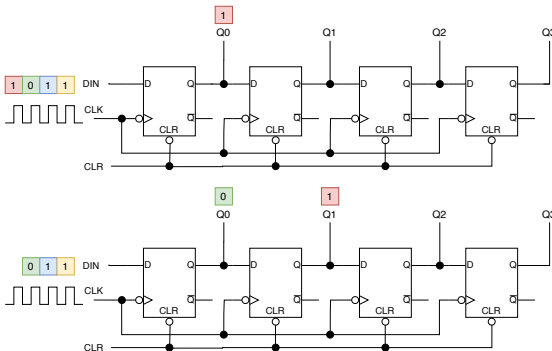


## Serie di FF D-Type / Shift-Register

- Realizziamo un circuito in cui mettiamo 4 FF D-type “in serie”, ovvero l’uscita del FF  $n$  diventa il  $D$  del FF  $n + 1$
- Colleghiamo inoltre i  $CLK$  e i  $CLR$  tutti insieme
- Ad ogni colpo di clock, il dato memorizzato nel FF  $n$  “passerà” al FF  $n + 1$ , e il dato presente su  $DIN$  sarà memorizzato sul primo FF
- Ovvero:  $Q_2 \rightarrow Q_3$ ,  $Q_1 \rightarrow Q_2$ ,  $Q_0 \rightarrow Q_1$ ,  $DIN \rightarrow Q_0$
- Abbiamo realizzato un **registro a scorrimento** o **shift register** a 4 Bit

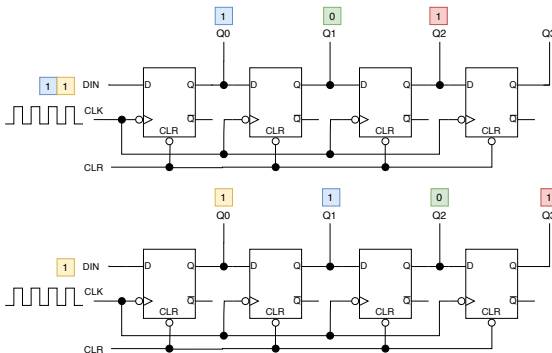
## Shift-Register, Serial-to-Parallel

- Uno degli impieghi degli shift registers è nelle periferiche di comunicazione
- Essi sono in grado di trasformare un “treno di bit” in un pattern completo (serial-to-parallel)



## Shift-Register, Serial-to-Parallel

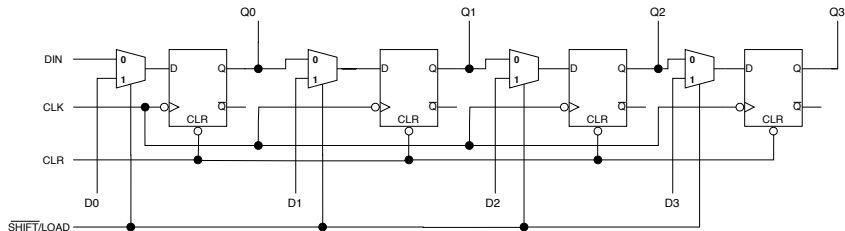
- Uno degli impieghi degli shift registers è nelle periferiche di comunicazione
- Essi sono in grado di trasformare un “treno di bit” in un pattern completo di  $n$  bit (**serial-to-parallel**)



# Shift Register

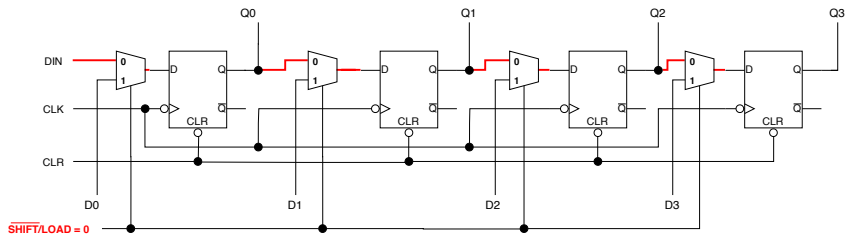
## Shift-Register, Parallel-to-Serial

- Modifichiamo uno shift-register aggiungendo un **multiplexer** nel path tra un FF e il successivo
- Il multiplexer permette di scegliere se inviare all'ingresso  $D$  del FF  $n$  l'uscita del FF  $n - 1$  oppure un bit da un ingresso esterno
- Tutti i mux sono controllati dallo stesso segnale che chiamiamo ***SHIFT/LOAD***



## Shift-Register, Parallel-to-Serial

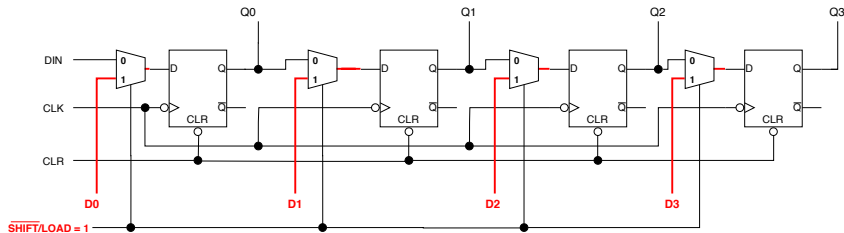
- Quando  $\overline{SHIFT}/LOAD = 0$ , i MUX connettono i FF in serie, pertanto il circuito agisce da **shift-register**





## Shift-Register, Parallel-to-Serial

- Quando  $\overline{SHIFT}/LOAD = 1$ , i MUX connettono gli ingressi  $D$  dei FF agli ingressi  $D_n$  del circuito
- Il circuito (al colpo di clock) **memorizza** il pattern binario su  $D_0, D_1, D_2, D_3$  il quale potrà poi essere **shiftato successivamente**



## Uso degli Shift-Register

- Periferiche di comunicazione
- Prodotto di un numero binario per le potenze del 2:

0 0 1 0 1 0 0 1  $\times 4 =$

0 0 1 0 1 0 0 1 *shift\_left 2 times =*

1 0 1 0 0 1 0 0

- Divisione (intera) di un numero binario per le potenze del 2:

0 0 1 0 1 0 0 1  $/ 4 =$

0 0 1 0 1 0 0 1 *shift\_right 2 times =*

0 0 0 0 1 0 1 0

# Circuiti Sequenziali

Corrado Santoro

Dipartimento di Matematica e Informatica

santoro@dmi.unict.it



Corso di Architettura degli Elaboratori