Registri e Contatori

Registri
Contatori

Introduzione

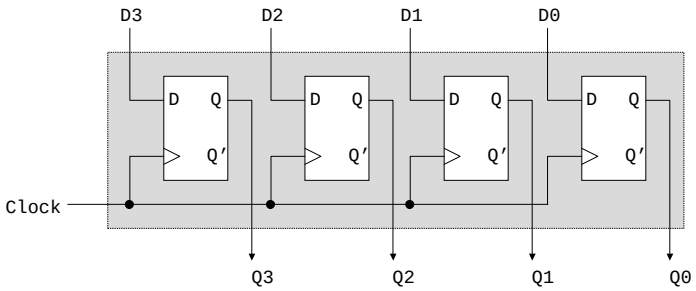
- Circuiti sequenziali speciali
  - Esiste una classe di circuiti sequenziali la cui progettazione potrebbe seguire il processo “classico” di sintesi ma che è più conveniente analizzare in altro modo.
    - La regolarità della struttura facilità la progettazione.
  - A questa classe appartengono:
    - Registri
      - Memorizzano una definita quantità di informazione
      - Possono operare sul contenuto una o più semplici trasformazioni.
        - » Shift destro/sinistro
        - » Caricamento parallelo/seriale
    - Contatori
      - Attraversano ripetutamente un numero definito di stati
        - » Contatori sincroni
        - » Contatori asincorni
    - Gestori di Code
      - FIFO, LIFO

Registri

- Un *registro* è un elemento di memoria in grado di conservare un insieme di bit, denominato *parola,* su cui può eventualmente operare una o più semplici trasformazioni.
  - Benché si possa utilizzare un qualunque tipo di bistabile, per realizzare i registri si preferisce utilizzare *FF D* (*master-slave* o *edge-triggered*).
- I registri si distinguono sulla base dei seguenti aspetti:
  - Modalità di caricamento dati
    - Parallelo
    - Seriale
  - Modalità di lettura dati
    - Parallelo
    - Seriale
  - Operazioni di scorrimento sui dati:
    - a destra e/o a sinistra (aritmetico o non aritmetico) e circolare.

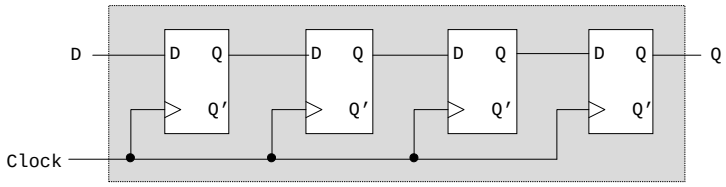
Registri

- Registro *parallelo-parallelo*
  - Esempio di registro a 4 bit.



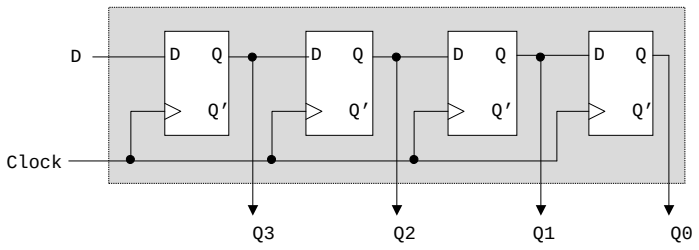
Registri

- Registro *serie-serie* (*Shift Register – Registro a Scorimento*)
  - Esempio di registro a 4 bit



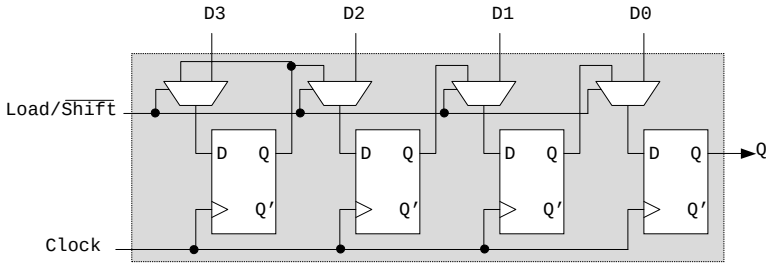
Registri

- Registro *serie-parallelo*
  - Esempio di registro a 4 bit



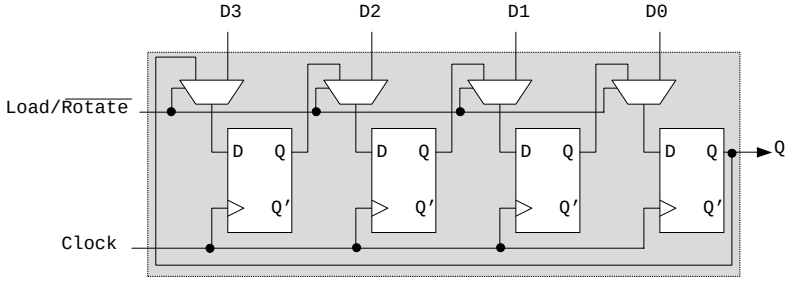
Registri

- Registro *parallelo-serie*
  - Esempio a 4 bit con shift-aritmetico (Shift Destro)
    - In fase di traslazione, ricopia il bit più significativo nella posizione più significativa (estensione del segno)



Registri

- Registro *circolare* a 4 bit
  - Esempio a 4 bit con rotazione a destra
    - In fase di traslazione, trasferisce il bit meno significativo al posto di quello più significativo, spostando i rimanenti di una posizione a destra.

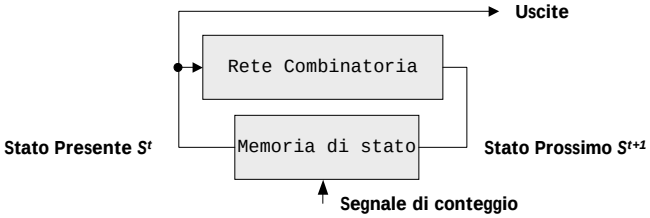


Contatori

- Un contatore è una rete *sequenziale* che, solitamente, riceve in ingresso solamente *un evento di conteggio* che sposta la posizione corrente in avanti - *upwards* – (o indietro - *downwards*) di una unità.
  - Il valore raggiunto è associato allo *stato presente*.
    - Possono esistere altri ingressi di *controllo* per la realizzazione di contatori bidirezionali; il metodo di progetto cambia di poco.
- Il contatore appartiene ad una famiglia di reti sequenziali "omogenee" caratterizzate da:
  - Specifiche di funzionamento analoghe per l'intera famiglia;
  - Ripetitività e località della struttura (in molto casi).
  - Metodologia di specifica semplificata rispetto alla generica tabella degli stati e metodologia di progetto semplificata rispetto a quella generale per le reti sequenziali;

Contatori

- Un contatore può anche essere visto come una *generica rete sequenziale dove,* a meno di particolari specifiche*, il valore d'uscita coincide con il valore di stato.*
  - A meno di casi particolari, non sono presenti reti di transcodifica
  - Si utilizzano Flip-flop *(*master slave o a commutazione sul fronte*)*



- In pratica, adottare le *tecniche generali di progetto* delle reti sequenziali risulta eccessivamente oneroso; si ricorre a tecniche *specifiche*, più semplici, secondo criteri *tipici della classe di circuiti*.
  - consentono buona *ottimizzazione*.

Contatori

- Un contatore si distingue per:
  - Il *modulo* M
    - il contatore conterà da 0 a M-1 e, al successivo impulso, *torna a 0*;
  - Il *codice*
    - Il contatore presenta all'esterno il valore del conteggio secondo un codice stabilito.
      - A numero minimo di bit: Il numero di tali bistabili è $\lceil \log_2 M \rceil$. (es: Gray, Binario Naturale)
      - Altri codici: se il codice è su *k* bit, converrà usare *k* bistabili anche qualora fosse *k*> $\lceil \log_2 M \rceil$, per evitare di inserire una rete di transcodifica fra i bistabili e l'uscita del contatore. (es: 1-hot, parità)
  - La *codifica*
    - Definisce la successione degli M valori associati allo stato attraverso cui il contatore evolve.
      - Nota: la codifica dello stato è definita a priori
      - Es: M=4 - codice Gray(codice a numero minimo di bit) - codifica: S0=00 S1=01 S2=11 S3=10
      - Es: M=4 - codice Parità Pari (codice a numero non minimo di bit): codifica: S0=000 S1=011 S2=101 S3=110

Contatori

- Oltre che per *modulo, codice e codifica*, i contatori si distinguono in *sincroni* e *asincroni*:
  - Contatore *sincrono*:
    - Tutti i bistabili ricevono simultaneamente in ingresso l'evento di conteggio;
      - Clock oppure Gated Clock (clock attraversa una rete combinatoria).
    - Le eventuali commutazioni sono tutte simultanee (*sincrone*), a parte modeste variazioni dovute alla propagazione attraverso le reti di eccitazione dei bistabili;
  - Contatore *asincrono*:
    - Almeno un bistabile *non* riceve in ingresso il segnale di conteggio
    - La sua eventuale commutazione è comandata da quella degli altri bistabili e avverrà con un ritardo dovuto alla propagazione attraverso tali bistabili (oltre che alle reti combinatorie eventualmente presenti);
- Nel seguito si tratterà in dettaglio il progetto dei contatori, essenzialmente quelli sincroni.

Contatori sincroni: *Contatore Binario Naturale*

- Contatore binario (modulo 2^n)
  - Modulo: 2^n; Codice: A numero minimo bit; Codifica: Binaria Naturale
  - Bistabile utilizzato: T

Tabella delle transizioni e delle eccitazioni per M= 2^1

Q_0	Q_0^+	Q_0	T_0
0	1	0	1
1	0	1	1

Tabella delle transizioni e delle eccitazioni per M= 2^2

Q_1Q_0	$Q_1^+Q_0^+$	Q_1Q_0	T_1T_0
0 0	0 1	0 0	0 1
0 1	1 0	0 1	1 1
1 0	1 1	1 0	0 1
1 1	0 0	1 1	1 1

Tabella delle transizioni e delle eccitazioni per M= 2^3

$Q_2Q_1Q_0$	$Q_2^+Q_1^+Q_0^+$	$Q_2Q_1Q_0$	$T_2T_1T_0$
0 0 0	0 0 1	0 0 0	0 0 1
0 0 1	0 1 0	0 0 1	0 1 1
0 1 0	0 1 1	0 1 0	0 0 1
0 1 1	1 0 0	0 1 1	1 1 1
1 0 0	1 0 1	1 0 0	0 0 1
1 0 1	1 1 0	1 0 1	0 1 1
1 1 0	1 1 1	1 1 0	0 0 1
1 1 1	0 0 0	1 1 1	1 1 1

Contatori sincroni: *Contatore Binario Naturale*

- L'analisi delle tabelle delle eccitazioni evidenzia la seguente regolarità (M=2^4):

$Q_3Q_2Q_1Q_0$	$T_3T_2T_1T_0$
0 0 0 0	0 0 0 1
0 0 0 1	0 0 1 1
0 0 1 0	0 0 0 1
0 0 1 1	0 1 1 1
0 1 0 0	0 0 0 1
0 1 0 1	0 0 1 1
0 1 1 0	0 0 0 1
0 1 1 1	1 1 1 1
1 0 0 0	0 0 0 1
1 0 0 1	0 0 1 1
1 0 1 0	0 0 0 1
1 0 1 1	0 1 1 1
1 1 0 0	0 0 0 1
1 1 0 1	0 0 1 1
1 1 1 0	0 0 0 1
1 1 1 1	1 1 1 1

$T_0 \equiv 1$

$Q_3Q_2Q_1Q_0$	$T_3T_2T_1T_0$
0 0 0 **0**	0 0 0 1
0 0 0 **1**	0 0 **1** 1
0 0 1 0	0 0 0 1
0 0 1 1	0 1 1 1
0 1 0 0	0 0 0 1
0 1 0 1	0 0 1 1
0 1 1 0	0 0 0 1
0 1 1 1	1 1 1 1
1 0 0 0	0 0 0 1
1 0 0 1	0 0 1 1
1 0 1 0	0 0 0 1
1 0 1 1	0 1 1 1
1 1 0 0	0 0 0 1
1 1 0 1	0 0 1 1
1 1 1 0	0 0 0 1
1 1 1 1	1 1 1 1

$T_1 = Q_0$

$Q_3Q_2Q_1Q_0$	$T_3T_2T_1T_0$
0 0 0 0	0 0 0 1
0 0 0 1	0 0 1 1
0 0 1 0	0 0 0 1
0 0 1 1	0 1 1 1
0 1 0 0	0 0 0 1
0 1 0 1	0 0 1 1
0 1 1 0	0 0 0 1
0 1 1 1	1 1 1 1
1 0 0 0	0 0 0 1
1 0 0 1	0 0 1 1
1 0 1 0	0 0 0 1
1 0 1 1	0 1 1 1
1 1 0 0	0 0 0 1
1 1 0 1	0 0 1 1
1 1 1 0	0 0 0 1
1 1 1 1	1 1 1 1

$T_2=Q_1*Q_0=Q_1*T_1$

$Q_3Q_2Q_1Q_0$	$T_3T_2T_1T_0$
0 **0 0 0**	0 0 0 1
0 **0 0 1**	0 **0** 1 1
0 **0 1 0**	0 **0** 0 1
0 **0 1 1**	0 **1** 1 1
0 1 0 0	0 0 0 1
0 1 0 1	0 0 1 1
0 1 1 0	0 0 0 1
0 1 1 1	1 1 1 1
1 0 0 0	0 0 0 1
1 0 0 1	0 0 1 1
1 0 1 0	0 0 0 1
1 0 1 1	0 1 1 1
1 1 0 0	0 0 0 1
1 1 0 1	0 0 1 1
1 1 1 0	0 0 0 1
1 1 1 1	1 1 1 1

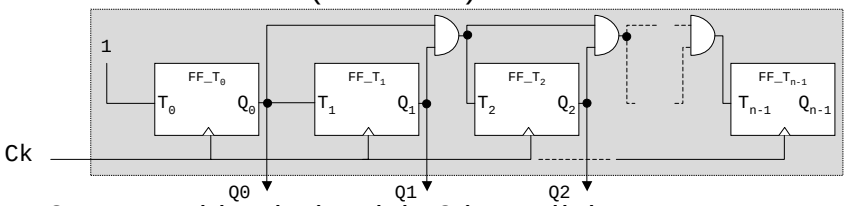
$T_3=Q_2*Q_1*Q_0=Q_2*T_2$

Contatori sincroni: *Contatore Binario Naturale*

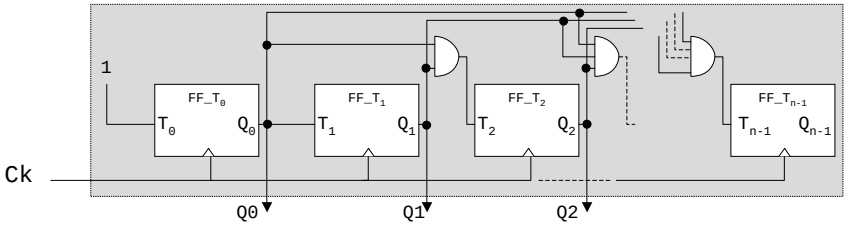
- Sono evidenti due possibili implementazioni per le funzioni di eccitazione:
  - Contatore serie: $T_0=1$; $T_1=Q_0$; $T_n=Q_{n-1}*T_{n-1}$
    - Tutti gli stadi, ad esclusione dei primi due, risultano perfettamente identici.
    - La regolarità della struttura è "pagata" con un maggior ritardo di propagazione (limita la frequenza di funzinamento).
    - Nota: la frequenza di funzionamento si riduce linearmente con la dimensione del contatore poiché T_i diventa stabile solo dopo che lo è diventato T_{i-1}.
  - Contatore parallelo: $T_0=1$; $T_1=Q_0$; $T_n=Q_{n-1}*Q_{n-2}*Q_{n-3}...*Q_0$
    - Schema molto semplice e regolare.
    - Minor ritardo di propagazione rispetto al caso precedente (frequenza di funzionamento maggiore rispetto al caso precedente).
    - Nota: la frequenza di funzionamento si riduce all'aumentare delle dimensioni del contatore a causa dell'aumento del numero degli ingressi alle porte AND.
- In generale, la regolarità deriva dal *ciclo di conteggio:* cambiando tipo di bistabile (es: FFD) le funzioni di eccitazione cambiano, ma la regolarità resta.

Contatori sincroni: *Contatore Binario Naturale*

- Contatore binario (modulo 2^n) serie:



- Contatore binario (modulo 2^n) parallelo:



Contatori sincroni: *Contatore Binario Naturale*

- Contatore binario (modulo 2^n)
  - Modulo: 2^n; Codice: A numero minimo bit; Codifica: Binaria Naturale
  - Bistabile utilizzato: D

Tabella delle eccitazioni per M= 2^1

Q_0	D_0
0	1
1	0

Tabella delle eccitazioni per M= 2^2

Q_1Q_0	D_1D_0
0 0	0 1
0 1	1 0
1 0	1 1
1 1	0 0

Tabella delle eccitazioni per M= 2^3

$Q_2Q_1Q_0$	$D_2D_1D_0$
0 0 0	0 0 1
0 0 1	0 1 0
0 1 0	0 1 1
0 1 1	1 0 0
1 0 0	1 0 1
1 0 1	1 1 0
1 1 0	1 1 1
1 1 1	0 0 0

Contatori sincroni: *Contatore Binario Naturale*

- L'analisi delle tabelle delle eccitazioni evidenzia la seguente regolarità (M=2^4):

$Q_3Q_2Q_1Q_0$	$D_3D_2D_1D_0$
0 0 0 **0**	0 0 0 **1**
0 0 0 **1**	0 0 1 **0**
0 0 1 0	0 0 1 1
0 0 1 1	0 1 0 0
0 1 0 0	0 1 0 1
0 1 0 1	0 1 1 0
0 1 1 0	0 1 1 1
0 1 1 1	1 0 0 0
1 0 0 0	1 0 0 1
1 0 0 1	1 0 1 0
1 0 1 0	1 0 1 1
1 0 1 1	1 1 0 0
1 1 0 0	1 1 0 1
1 1 0 1	1 1 1 0
1 1 1 0	1 1 1 1
1 1 1 1	0 0 0 0

$Q_3Q_2Q_1Q_0$	$D_3D_2D_1D_0$
0 0 **0 0**	0 0 **0** 1
0 0 **0 1**	0 0 **1** 0
0 0 **1 0**	0 0 **1** 1
0 0 **1 1**	0 1 **0** 0
0 1 0 0	0 1 0 1
0 1 0 1	0 1 1 0
0 1 1 0	0 1 1 1
0 1 1 1	1 0 0 0
1 0 0 0	1 0 0 1
1 0 0 1	1 0 1 0
1 0 1 0	1 0 1 1
1 0 1 1	1 1 0 0
1 1 0 0	1 1 0 1
1 1 0 1	1 1 1 0
1 1 1 0	1 1 1 1
1 1 1 1	0 0 0 0

$Q_3Q_2Q_1Q_0$	$D_3D_2D_1D_0$
0 **0 0 0**	0 **0** 0 1
0 **0 0 1**	0 **0** 1 0
0 **0 1 0**	0 **0** 1 1
0 **0 1 1**	0 **1** 0 0
0 **1 0 0**	0 **1** 0 1
0 **1 0 1**	0 **1** 1 0
0 **1 1 0**	0 **1** 1 1
0 **1 1 1**	1 **0** 0 0
1 0 0 0	1 0 0 1
1 0 0 1	1 0 1 0
1 0 1 0	1 0 1 1
1 0 1 1	1 1 0 0
1 1 0 0	1 1 0 1
1 1 0 1	1 1 1 0
1 1 1 0	1 1 1 1
1 1 1 1	0 0 0 0

$Q_3Q_2Q_1Q_0$	$D_3D_2D_1D_0$
0 0 0 0	**0** 0 0 1
0 0 0 1	**0** 0 1 0
0 0 1 0	**0** 0 1 1
0 0 1 1	**0** 1 0 0
0 1 0 0	**0** 1 0 1
0 1 0 1	**0** 1 1 0
0 1 1 0	**0** 1 1 1
0 1 1 1	**1** 0 0 0
1 0 0 0	**1** 0 0 1
1 0 0 1	**1** 0 1 0
1 0 1 0	**1** 0 1 1
1 0 1 1	**1** 1 0 0
1 1 0 0	**1** 1 0 1
1 1 0 1	**1** 1 1 0
1 1 1 0	**1** 1 1 1
1 1 1 1	**0** 0 0 0

Parallelo: $D_0=Q_0\oplus 1=Q_0'$ $D_1=Q_1\oplus Q_0$ $D_2=Q_2\oplus(Q_1*Q_0)$ $D_3=Q_3\oplus(Q_2*Q_1*Q_0)$

Serie: $D_0=Q_0\oplus 1=Q_0'$ $D_1=Q_1\oplus Q_0$ $D_2=Q_2\oplus(Q_1*Q_0)=Q_2\oplus(Q_1*K_0)$ $D_3=Q_3\oplus(Q_2*K_1)$

Contatori sincroni: *codici e moduli liberi*

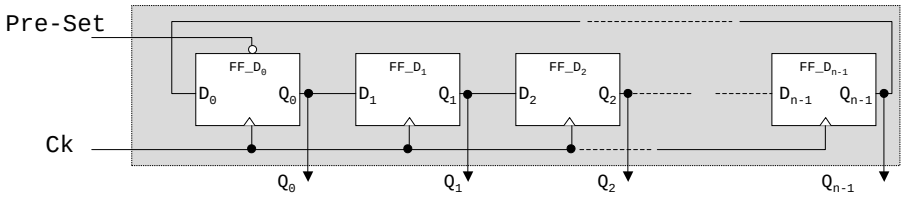
- Due casi diversi:
  - Progetto di contatori con modulo libero (2^n o diverso da 2^n), codice a numero **non** minimo bit e codifica **non** binaria naturale;
    - Struttura regolare.
      - Contatori ad anello (*codice one-hot*)
      - Contatore ad anello incrociato;
    - A struttura non regolare.
      - Si applica una metodologia di progetto semplificata rispetto a quella generale per le reti sequenziali;
  - Progetto di contatori con modulo diverso da 2^n, codice a numero minimo bit e codifica binaria naturale;
    - A struttura non regolare.
      - Si applica una metodologia di progetto semplificata rispetto a quella generale per le reti sequenziali;

Contatori sincroni: *codici e moduli liberi*

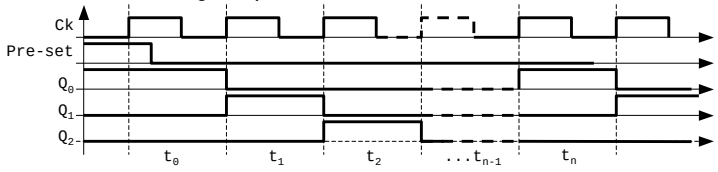
- Contatore "ad anello"
  - Modulo: n; Codice: *One hot*; Codifica: 2^k
  - Bistabile utilizzato: D
  - Codice *one-hot*:
    - In ogni codifica valida uno e un solo bit assume valore 1, tutti gli altri valgono 0;
    - Per codificare *n* informazioni diverse occorrono *n* bit
      - il codice non è a numero minimo di bit.
    - Esempio: i numeri da 0 a 3 sono codificati come:
      - 0 = 0001 (2^0)
      - 1 = 0010 (2^1)
      - 2 = 0100 (2^2)
      - 3 = 1000 (2^3)
      - esiste una corrispondenza 1-a-1 fra l'entità codificata e la posizione dell'unico 1 nella codifica.

Contatori sincroni: *codici e moduli liberi*

- Contatore "ad anello" (*ring counter*) modulo n:
  - È un registro a scorrimento con riporto tra stadio iniziale e finale



  - Il valore del FFD0 viene posto a 1 prima dell'inizio del conteggio; i rimanenti FFD vengono posti a 0.



Contatori sincroni: *codici e moduli liberi*

- Il contatore "ad anello" ha una struttura ad alto costo ma molto semplice, compatta e veloce
  - il numero di bistabili è molto più elevato del minimo e cresce linearmente.
- Viene utilizzato in applicazioni nelle quali si deve abilitare uno e un solo sottosistema; il contatore svolge il ruolo di unità di controllo.
  - Lo stato di ogni bistabile del contatore costituisce immediatamente il segnale di controllo e non occorre alcuna rete di transcodifica.
  - Se gli stati del contatore sono *n*, le linee di segnale che si inviano ai sottosistemi controllati sono ancora *n*.
    - Nota: l'uso di un contatore con un codice a numero minimo bit – es., binario naturale – richiede una rete di *transcodifica* che per ogni stato del contatore generi un valore attivo su una sola delle *n* linee di segnale in uscita (rete combinatoria con $k = \lceil \log n \rceil$ ingressi e *n* uscite); la rete di transcodifica, al crescere di *n*, ha costi crescenti e introduce crescenti ritardi di propagazione

Contatori sincroni: *codici e moduli liberi*

- Contatore "ad anello incrociato"
  - Modulo: 2*n (nota: sempre pari);
  - Codice e Codifica (esempio):

	Q_1Q_0	$Q_1'Q_0'$
0	0 0	0 1
1	0 1	1 1
2	1 1	1 0
3	1 0	0 0

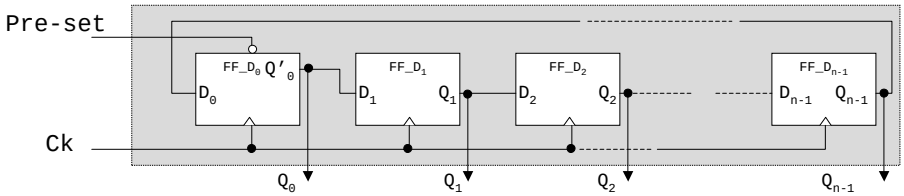
	$Q_2Q_1Q_0$	$Q_2'Q_1'Q_0'$
0	0 0 0	0 0 1
1	0 0 1	0 1 1
2	0 1 1	1 1 1
3	1 1 1	1 1 0
4	1 1 0	1 0 0
5	1 0 0	0 0 0

	$Q_3Q_2Q_1Q_0$	$Q_3'Q_2'Q_1'Q_0'$
0	0 0 0 0	0 0 0 1
1	0 0 0 1	0 0 1 1
2	0 0 1 1	0 1 1 1
3	0 1 1 1	1 1 1 1
4	1 1 1 1	1 1 1 0
5	1 1 1 0	1 1 0 0
6	1 1 0 0	1 0 0 0
7	1 0 0 0	0 0 0 0

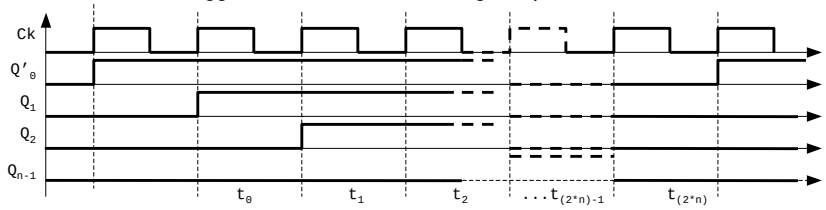
  - Bistabile utilizzato: D
  - Per codificare 2*n informazioni diverse occorrono n bit
    - Il codice non è a numero minimo di bit.
  - Svantaggi principali: modulo sempre pari, codice e codifica senza particolare campo di applicabilità
  - Vantaggio principale: distanza di Hamming unitaria, prestazioni elevate, meno elementi di memoria rispetto al contatore ad anello

Contatori sincroni: *codici e moduli liberi*

- Contatore "ad anello incrociato" modulo 2*n:



  - Il valore del FFD_0 viene inizializzato a 1 (quindi Q'_0 vale 0) prima dell'inizio del conteggio; i rimanenti FFD vengono posti a 0.



Contatori sincroni: *codici e moduli liberi*

- Contatore modulo diverso da 2^n: Esempio1.
  - Modulo: 6; Codice: A numero minimo bit; Codifica: Binaria Naturale
  - Bistabile utilizzato: T

Tabella delle transizioni e delle eccitazioni per M= 6

$Q_2Q_1Q_0$	$Q_2^*Q_1^*Q_0^*$
0 0 0	0 0 1
0 0 1	0 1 0
0 1 0	0 1 1
0 1 1	1 0 0
1 0 0	1 0 1
1 0 1	0 0 0

⇒

$Q_2Q_1Q_0$	$T_2T_1T_0$
0 0 0	0 0 1
0 0 1	0 1 1
0 1 0	0 0 1
0 1 1	1 1 1
1 0 0	0 0 1
1 0 1	1 0 1

⇒

$T_2=Q_1^*Q_0 + Q_2^*Q_0$
$T_1=Q_2'^*Q_0$
$T_0=1$

Nota: le equazioni derivano dalla sintesi delle tre funzioni combinatorie T_0, T_1 e T_2

Contatori sincroni: *codici e moduli liberi*

- Contatore modulo diverso da 2^n: Esempio2.
  - Modulo: 10; Codice: A numero minimo bit; Codifica: Binaria Naturale (Contatore *BCD* o *Decadico*)
  - Bistabile utilizzato: T

Tabella delle transizioni e delle eccitazioni per M= 10

$Q_3Q_2Q_1Q_0$	$Q_3^*Q_2^*Q_1^*Q_0^*$
0 0 0 0	0 0 0 1
0 0 0 1	0 0 1 0
0 0 1 0	0 0 1 1
0 0 1 1	0 1 0 0
0 1 0 0	0 1 0 1
0 1 0 1	0 1 1 0
0 1 1 0	0 1 1 1
0 1 1 1	1 0 0 0
1 0 0 0	1 0 0 1
1 0 0 1	0 0 0 0

⇒

$Q_3Q_2Q_1Q_0$	$T_3T_2T_1T_0$
0 0 0 0	0 0 0 1
0 0 0 1	0 0 1 1
0 0 1 0	0 0 0 1
0 0 1 1	0 1 1 1
0 1 0 0	0 0 0 1
0 1 0 1	0 0 1 1
0 1 1 0	0 0 0 1
0 1 1 1	1 1 1 1
1 0 0 0	0 0 0 1
1 0 0 1	1 0 0 1

⇒

$T_3=Q_3^*Q_0+Q_2^*Q_1^*Q_0$
$T_2=Q_1^*Q_0$
$T_1=Q_3'^*Q_0$
$T_0=1$

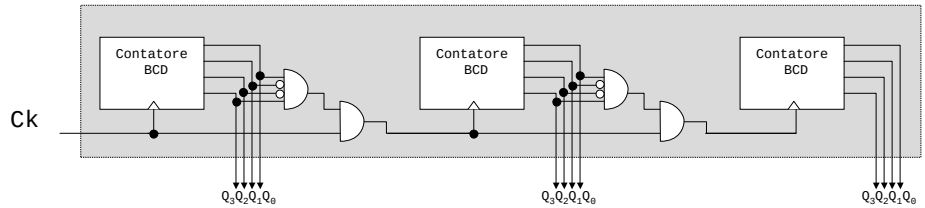
Nota: In modo analogo si potrebbe ottenere la realizzazione mediante FFD.
$D_0=Q_0'$
$D_1=Q_3'^*Q_1'^*Q_0+Q_3'^*Q_1^*Q_0'$
$D_2=Q_3'^*Q_2'^*Q_1^*Q_0+Q_2^*Q_0'+Q_2^*Q_1'$
$D_3=Q_3^*Q_1'+Q_2^*Q_1^*Q_0$

Contatori sincroni: *Composizione di contatori*

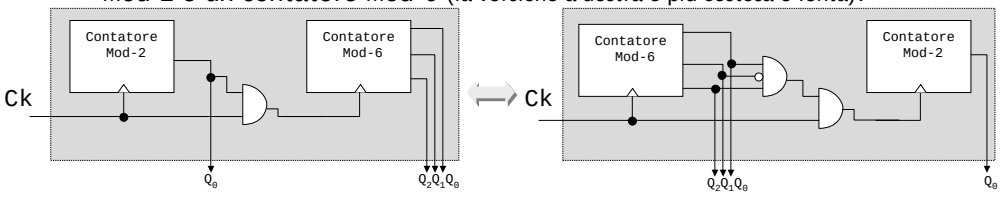
- É possibile realizzare contatori per moduli elevati partendo da contatori più semplici
  - Esempio: realizzare un contatore a k cifre decimali utilizzando K blocchi del contatore decadico (Mod-10 ([0..9]));
- Ogni sotto-contatore genera un segnale di traboccamento (*carry*) che, quando raggiunge valore 1, consente al clock di attivare il sotto-contatore collegato ad esso in cascata.
- La condizione di traboccamento è quella indicata dalla ultima configurazione di stato presente prodotta dal contatore a valle.
  - Esempio: nel contatore BCD, la condizione di traboccamento è 1001 che corrisponde a $f(Q_3,Q_2,Q_1,Q_0)=Q_3*Q_2'*Q_1'*Q_0$
- Il modulo del contatore complesso è il prodotto dei moduli.
  - Esempio: Due contatori Mod-2 e Mod-5 possono produrre un contatore decadico.

Contatori sincroni: *Composizione di contatori*

- Esempio: contatore BCD a 3 Cifre (Mod-1000).



- Esempio: contatore Mod-12 mediante composizione di un contatore Mod-2 e un contatore Mod-6 (la versione a destra è più costosa e lenta).

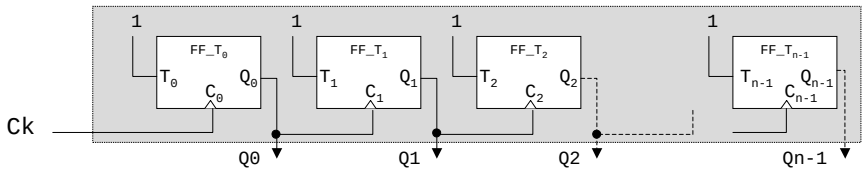


Contatori asincroni: *Generalità*

- Contatore *asincrono*:
  - Almeno un bistabile *non* riceve in ingresso il segnale di conteggio ne in modo diretto ne in modo indiretto;
    - Indiretto: clock in AND con una funzione logica che ha, come supporto, alcune/tutte le variabili di stato (*gated clock*).
  - La sua commutazione dei bistabili che non sono collegati al clock è comandata da quella degli altri bistabili e avverrà con un ritardo dovuto alla propagazione attraverso tali FF (oltre che alle eventuali reti combinatorie);
  - La commutazione è generata dall'opportuno fronte di commutazione sull'uscita di almeno uno degli altri bistabili coinvolti.
- *Modulo*, *codice* e *codifica* possono anche essere specificati arbitrariamente; purtrppo, può accadere che un contatore asincrono con queste caratteristiche non sia realizzabile.
- Ipotesi: uso di bistabili T che commutano sul fronte ed in cui T viene posto ad 1.

Contatori asincroni: *Contatore Binario Naturale*

- Contatore binario (modulo 2^n)
  - Modulo: 2^n; Codice: A numero minimo bit; Codifica: Binaria Naturale
  - Bistabile utilizzato: T
    - Il contatore è ottenuto collegando in cascata i bistabili.
      - Con FFT che commutano sul fronte di salita il contatore *conta indietro*.
        - » *conta avanti* con FF che commutano sul fronte di discesa.
      - Funzionamento: il fronte di salita del clock modifica lo stato di FF_T_0 che passa da 0 ad 1; questo fronte di salita modifica lo stato di FF_T_1 che, a sua volta, genera un fronte di salita che modifica modifica lo stato di FF_T_2 ; ….
        - » 000…00, 111…11, 111..10, …



Contatori asincroni: *Contatore Binario Naturale*

- Contatore binario (modulo 2^n)
  - Modulo: diverso da 2^n; Codice: A numero minimo bit; Codifica: Binaria Naturale
  - Bistabile utilizzato: T
    - Contatore indietro (conta in avanti se il fronte di commutazione è quello di discesa)
      - Il fronte di salita del clock modifica lo stato di FF_T0 che passa da 0 ad 1; questo fronte di salita modifica lo stato di FF_T1; ….
      - La configurazione successiva all'ultima valida pone lo stato di tutti i FF a 000…00.
    - Esempio: contatore in avanti MOD5