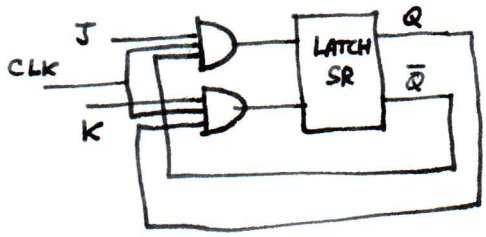
16 ottobre 2017

Riassunto della lezione odierna (vedi foto).

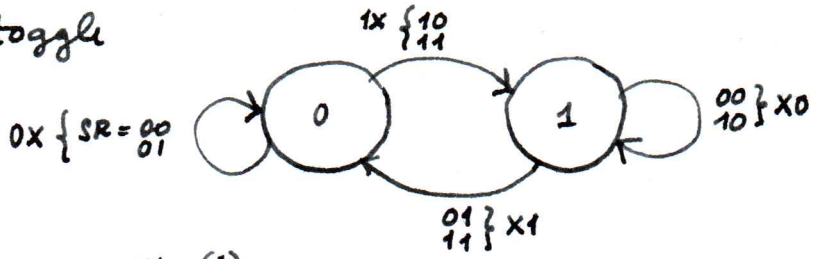
Prima parte: conclusione capitolo flip-flop, e introduzione del diagramma degli stati.

JK: $Q' = J\bar{Q} + \bar{K}Q$

T: $Q' = T \oplus Q$

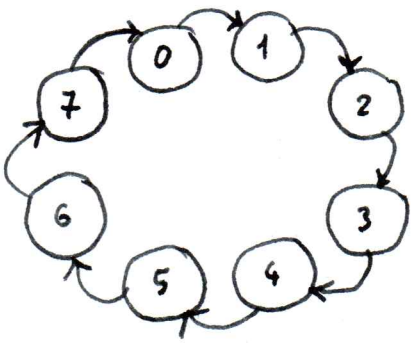


JK è il più completo:
hold, set, reset, toggle

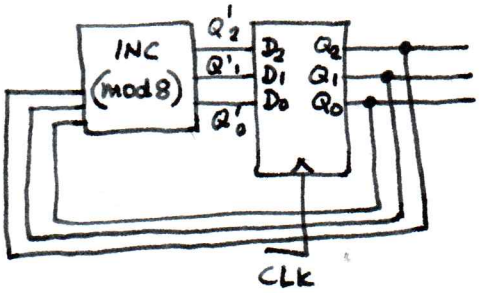


FF edge-triggered → positive (↑) / negative (↓)

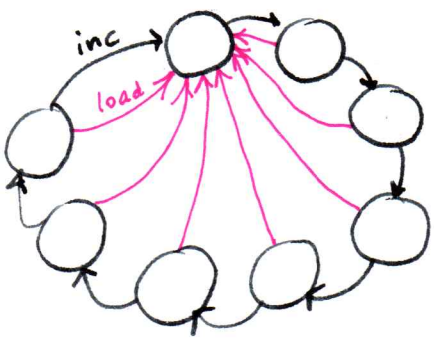
Contatore modulo 8:

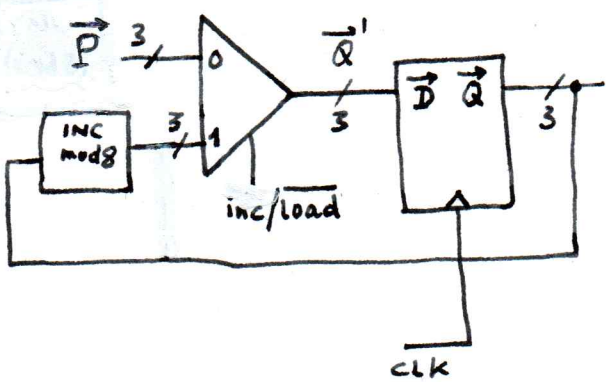


rete autonoma



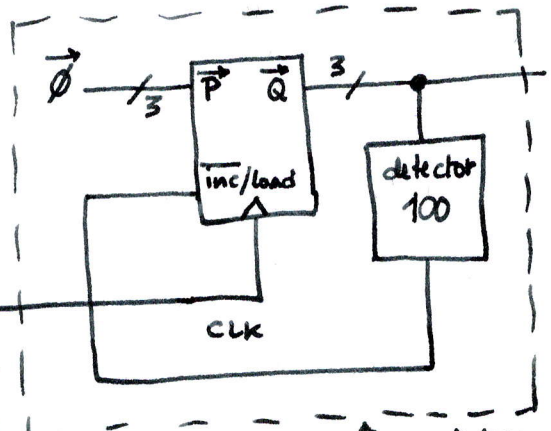
Aggiunta dell'ingresso di LOAD:

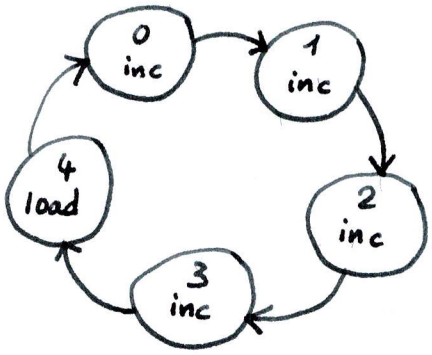




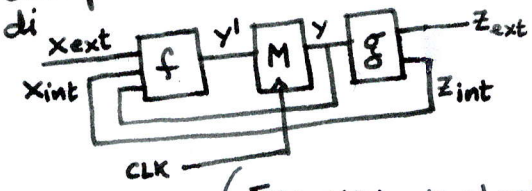
non è più una rete autonoma
(anche se $\vec{P}$ è costante, c'è l'ingresso $\overline{inc/load}$)

Costruzione di un contatore fra 5 a partire da uno per 8 :





NOTA : lo schema a fianco è un caso particolare di

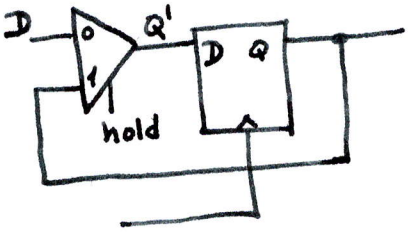


Σ contatore MOD 5

E' comunque una rete autonoma (se si mantiene costante $\vec{P}$)

(Esercizio proposto: aggiungere la funzionalità di HOLD al FFD.

Soluzione :



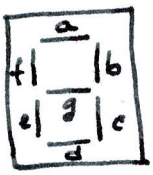
)

Uso della funzione g : periodo 5

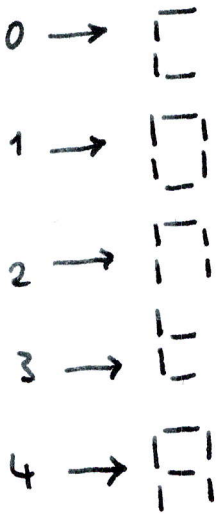
Se l'output invece di 0 1 2 3 4 0 1 2 3 4 ...

deve servire a generare la scritta CONTACONTA ... (stesso periodo)

su un display 7 segmenti

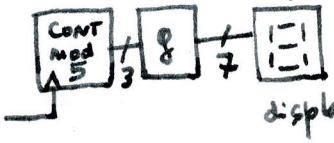


es :



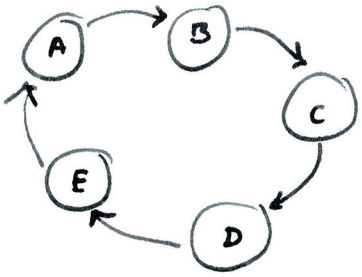
$Q_2 Q_1 Q_0$	a b c d e f g
000	1 0 0 1 1 1 0
001	
010	etc.
011	
100	
101	x x x x x x x
110	x x x x x x x
111	x x x x x x x

funzione $g(Q_2, Q_1, Q_0)$



Sintesi "monoblocco" di un contatore mod 5:

Si parte dal diagr. di stato

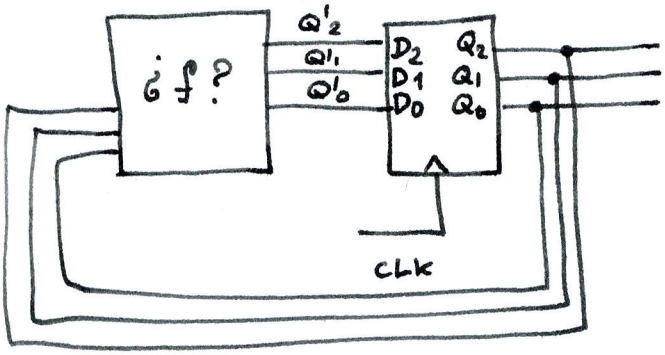


Si codificano gli stati in modo univoco e consecutivo:

A → 000
B → 001
C → 010
D → 011
E → 110

Servono 3 bit di stato:

disegno subito un registro di stato con 3 FF D:

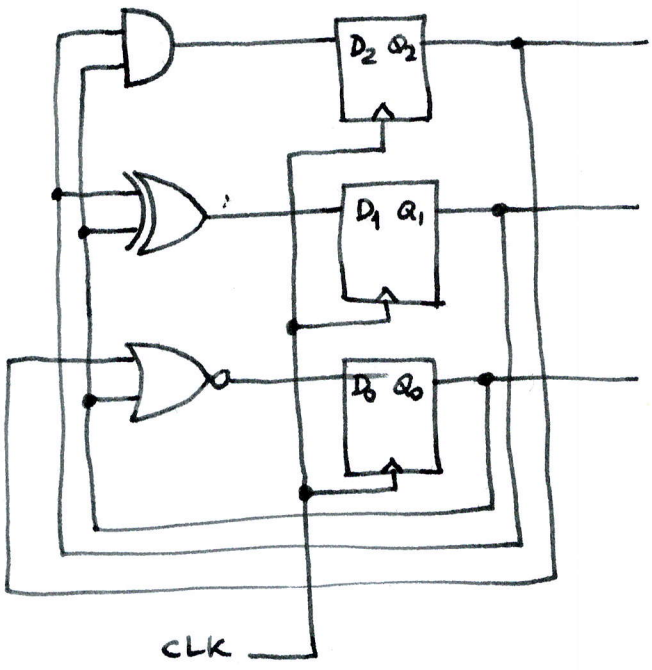


mi manca da progettare $f(Q_2Q_1Q_0)$:

$Q_2Q_1Q_0$	$Q'_2Q'_1Q'_0$
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	0 0 0
1 0 1	X X X
1 1 0	X X X
1 1 1	X X X

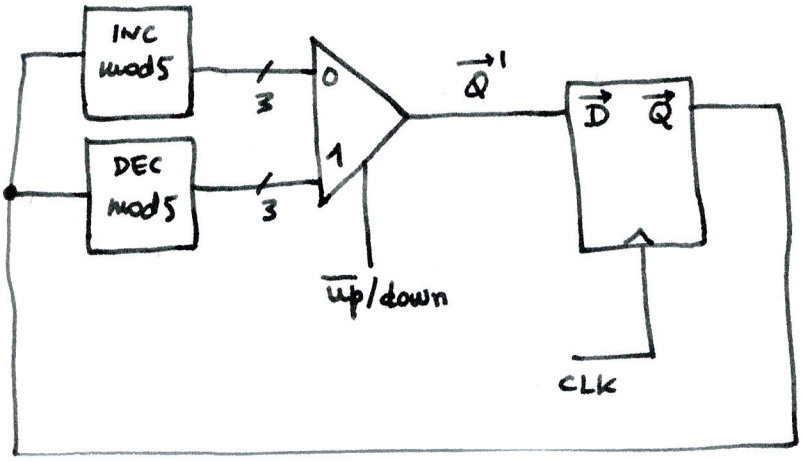
semplificando con Karnaugh, otteniamo:

$$\begin{cases} Q'_2 = Q_1Q_0 \\ Q'_1 = Q_1 \oplus Q_0 \\ Q'_0 = \overline{Q_2 + Q_0} \end{cases}$$

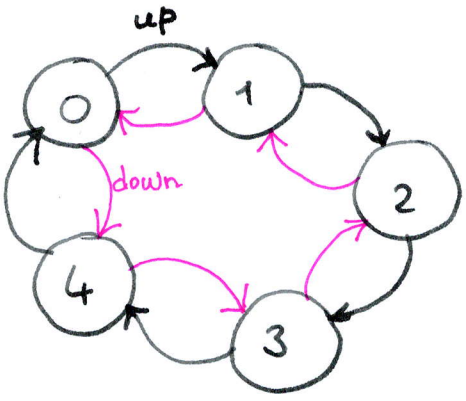


Supp. ora di voler produrre l'uscita 0123432 10 12343210 ....
periodo 8

Una possibile soluzione è quella di estendere il contatore per farlo contare sia up che down:

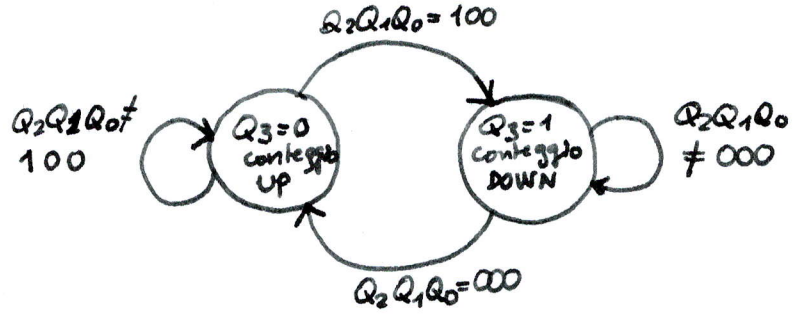


Questa NON è una rete autonoma, perché possiede l'ingresso esterno up/down:



Verrebbe naturale procedere come nel progetto di pag. 2 (detector delle configurazioni 0 e 4 attraverso una funzione g con 0 ↔ up e 4 ↔ down), ma ci accorgiamo che qui non funzionerebbe perché la nostra funzione g deve attribuire un valore al bit up/down anche negli stati intermedi (1, 2 e 3), e in tali stati il valore di questo bit deve poter cambiare, a seconda che il conteggio sia in corso sia up o down: una funzione combinatoria non riesce a far ciò!

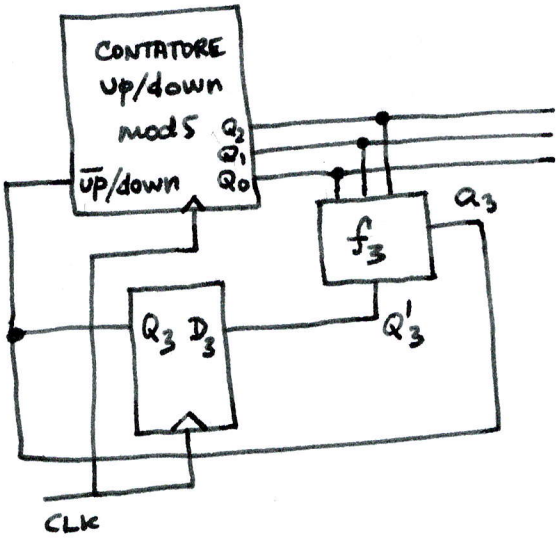
Quindi è necessario mantenere MEMORIA della direzione attuale di conteggio (up o down): è un'informazione che richiede un bit di stato extra fra la macchina:



uscita

Q_3	Q_2	Q_1	Q_0	Q'_3
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	X
0	1	1	0	X
0	1	1	1	X
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

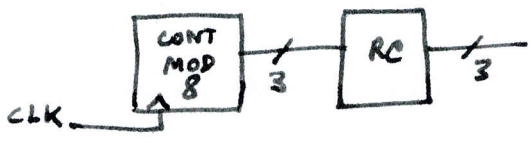
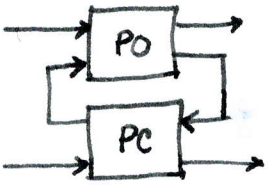
Lo schema complessivo è



Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	0	0	0
01	1	X	X	X
11	1	X	X	X
10	0	1	1	1

$Q'_3 = Q_2 + Q_3Q_1$

NOTA: questa è una rete autonoma, finché l'ingresso $\overline{up}$/down è gestito internamente, essendo generato dal FF aggiuntivo.

NOTA 2: E' questo il primo esempio di una macchina sequenziale composta da due parti, entrambe sequenziali, in comunicazione tra loro. Questo concetto si svilupperà nella rappresentazione PARTE OPERATIVA / PARTE DI CONTROLLO

NOTA 3: Progetto alternativo (che tra l'altro ci consente di risparmiare un FF): uso di un contatore UP (o down) MOD 8 seguito da una rete combinatoria per la produzione delle uscite desiderate: