Sintesi e analisi di schemi con contatori

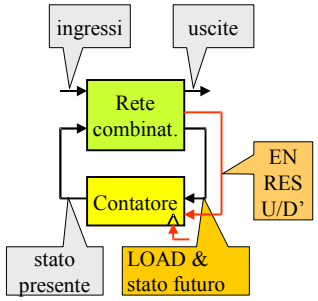
La memorizzazione dello stato interno di una qualsiasi rete sequenziale sincrona può essere affidata ad un contatore: basta infatti sceglierne uno con comando di LOAD, fissare il comando a 1 e disporre il registro di stato così ottenuto in retroazione alla rete che calcola lo stato futuro.

Non è però questo il modo migliore per impiegarlo. Utilizzando infatti opportunamente anche i comandi ENABLE, LOAD, RESET, UP/DOWN per definire lo stato futuro è spesso possibile ottenere realizzazioni più semplici.

Per avere risultati ottimali, l'uso di un contatore come componente primitivo di un progetto deve comunque essere previsto nei primissimi passi del procedimento di sintesi.

La sua presenza nello schema potrà infatti avvenire solo adottando una particolare codifica degli stati e richiederà poi il progetto di una rete combinatoria particolare, atta a generare non più lo stato futuro, come abbiamo visto finora, ma i comandi idonei ad imporlo all'interno del modulo.

Sintesi con contatori



Le cose però sono più semplici di quello che potrebbe sembrare a prima vista. La scelta di usare un contatore è scontata quando si devono realizzare:

- **circuiti di servizio** dedicati a misurare il **trascorrere del tempo** (aspetto che discuteremo nei casi di studio A e B) e ad enumerare il **verificarsi di eventi** (aspetto incluso nell'esercitazione N. 19);
- **circuiti di controllo** di altri circuiti, a cui devono inviare **segnali con forme d'onda periodiche o aperiodiche** (aspetto che discuteremo nel caso di studio C, che si dovrà mettere in conto nell'esercitazione N. 20 e che verrà succintamente esaminato anche a livello architettonico con un "approfondimento").

Come ulteriore preparazione al compito d'esame, l'uso di contatori nella realizzazione di semplici macchine sincrone è infine richiesto nelle esercitazioni N. 21 e N. 22.

CASO DI STUDIO A – Si vuole realizzare un orologio digitale che visualizzi, tramite display a 7 segmenti, le decine e le unità delle ore, dei minuti e dei secondi.

L'unità di tempo deve essere il **secondo**. Per ottenere un clock con frequenza 1 Hz è usuale ricorrere ad un oscillatore a quarzo, molto preciso ma anche di frequenza ben più alta, ed impiegare poi un contatore binario come **divisore di frequenza**.

In figura è previsto un oscillatore con frequenza di 32.768 Hz (2^{15}) ed un contatore binario con base 2^{15}: si noti che l'onda quadra disponibile sul bit di maggior peso (Q_{14}) ha la desiderata frequenza di un Hz.

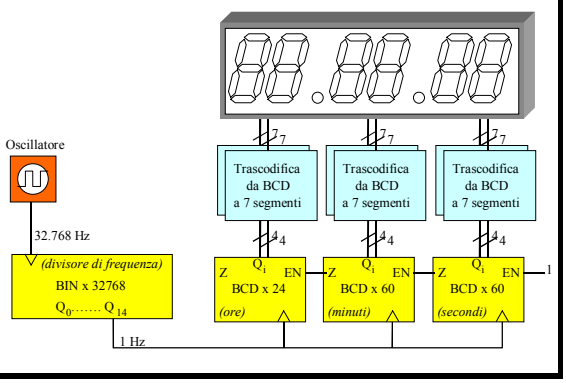
Il tipo di visualizzazione richiesto e la disponibilità di circuiti integrati che trasformano le configurazioni del codice BCD nelle configurazioni del codice a 7 segmenti, suggeriscono di adottare il primo all'interno della macchina.

Il conteggio dei minuti e dei secondi deve dunque essere svolto da due contatori ×60 con stato codificato in BCD, il conteggio delle ore da un contatore ×24, anch'esso con codice BCD.

Tali tre moduli di conteggio devono poi essere disposti in cascata per ottenere una base complessiva

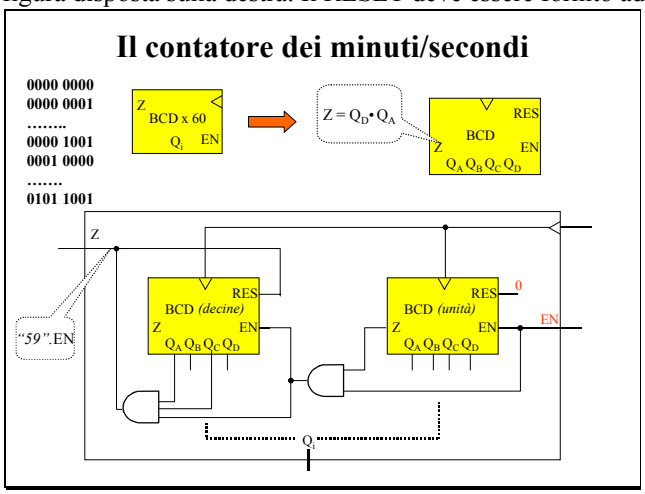
B = 60 × 60 × 24.

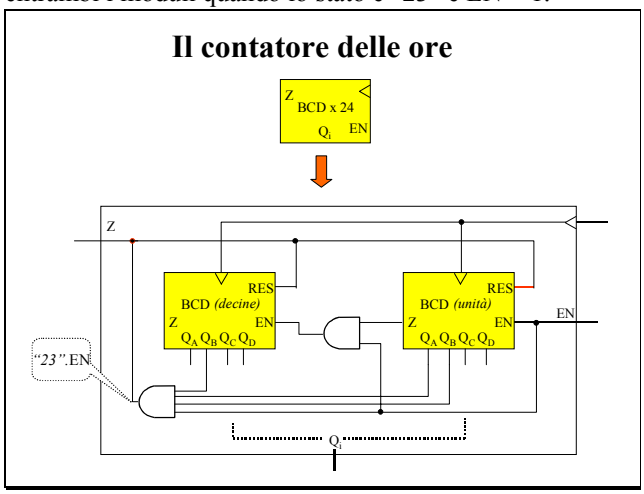
Orologio digitale



Il modulo di conteggio con base 60 è facilmente ottenibile disponendo in cascata due contatori BCD ed attivando il RESET di quello che conta le decine ogniqualvolta è presente lo stato "59" con EN = 1. Lo schema logico è mostrato nella sottostante figura disposta sulla sinistra.

Anche per contare le ore occorrono due contatori BCD disposti in cascata, come evidenziato nella sottostante figura disposta sulla destra. Il RESET deve essere fornito ad entrambi i moduli quando lo stato è "23" e EN = 1.



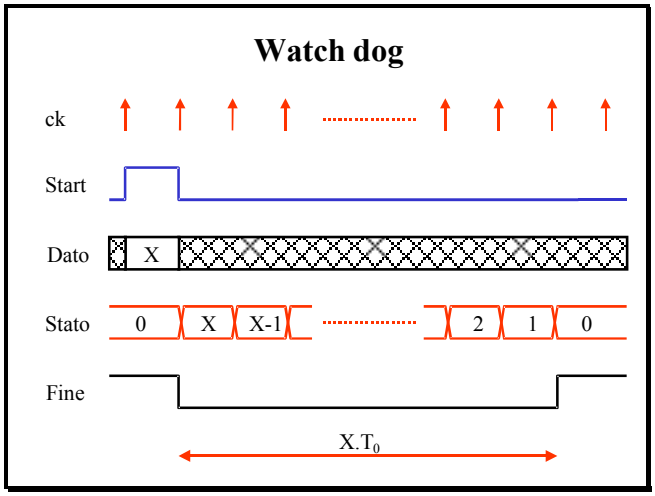


Per sincronizzare manualmente l'orologio, è usuale consentire all'utente, con un primo pulsante, di selezionare le cifre una dopo l'altra e, con un secondo pulsante, di incrementare di quanto serve con il clock a 1 Hz il solo contatore corrispondente alla cifra selezionata (*come si modificano i precedenti schemi?*).

CASO DI STUDIO B – Il comportamento del circuito di servizio **watch dog** (v. cap. 3, pag. 50) è illustrato dalle forme d'onda indicate in figura.

Per ogni Start =1 (il valore per ipotesi può durare un solo intervallo elementare e deve capitare solo quando lo stato è "0"), si deve generare una sequenza di X (il valore trasmesso quando Start=1) transizioni durante le quali il segnale "Fine" mantiene il valore 0.

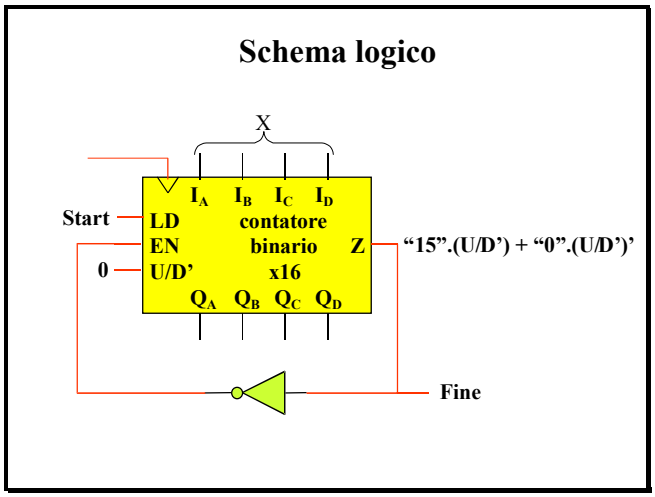
Si noti che nel diagramma a occhio dello stato interno è stata evidenziata la scelta usuale di eseguire un **conteggio all'indietro** a partire dal valore X.



Supponiamo ora, per semplicità, che il numero binario X sia espresso da 4 bit e decidiamo di conseguenza di realizzare la rete a partire da un contatore binario x16, dotato di ENABLE, di LOAD, di UP/DOWN e di un'uscita Z che segnali un prossimo "trabocco" (Z = 1 se lo stato presente è "15" e U/D' = 1 oppure se lo stato è "0" e U/D'= 0).

Per ottenere il circuito che presenta il comportamento desiderato occorre (v figura)

1. predisporre **U/D' = 0** (con ciò si è scelto il conteggio all'indietro),
2. collegare agli ingressi I_A , I_B , I_C , I_D i 4 segnali che trasportano il numero binario **X** da cui si vuole far partire il conteggio,
3. collegare **Start** all'ingresso **LD** (con ciò si riesce a caricare X sui quattro flip-flop del contatore al termine dell'intervallo in cui si verifica **Start = 1**),
4. collegare **Z'** all'ingresso **EN** (con ciò si riesce ad arrestare il conteggio quando il circuito ha di nuovo raggiunto la configurazione di riposo 0000).



CASO DI STUDIO C – Una rete sequenziale sincrona ha un ingresso X che assume il valore 1 molto di rado e comunque sempre per un solo periodo di clock.

L'uscita Z della rete deve sia ritardare l'impulso d'ingresso di quattro unità di tempo, sia raddoppiarne la durata. Le forme d'onda indicate in figura illustrano il comportamento desiderato.

Il tracciamento del grafo è immediato se si pensa che la macchina, con ingresso 0, deve essere sempre pronta a contare cinque intervalli dopo quello in cui si verifica X = 1.

La numerazione degli stati è stata invece scelta pensando già ad una realizzazione con un contatore dotato di ENABLE e di RESET: una volta che i numeri sono stati rappresentati in binario, le transizioni 0→1, 1→2, 2→3, 3→4 richiedono infatti semplici operazioni di incremento; il "salto" conclusivo da 5 a 0 richiede solo l'attivazione del RESET.

In figura è evidenziata la tabella delle transizioni. A lato sono indicati i valori che devono assumere i segnali EN, RES e Z per ottenere il comportamento desiderato in corrispondenza di tutte le situazioni possibili di stato e d'ingresso:

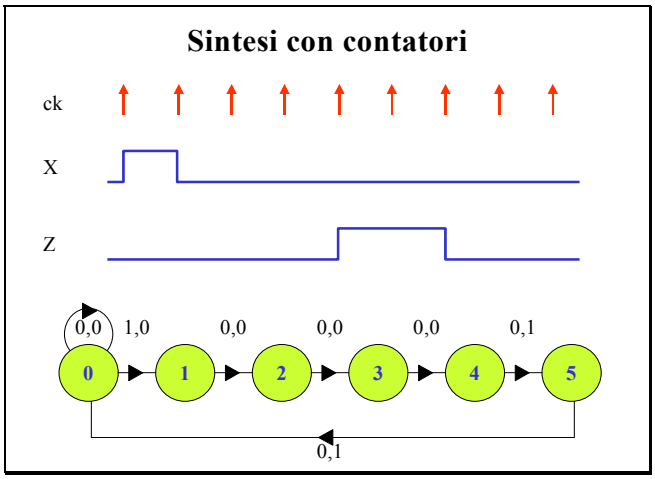
- la stabilità in 0 per X = 0 richiede EN = RES = 0;
- la transizione 0→1 per X = 1 richiede EN=1 e RES=0;
- le transizioni 1→2, 2→3, 3→4 possono avvenire solo nella colonna X=0 e richiedono EN=1 e RES=0;
- anche la transizione 5→0 può avvenire solo nella colonna X=0: per eseguirla è necessario imporre RES=1 (il valore di EN è indifferente, dato che all'interno del contatore il comando di RESET è prioritario rispetto a quello di ENABLE).

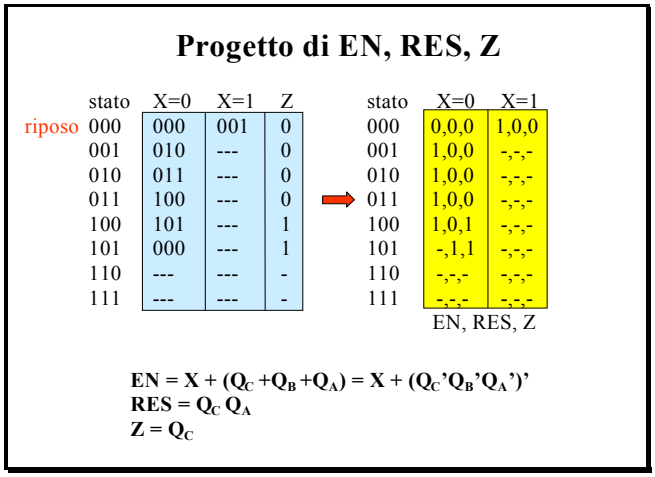
In questo semplice caso le espressioni minime dei tre segnali. possono essere individuate anche senza l'ausilio delle mappe. Deve infatti essere:

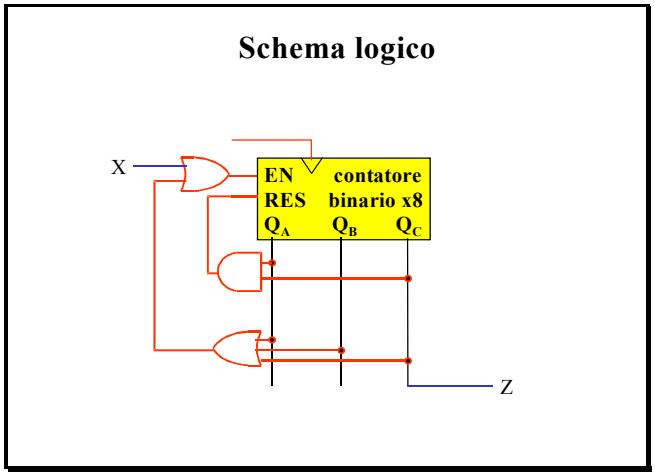
- EN = 1 se X = 1 oppure se lo stato non è 0;
- RES = 1 se lo stato è o 5 o 7 (peraltro impossibile);
- Z = 1 se lo stato è maggiore o uguale a 4.

Lo schema logico della macchina, indicato in figura, richiede un contatore binario x8 in retroazione alla rete che gli genera i comandi EN e RES.

Si noti che in generale un contatore fornisce uscite solo in forma vera: nel calcolo dell'ENABLE conviene dunque impiegare la frase "se X=0 oppure se uno qualsiasi dei bit di stato ha valore 1" (in altre parole conviene trasformare con il teorema di De Morgan il fatto di non essere in presenza della configurazione di tutti "zeri").







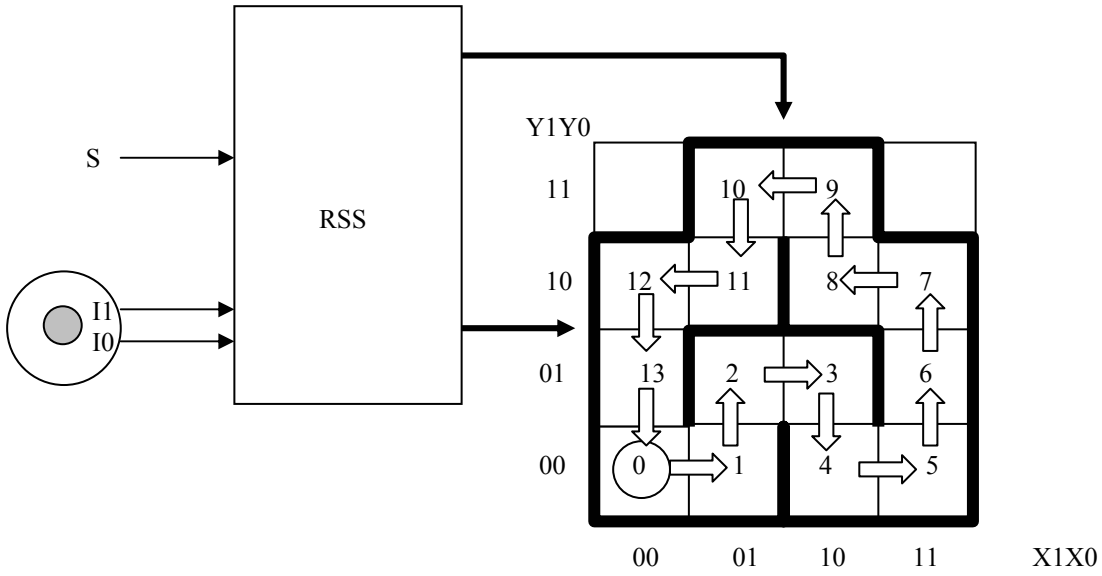
APPROFONDIMENTO – L'unità di controllo di tutti i calcolatori contiene un **contatore** (dotato dei comandi EN, LD e usualmente denominato **Program counter**), che utilizza per reperire in memoria principale l'istruzione da eseguire. Per ipotesi istruzioni da eseguire in sequenza sono alloggiate in celle della memoria il cui indirizzo differisce per un'unità.

Durante la fase di fetch (v. pag. 4), l'unità di controllo dapprima incrementa il PC di un'unità (EN = 1), poi avvia un ciclo di lettura della memoria principale trasmettendo il contenuto del PC come indirizzo (v. pag. 110).

Una volta ottenuta l'istruzione alloggiata in quella cella, l'unità di controllo ne controlla il "formato" per decidere cosa occorre fare durante la fase di execute: il linguaggio di macchina prevede infatti sia istruzioni che gestiscono il Data Path, sia istruzioni di "salto" che riguardano il solo Controller e che contengono l'indirizzo da impiegare per accedere all'istruzione successiva. In questo secondo caso, se sussistono le condizioni (il salto può essere infatti condizionato dal valore di uno dei flag), l'unità di controllo carica il nuovo indirizzo sul PC (LD = 1) e lo comunica alla memoria.

Da questa succinta descrizione ci interessa qui trarre una conclusione: per realizzare una macchina sequenziale sincrona "programmabile" sono sufficienti **una memoria** (RAM o ROM) ed **un contatore** che la indirizza.

Esercitazione N. 19



Il videogioco di figura prevede che un punto mobile su una griglia di sedici caselle segua il percorso indicato senza arretrare o andare a sbattere contro le pareti.

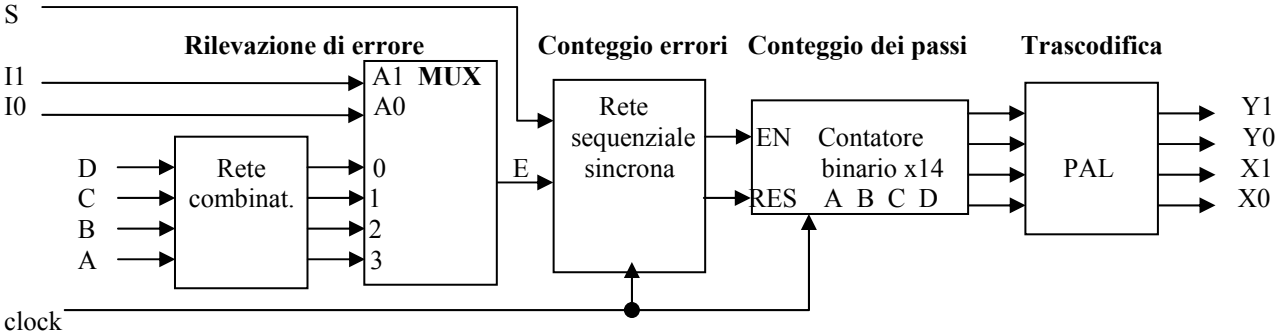
A gioco fermo il punto resta nella casella 0 (coordinate Y1= 0, Y0 = 0, X1 = 0, X0 = 0*).*

Nell'intervallo successivo a quello in cui il segnale sincrono S assume il valore 1 (per ipotesi tale valore si mantiene per un solo intervallo di clock e si può presentare solo a gioco fermo) la RSS che controlla il gioco comincia a prendere atto dei comandi di spostamento del punto che il giocatore invia tramite il joystick. I comandi possibili sono quattro e sono codificati dai due bit I1,I0*:*

00 = destra, 10 = sinistra, 11 = alto, 01 = basso.

Se il comando è corretto, il punto si sposta nella posizione successiva del percorso; se è errato, e gli errori non sono stati più di due, il punto viene riportato nella posizione iniziale. Al terzo errore il gioco si arresta.

La RSS deve essere decomposta come indicato in figura.



La rete combinatoria a due stadi denominata "rilevazione di errore" esamina la codifica binaria della posizione corrente del punto (fornita dai bit D,C,B,A *del contatore adibito al "conteggio dei passi":* A *è il bit di minor peso) ed i valori attuali di* I1, I0 *al fine di generare un segnale di errore* E*:* **E = 0** ***se il comando è appropriato,*** **E = 1** ***se il comando è errato****.*

La rete sequenziale sincrona denominata "conteggio errori" esamina i valori di S, E ed indica al contatore binario per 14, tramite i comandi EN *e di* RES*, se il punto deve o stare fermo nella casella 0 in attesa dell'evento* S = 1*, o andarvi per punizione, o avanzare di una posizione lungo il percorso.*

I bit D, C, B, A *vanno anche ad una PAL che genera le coordinate del punto sullo schermo.*

DOMANDA N.1 –Progettare le quattro reti minime a NAND da mettere a monte del MUX.

	B A			
D C	00	01	11	10
.00				
01				
11				
10				

Z_0 =

	B A			
D C	00	01	11	10
00				
01				
11				
10				

Z_1 =

	B A			
D C	00	01	11	10
00				
01				
11				
10				

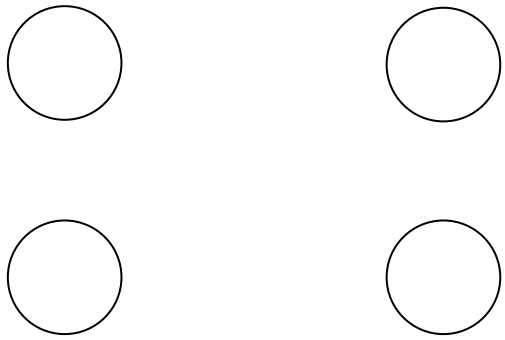
Z_2 =

	B A			
D C	00	01	11	10
00				
01				
11				
10				

Z_3 =

DOMANDA N.2 – Individuare grafo e tabella di flusso della rete sequenziale sincrona che genera EN *e* RES

S E,EN RES



	S E			
s^n	00	01	11	10
A				
B				
C				
D				

s^{n+1}, EN RES

DOMANDA N.3 -Dimostrare che la realizzazione dei segnali EN *e* RES *richiede in tutto un OR a 2 ingressi ed un AND a due ingressi.*

	S E			
$y_1^n\ y_2^n$	00	01	11	10
00				
01				
11				
10				

EN

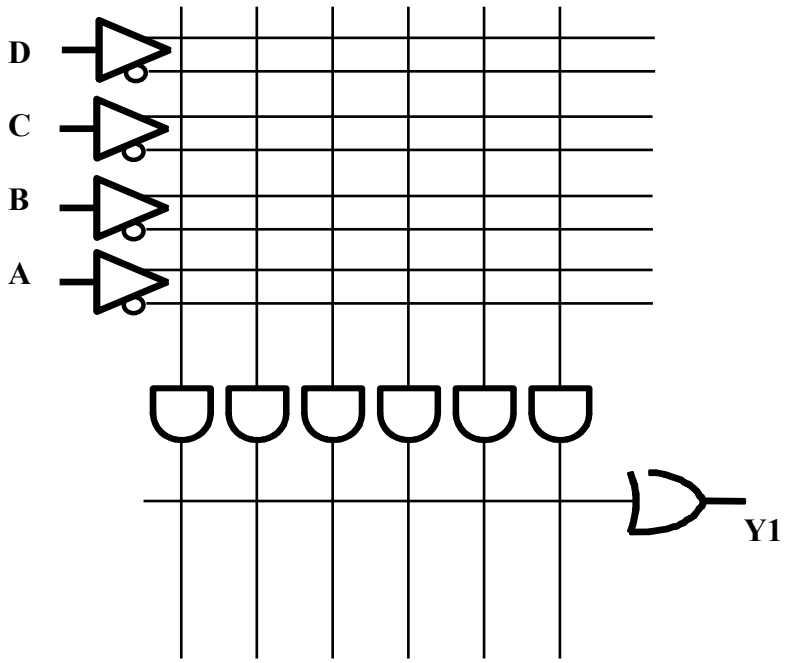
	S E			
$y_1^n\ y_2^n$	00	01	11	10
00				
01				
11				
10				

RES

DOMANDA N.4 – Individuare il numero degli AND necessari per generare il segnale Y1 *e programmare la PLA di figura.*

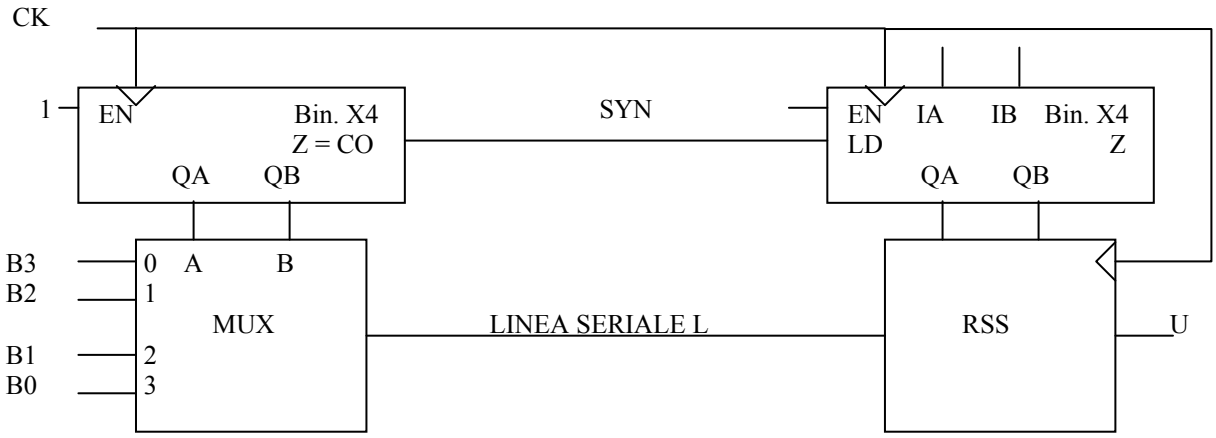
	B A			
D C	**00**	**01**	**11**	**10**
00				
01				
11				
10				

Y1



Esercitazione N. 20

Le unità di controllo del trasmettitore e del ricevitore di una linea seriale sono azionate dallo stesso clock, impiegano entrambe un contatore binario per 4 e si mantengono "al passo" con il segnale SYN *(v. figura).*



DOMANDA N.1 - Quale configurazione binaria occorre predisporre sugli ingressi EN, IA, IB *per garantirsi che il contatore in ricezione presenti lo stesso stato di quello in trasmissione?*_____

DOMANDA N.2 - Supponendo che il trasmettitore invii numeri binari di 4 bit uno dietro l'altro (il bit più pesante quando il contatore è nello stato "zero", il più leggero quando è nello stato "tre"), quale grafo a due stati consente a RSS di attribuire a U *il valore 1 in corrispondenza del bit più leggero e solo se il numero appena ricevuto è 3?*

DOMANDA N.3 - Tracciare la tabella delle transizioni di RSS.

DOMANDA N.4 - Individuare la funzione di eccitazione del FF JK necessario per la realizzazione di RSS.

DOMANDA N.5 – Tracciare lo schema logico di RSS.

Esercitazione N. 21

La tabella di flusso a lato indica il comportamento richiesto ai semafori per la strada a senso unico alternato (v. Esercitazione N. 4).

Si deve realizzarla con una rete combinatoria a Nand retroazionata da un contatore binario x4 dotato di comandi EN *e* U/D'.

	(x1x2)^n			
Q^n	00	01	10	11
A	A,00	C,01	B,10	B,10
B	D,00	D,00	B,10	D,00
C	A,00	C,01	A,00	A,00
D	D,00	C,01	B,10	C,01

Q^{n+1}, $(s1s2)^n$

	$(Q_BQ_A)^n$	(x1x2)^n 00	01	10	11
A					
B					
C					
D					

EN^n

	$(Q_BQ_A)^n$	(x1x2)^n 00	01	10	11
A					
B					
C					
D					

U/D'^n

	$(Q_BQ_A)^n$	(x1x2)^n 00	01	10	11
A					
B					
C					
D					

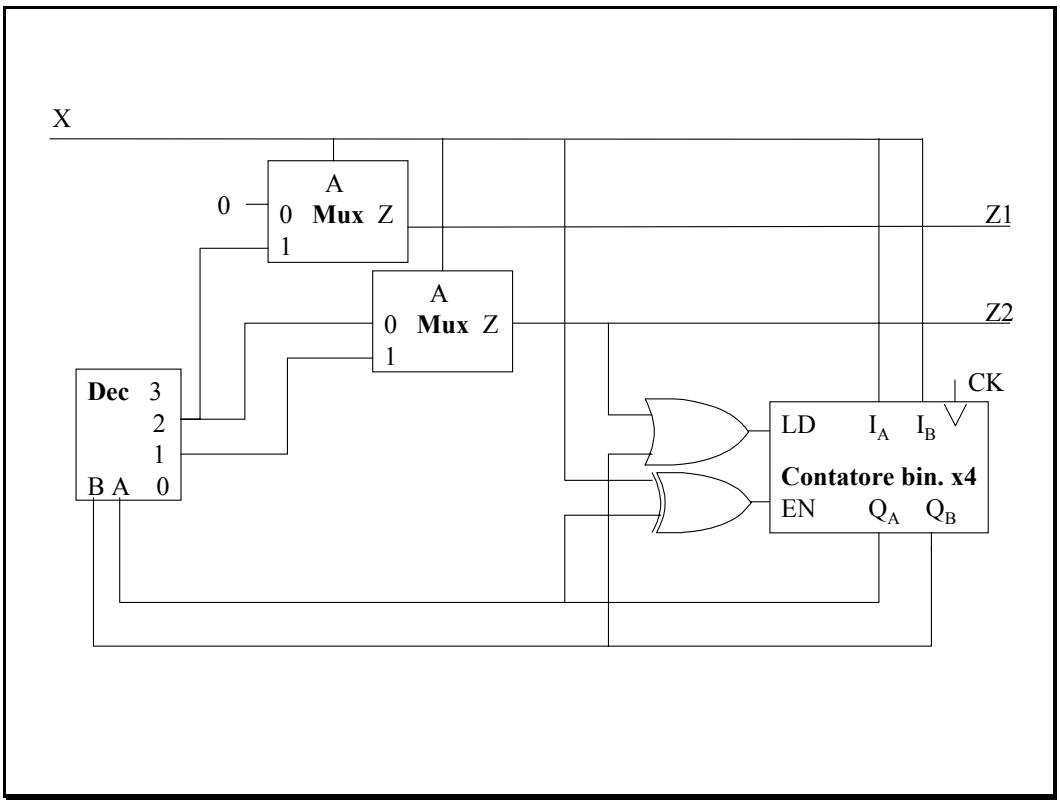
$s1^n$

	$(Q_BQ_A)^n$	(x1x2)^n 00	01	10	11
A					
B					
C					
D					

$s2^n$

Esercitazione N. 22

Una rete logica sequenziale sincrona ha la struttura indicata in figura:



DOMANDA N. 1 – Individuare le espressioni SP dei segnali d'uscita (Z1,Z2) e dei segnali di aggiornamento dello stato interno (EN, LD, I_A, I_B).

Z1 =

Z2 =

EN =

LD =

$I_A = I_B =$

DOMANDA N. 2 – Riportare le precedenti funzioni sulle due prime mappe e dedurre dalla prima come deve essere riempita la terza mappa.

$Q_B\ Q_A$	X=0	X=1
00		
01		
11		
10		

$(EN, LD, I_B, I_A)^n$

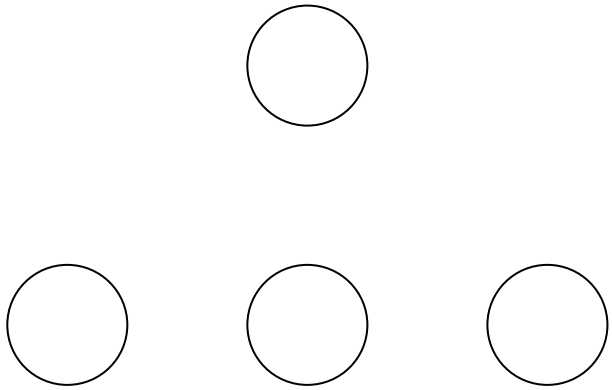
$Q_B\ Q_A$	X=0	X=1
00		
01		
11		
10		

$(Z1, Z2)^n$

$Q_B\ Q_A$	X=0	X=1
00		
01		
11		
10		

$(Q_B, Q_A)^{n+1}$

DOMANDA N. 3 – Tracciare il grafo degli stati



DOMANDA N. 4 – Dimostrare che è possibile eliminare dallo schema l'OR che genera il comando LD senza modificare il comportamento della rete.

Registri a scorrimento

Nelle famiglie logiche sono stati resi disponibili registri a scorrimento (***shift register***) formati da un certo numero di flip-flop. L'ingresso del primo flip-flop della cascata è usualmente denominato **SI** (***Serial Input***); l'uscita dell'ultimo flip-flop è denominata **SO** (***Serial Output***).

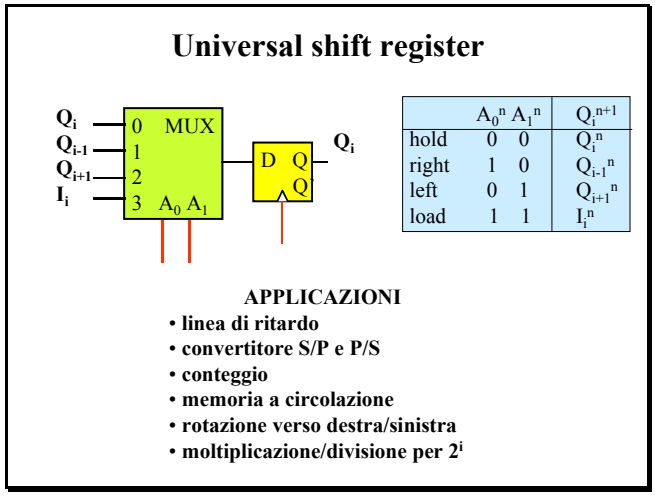
La disponibilità di un componente primitivo di questo tipo è utile quando si deve **ritardare** da **1** a **k** intervalli la forma d'onda di un segnale qualsiasi, **riconoscere** il verificarsi di stringhe d'ingresso di lunghezza **k, attribuire** la forma parallela ad un dato ricevuto in forma seriale.

Altre applicazioni riguardano il **conteggio**, la **conversione P/S**, la **rotazione di una stringa** e la **moltiplicazione/divisione** di un numero binario per una potenza di 2.

Per coprirle tutte è stato predisposto il cosiddetto ***universal shift register***, un circuito a 4 bit in cui ogni flip-flop della cascata è preceduto da un MUX avente in ingresso (v. figura a lato) la sua uscita (Q_i), quella del precedente (Q_{i-1}), quella del successivo (Q_{i+1}) ed un bit esterno (I_i).

Tramite i due bit d'indirizzo è possibile ottenere quattro differenti comportamenti:

- mantenimento dello stato ($A_0 = 0, A_1 = 0$)
- scorrimento verso destra ($A_0 = 1, A_1 = 0$)
- scorrimento verso sinistra ($A_0 = 0, A_1 = 1$)
- caricamento di un dato ($A_0 = 1, A_1 = 1$)

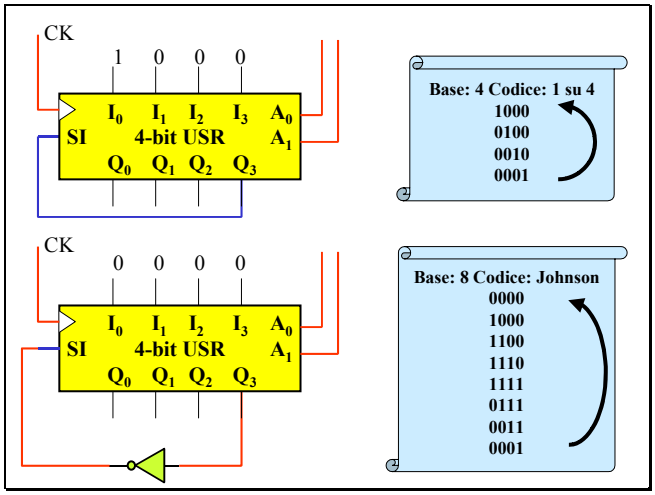


CASO DI STUDIO – Un ciclo di conteggio in base 4 può essere ottenuto facendo continuamente scorrere un solo "uno" dal primo all'ultimo flip-flop: si parla in questo caso di **contatore ad anello**.

Per realizzarlo è sufficiente collegare SO con SI. Occorre però essere sicuri che gli stati del ciclo siano realmente configurazioni del codice "uno su quattro" ed è quindi prudente inizializzare il circuito con una di queste (in figura 1000).

Un ciclo di conteggio in base 8 può essere ottenuto introducendo degli "uni" nel registro fino a quando non ne è tutto pieno ed introducendo poi degli "zeri" fino a quando non ne è tutto pieno: il contatore è detto di **Johnson** o **a riempimento/svuotamento**.

Per realizzarlo occorre collegare a SI il complemento di SO. Anche in questo caso bisogna preoccuparsi della corretta inizializzazione del circuito.



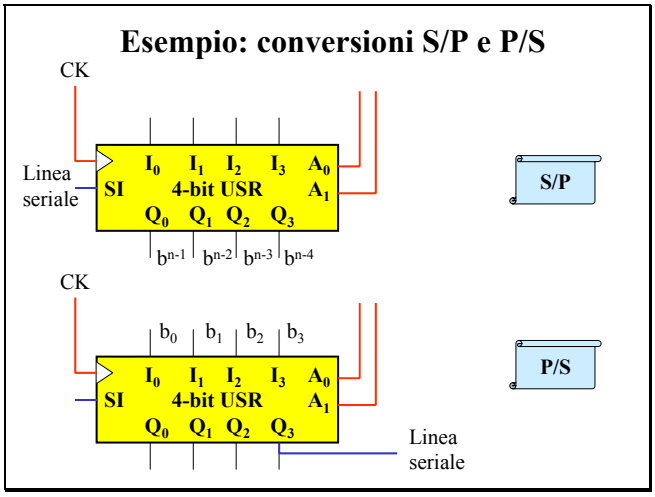
CASO DI STUDIO – La conversione S/P richiede la connessione a SI della linea seriale.

Se la trasmissione è continua, una apposita unità di controllo (in definitiva un contatore in base 4) deve preoccuparsi di trasferire su un registro buffer ogni quaterna di bit ricevuti.

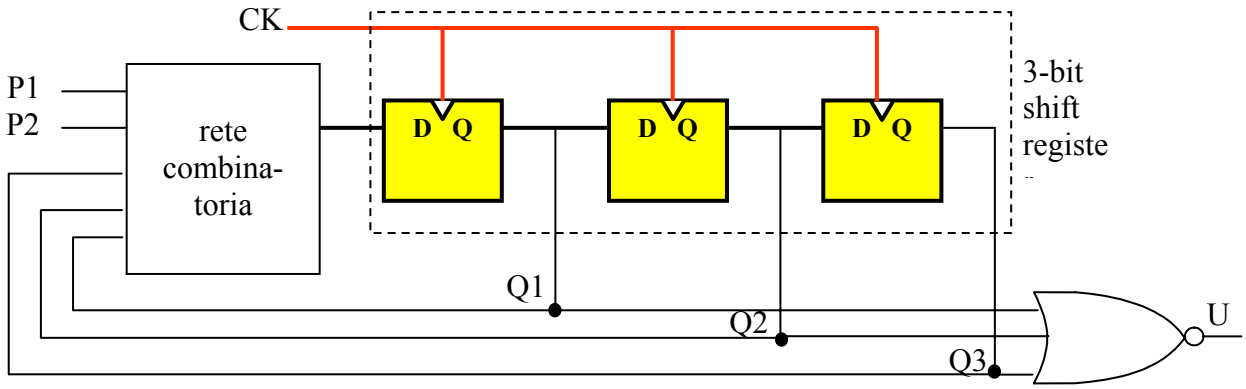
Se la trasmissione è basata su pacchetti di 4 bit intercalati da pause, l'unità di controllo deve solo mettere il registro in hold durante le pause.

Nella conversione P/S occorre prima introdurre nel circuito un pacchetto di 4 bit e poi inoltrare un bit alla volta sulla linea seriale collegata a SO:

Anche in questo caso occorre una apposita unità di controllo, che gestisca il funzionamento di questo trasmettitore.



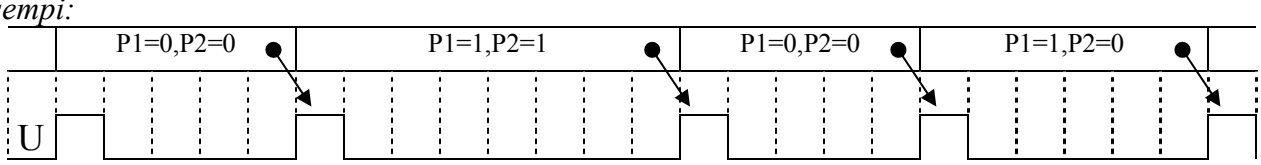
Esercitazione N. 23



La rete sequenziale sincrona riportata in figura deve produrre un segnale U *che assume valore* **1** *per un solo periodo di clock, e vale poi* **0** *per un numero di periodi programmabile mediante gli ingressi* P1 *e* P2. *I due segnali* P1 *e* P2 *possono modificare il loro valore* <u>*solo*</u> *nell'istante di sincronismo che segna l'inizio dell'intervallo in cui l'uscita* U *assume il valore* 1. *I comportamenti possibili sono:*

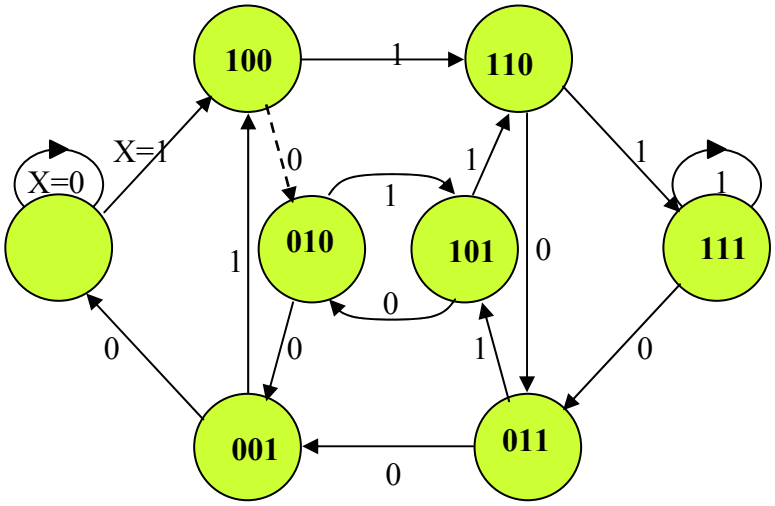
1. *quando* P1=0, P2=0 *il successivo* U= 1 *deve verificarsi dopo 5 intervalli di clock*
2. *quando* P1=1, P2=0 *il successivo* U= 1 *deve verificarsi dopo 6 intervalli di clock*
3. *quando* P1=0, P2=1 *il successivo* U= 1 *deve verificarsi dopo 7 intervalli di clock*
4. *quando* P1=1, P2=1 *il successivo* U= 1 *deve verificarsi dopo 8 intervalli di clock*

Esempi:



DOMANDA N.1 - Dedurre quali percorsi (sequenze di stati) sul grafo degli stati del 3-bit-shift-register consentono di ottenere i comportamenti d'uscita desiderati e quali sequenze d'ingresso occorre fornire allo shift register per seguire tali percorsi.

<u>*E' vietato impiegare la transizione corrispondente al ramo tratteggiato*</u>.



P1 P2 stato iniziale e stati successivi **sequenza d'ingresso**

0 0 : ____-____-____-____-____-____*e ritorno a* ____ __-__-__-__-__-__

1 0 : ____-____-____-____-____-____-____*e ritorno a* ____ __-__-__-__-__-__-__

0 1 : ____-____-____-____-____-____-____-____ *e ritorno a* ____ __-__-__-__-__-__-__-__

1 1 : ____-____-____-____-____-____-____-____-____ *e ritorno a* ____ __-__-__-__-__-__-__-__-__

DOMANDA N. 2 - Indicare sulla mappa la funzione X = F(P1,P2,Q1,Q2,Q3) *ed individuarne l'espressione minima a NAND.*

	P1P2			
Q2Q3	00	10	11	01
00				
01				
11				
10				

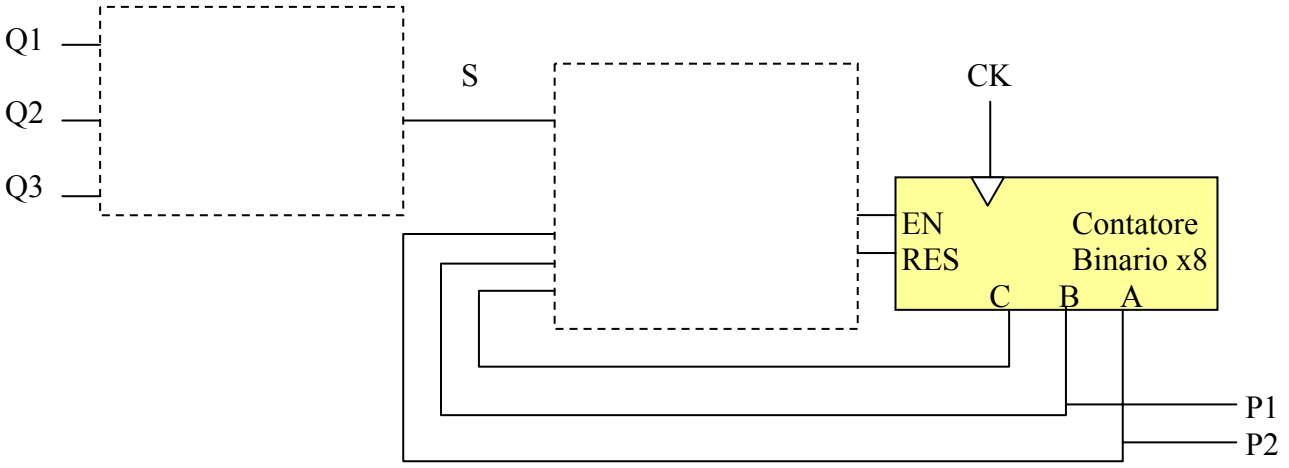
Q1 = 0

	P1P2			
Q2Q3	00	10	11	01
00				
01				
11				
10				

Q1 = 1

X = F(P1,P2,Q1,Q2,Q3) =

DOMANDA N.3 – Si supponga di voler affidare la generazione di una sequenza di valori dei segnali P1, P2 ad un ***contatore binario x5*** *realizzato dai tre blocchi indicati in figura.*



Il sistema deve rispettare il vincolo dato in precedenza, per cui le uscite A,B *del modulo di conteggio devono poter variare solo in corrispondenza dell'istante di sincronismo che segna l'inizio dell'intervallo in cui il registro a scorrimento ha* U=1. *Il primo blocco riceve* Q1,Q2,Q3 *(le uscite dei flip-flop dello shift-register) e genera a questo scopo un segnale S. Il secondo blocco deve generare i comandi* EN, RES *tenendo conto di* S *e riducendo a* 5 *la base del modulo di conteggio (dato un contatore in base* 8*).*

Completare con reti AND,OR, NOT <u>di costo minimo</u> lo schema logico di figura.
*Giustificazione del progetto di S:*_____

Giustificazione del progetto di EN*:*

Giustificazione del progetto di RES*:*_____