
Reti
sequenziali
sincrone

Esercizio 1

Gli accessi da parte di due processori P_1 e P_2 ad una risorsa condivisa sono coordinati da una rete sequenziale sincrona (*arbitro*), caratterizzata da 2 segnali di ingresso R_1, R_2 (richiesta di accesso da parte di P_1 e P_2) e da 2 segnali di uscita C_1, C_2 (consenso all'accesso per P_1 e P_2).

In assenza di richieste di accesso, l'arbitro deve mantenere disattivi (livello logico 0) sia C_1 che C_2.

Se un processore presenta una richiesta di accesso allorché la risorsa condivisa è libera, l'arbitro deve fornire il relativo consenso, confermandolo in seguito per tutto il tempo di attivazione della richiesta stessa.

Se un processore presenta una richiesta di accesso allorché la risorsa non è libera (*conflitto di accesso*), l'arbitro deve fornire il relativo consenso solo al termine dell'accesso in corso.

Richieste di accesso contemporanee da parte di P_1 e P_2, infine, debbono essere gestite dall'arbitro in maniera tale da privilegiare il processore che per ultimo ha dovuto attendere a seguito di un conflitto di accesso.

Nell'ipotesi che P_1 e P_2, una volta attivata una richiesta di accesso, rimangano comunque in attesa del relativo consenso confermando la richiesta stessa, si determini:

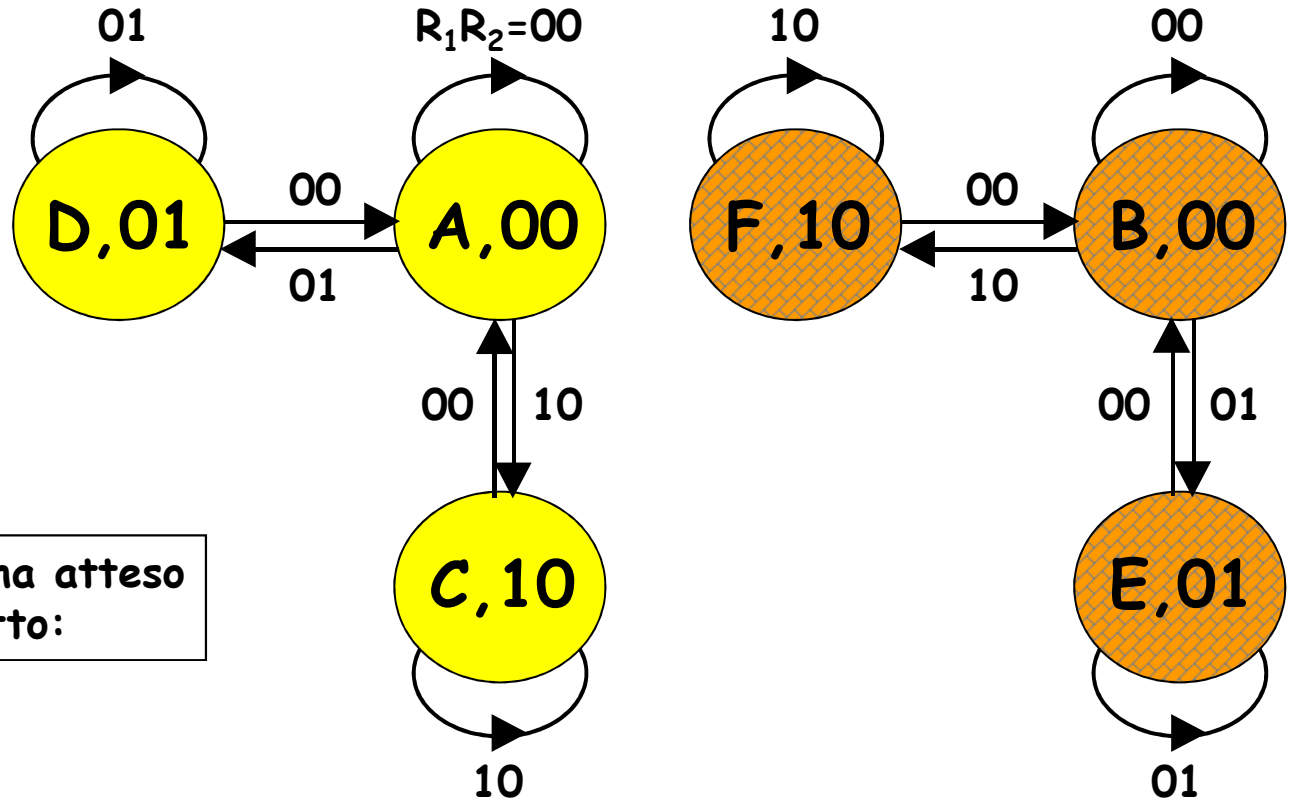
- l'automa minimo dell'arbitro secondo il modello di Moore;
- una possibile realizzazione mediante FF-JK e gate elementari.

Diagramma degli stati (modello di Moore)

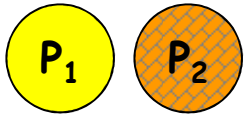
Il diagramma può essere costruito a partire dalla situazione corrispondente a risorsa libera e richieste di accesso disattive (R_1R_2=00). Occorre prevedere due distinti stati, entrambi caratterizzati da C_1C_2=00, per discriminare il processore che per ultimo ha atteso a seguito di un conflitto.

A fronte di una sola richiesta di accesso, da ciascuno stato si transiterà verso una nuova coppia di stati, l'uno caratterizzato da C_1C_2=10 (attivazione di R_1), l'altro da C_1C_2=01 (attivazione di R_2).

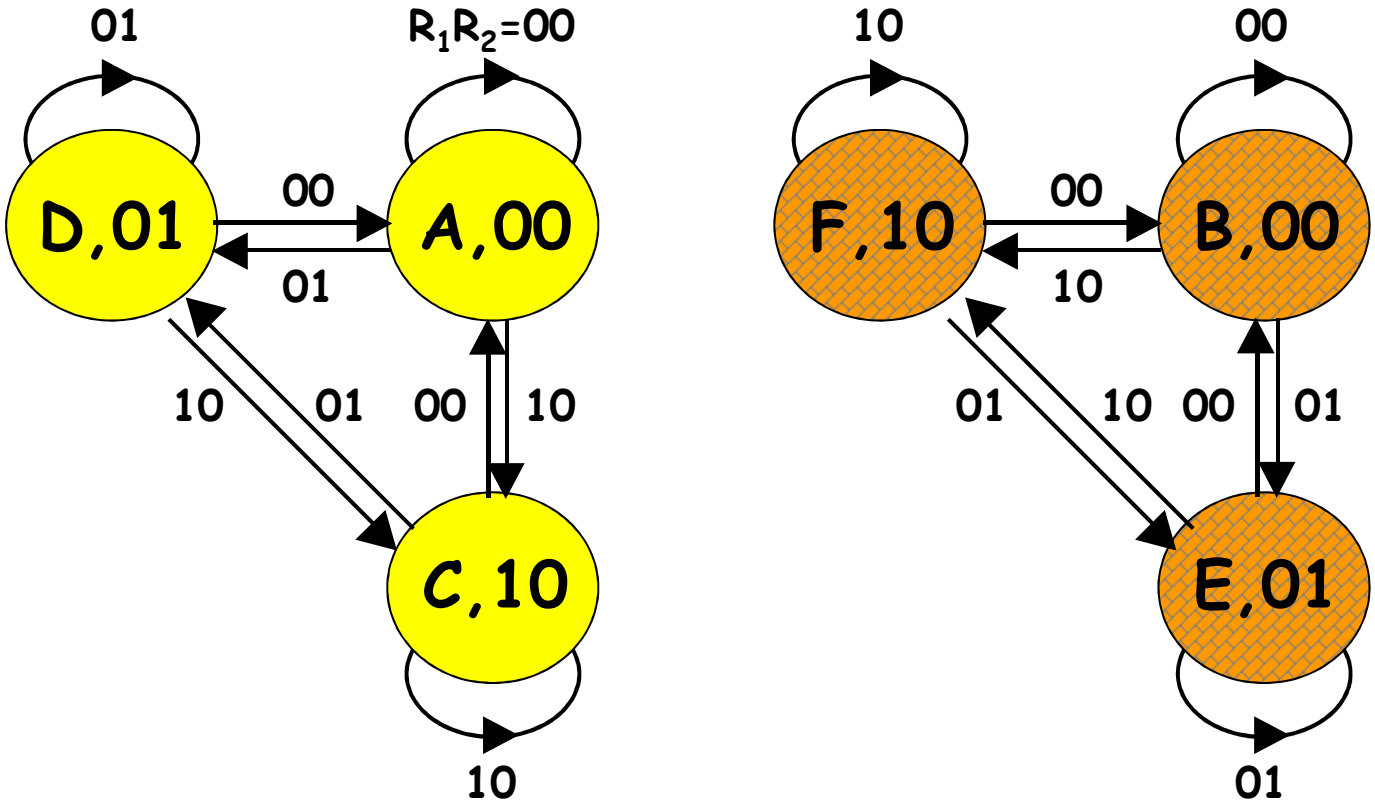
Terminato l'accesso (R_1R_2=00), da ogni stato si effettuerà poi la transizione opposta, non essendosi verificato per ipotesi un nuovo conflitto.



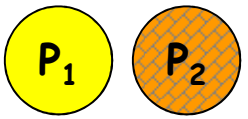
processore che per ultimo ha atteso a seguito di un conflitto:



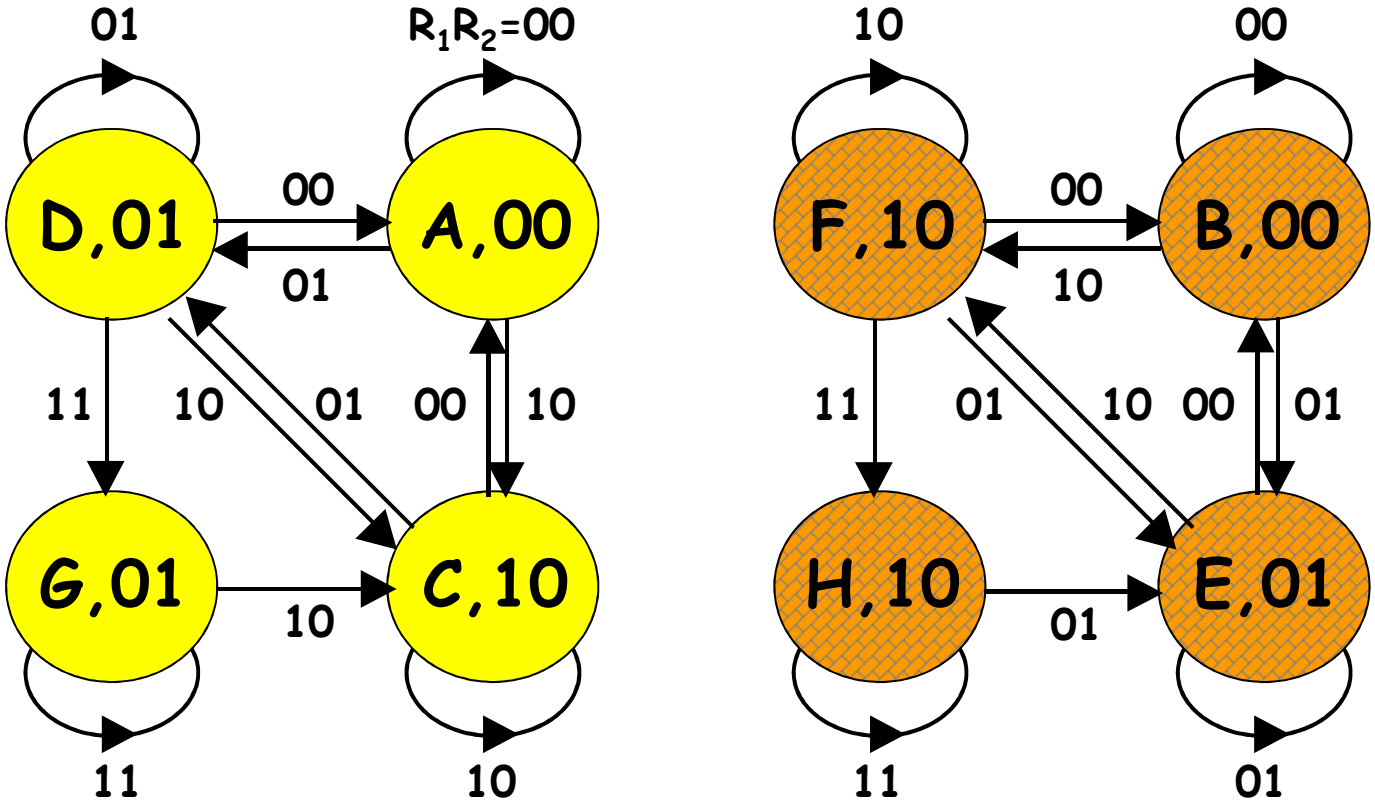
Non si ha conflitto anche nel caso in cui la richiesta di accesso alla risorsa condivisa da parte di un processore si manifesta nello stesso intervallo di clock in cui termina l'accesso da parte dell'altro processore. In tal caso il processore che per ultimo ha dovuto attendere a seguito di un conflitto di accesso continua ad essere quello precedentemente assunto.



processore che per ultimo ha atteso a seguito di un conflitto:



Nel caso in cui la risorsa sia già occupata e si manifesti una richiesta di accesso da parte dello stesso processore che in precedenza ha dovuto attendere a seguito di un conflitto, occorre ritardare l'attivazione del relativo consenso fino al completamento dell'accesso in corso, e confermare tale processore come l'ultimo cui è stato temporaneamente negato l'accesso.

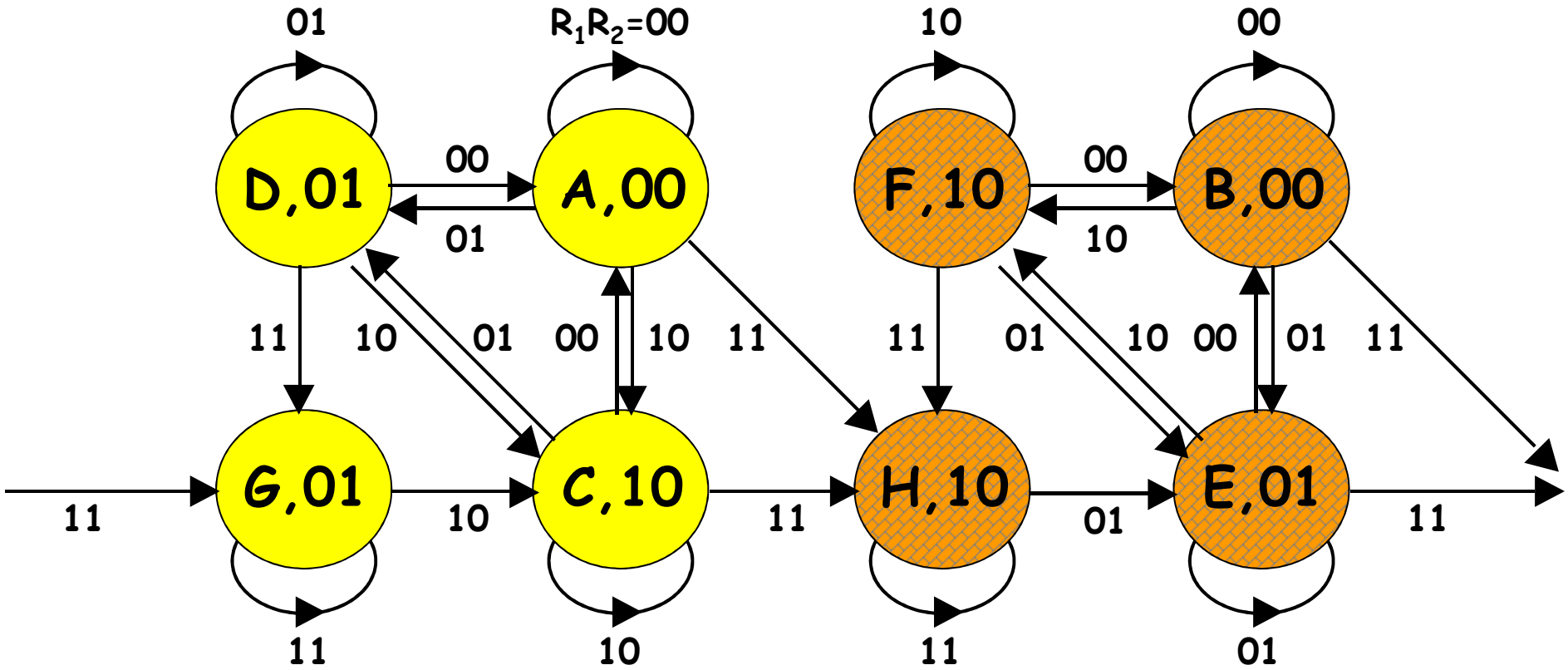


processore che per ultimo ha atteso a seguito di un conflitto:

P_1 P_2

Similmente, nel caso in cui la risorsa sia già occupata e la richiesta di accesso pervenga da parte dell'altro processore, occorre aggiornare lo stato in modo tale da riflettere quest'ultimo conflitto.

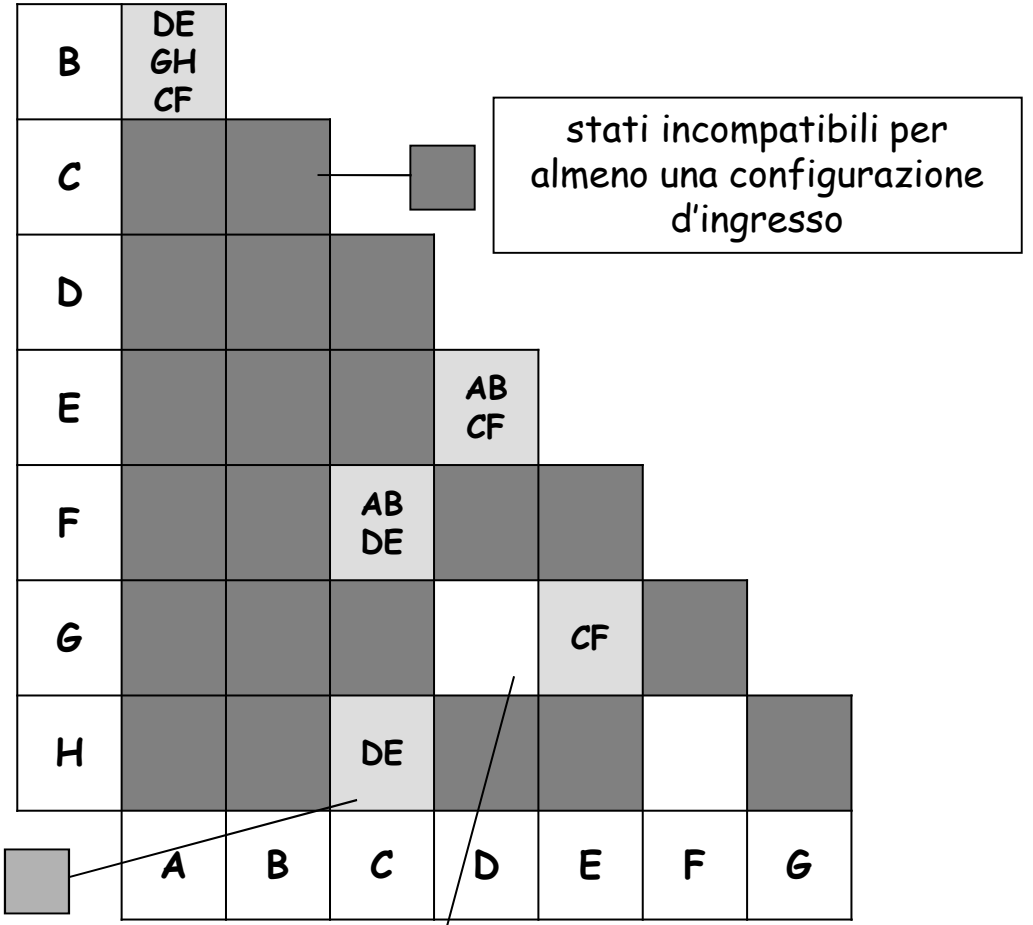
Occorre infine gestire la contemporanea attivazione di R_1 e R_2 allorché la risorsa è libera. La precedenza nell'accesso deve essere attribuita al processore penalizzato nell'ultimo conflitto.



processore che per ultimo ha atteso a seguito di un conflitto: P_1 P_2

Tabella di flusso e tabella triangolare

	R_1R_2 = 00	01	11	10	C_1C_2
A	A	D	H	C	00
B	B	E	G	F	00
C	A	D	H	C	10
D	A	D	G	C	01
E	B	E	G	F	01
F	B	E	H	F	10
G	-	-	G	C	01
H	-	E	H	-	10



Classi massime di compatibilità

{A}, {B}, {C}, {D,G}, {E}, {F,H}

Tabella di flusso minima

	R_1R_2 00	01	11	10	C_1C_2
A	A	D	F	C	00
B	B	E	D	F	00
C	A	D	F	C	10
{D,G} D	A	D	D	C	01
E	B	E	D	F	01
{F,H} F	B	E	F	F	10

Tabella delle transizioni

$(y_0y_1y_2)^n$	$(R_1R_2)^n$ 00	01	11	10	$(C_1C_2)^n$
000	000	001	110	010	00
001	000	001	001	010	01
011	---	---	---	---	--
010	000	001	110	010	10
100	100	101	001	110	00
101	100	101	001	110	01
111	---	---	---	---	--
110	100	101	110	110	10

$(y_0y_1y_2)^{n+1}$

Mappa di codifica

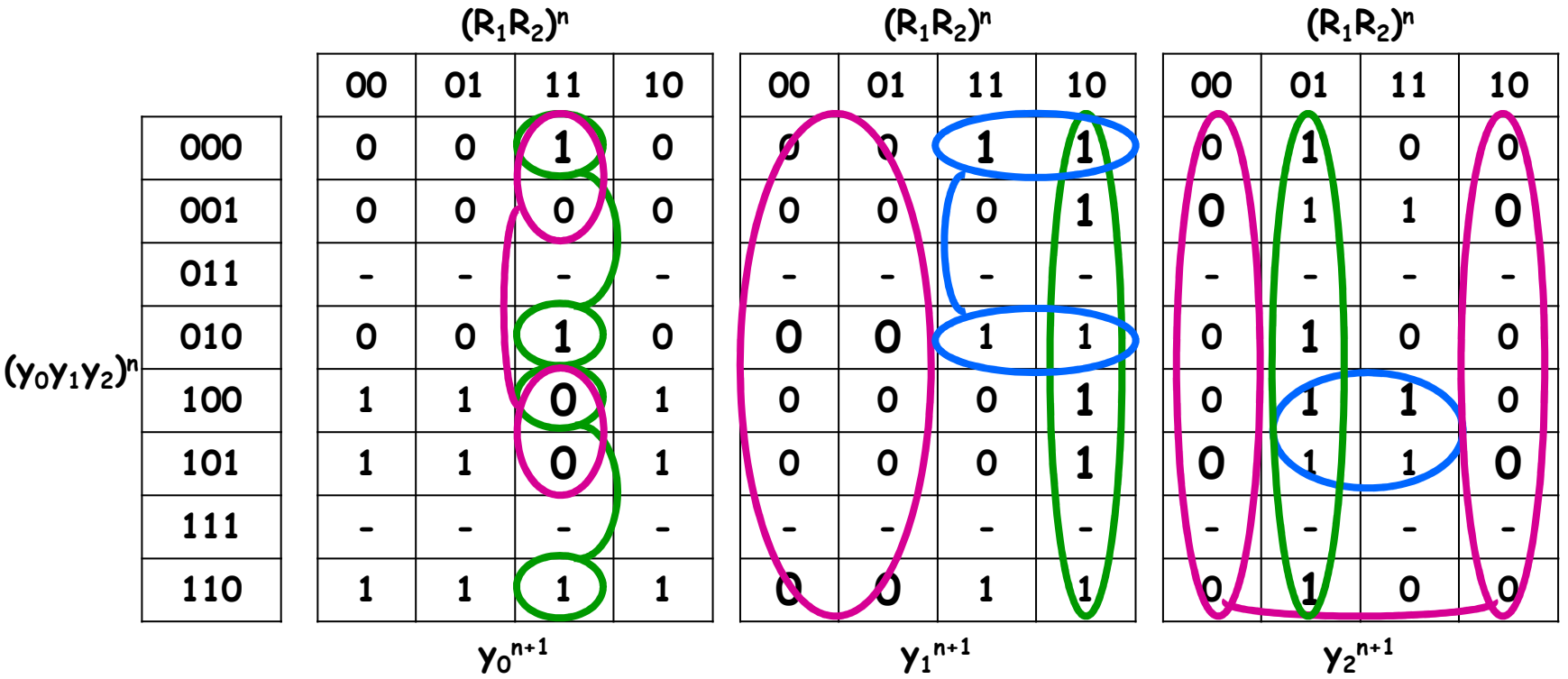
y_0	y_1y_2 00	01	11	10
0	A	D	-	C
1	B	E	-	F

priorità di accesso: $y_0=0$ P_1 $y_0=1$ P_2

Rete combinatoria di uscita

$C_1^n = y_1^n$ $C_2^n = y_2^n$

Rete combinatoria di aggiornamento dello stato



$J_0^n = (R_1R_2\,y_2')^n$ $\quad J_1^n = (R_1R_2' + R_1y_2'y_0')^n$ $\quad J_2^n = (R_2R_1' + R_2y_1'y_0)^n$

$K_0^n = (R_1R_2\,y_1')^n$ $\quad K_1^n = (R_1')^n$ $\quad K_2^n = (R_2')^n$

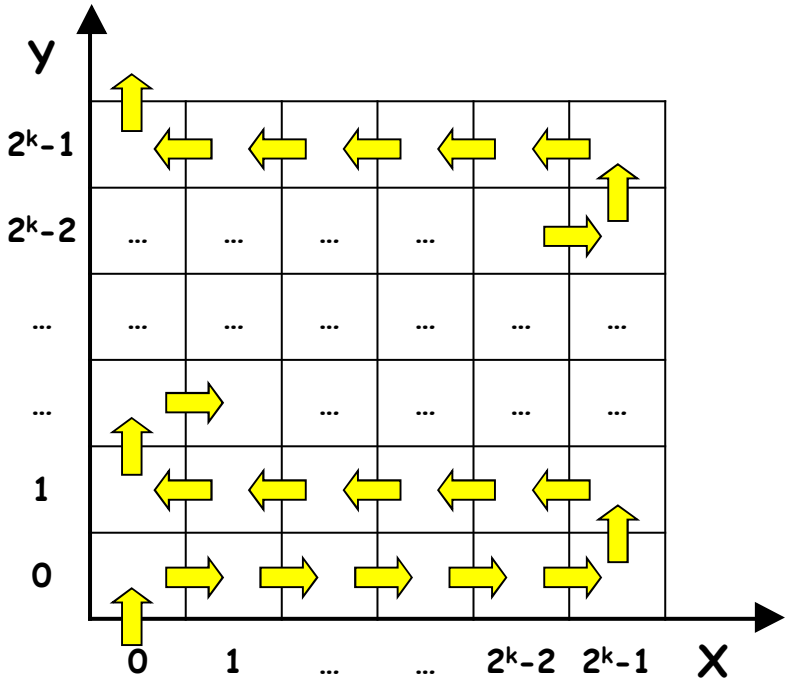
Esercizio 2

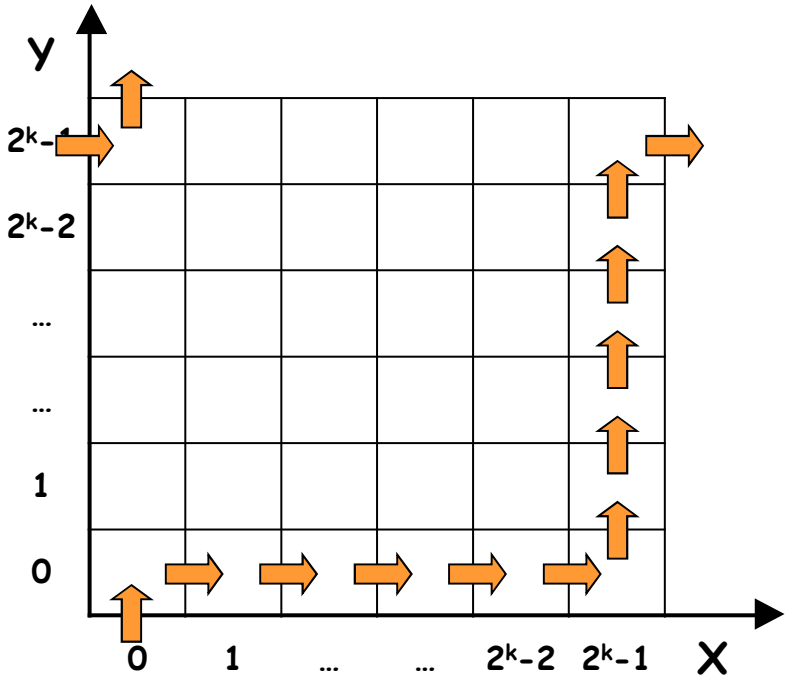
La posizione di un oggetto nel piano X, Y è definita dalle uscite di 2 contatori binari avanti-indietro (C_x, C_y) aventi base di conteggio 2^k: le uscite x_{k-1}, ..., x_1, x_0 di C_x stabiliscono il valore della coordinata X, le uscite y_{k-1}, ..., y_1, y_0 di C_y il valore della coordinata Y.

Una rete sequenziale sincrona operante in base allo stesso clock dei contatori ha il compito di gestire i segnali E_x, E_y (secondo il modello di Mealy) e U_x/D_x', U_y/D_y' (secondo il modello di Moore) dei contatori, in modo tale che l'oggetto descriva nel piano X, Y le due traiettorie indicate in figura, una di seguito all'altra e senza soluzione di continuità.

Si determini l'automa minimo della rete ed una sua realizzazione con FF-T e gate elementari, nell'ipotesi che essa disponga in ingresso unicamente dei segnali:

$$Z_x = x_{k-1} \dots x_0\, U_x/D_x' + x_{k-1}' \dots x_0'\, (U_x/D_x')', \quad Z_y = y_{k-1} \dots y_0\, U_y/D_y' + y_{k-1}' \dots y_0'\, (U_y/D_y')'.$$





In effetti occorre progettare una rete a due sole uscite (E_x, U_x/D_x'):
si può infatti imporre $U_y/D_y' = 1$, dovendo il contatore C_y contare soltanto in avanti, ed assumere $E_y = E_x'$, non essendo mai C_x e C_y contemporaneamente abilitati o disabilitati.

Evoluzione temporale dei segnali di ingresso e di uscita (k=2)

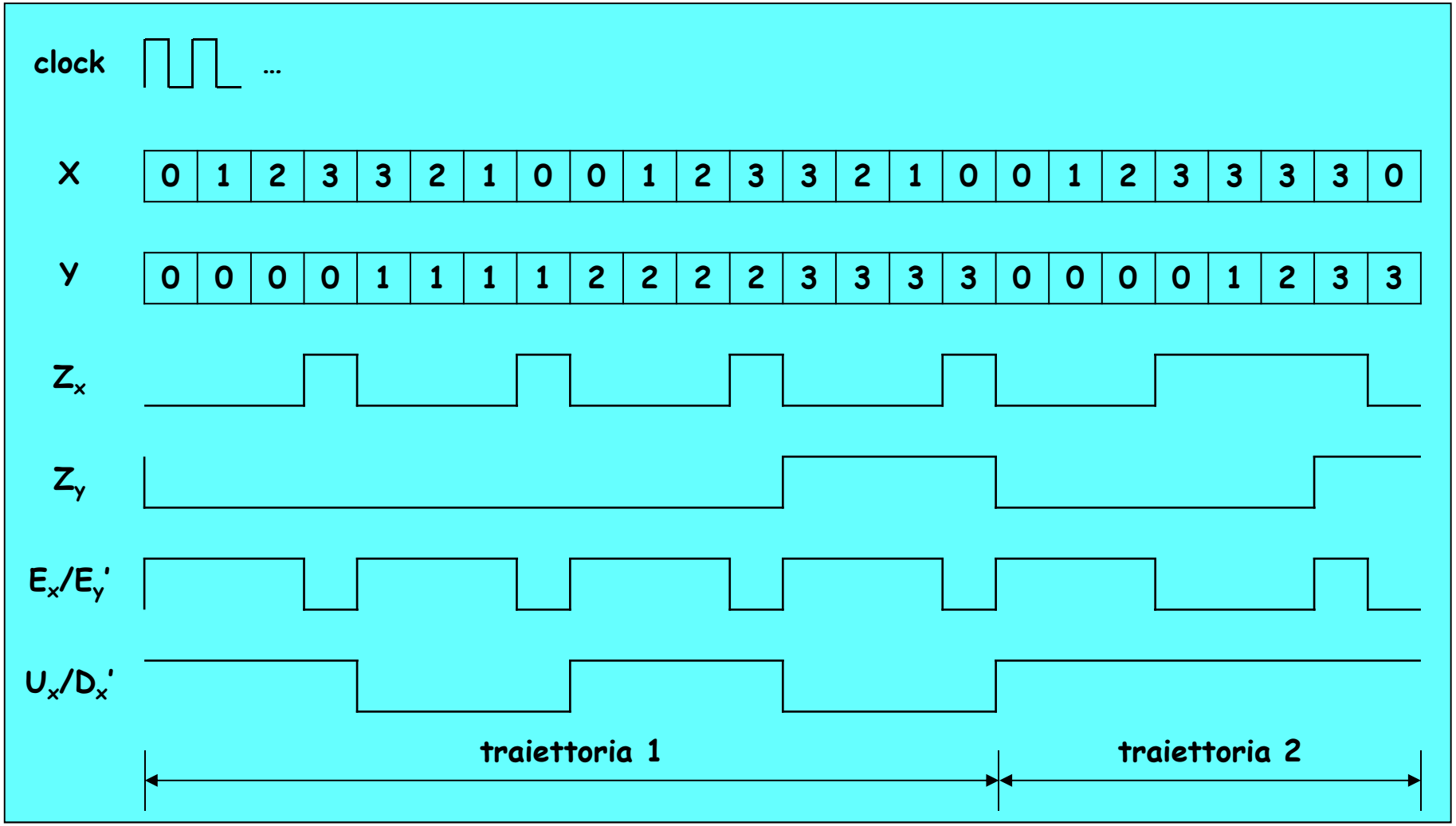
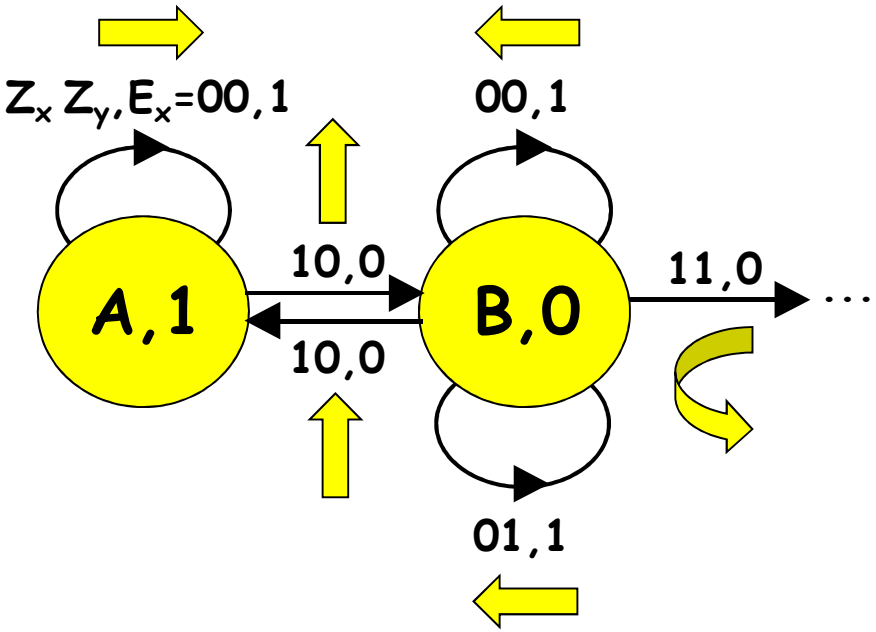


Diagramma degli stati

Il diagramma può essere costruito ipotizzando inizialmente che l'oggetto si trovi nell'origine del piano X, Y e che debba descrivere la prima traiettoria.

A tal fine occorre eseguire nell'ordine i seguenti passi:

- abilitare il contatore C_x in avanti fino a che $X=X_{max}$;
- abilitare il contatore C_y per un intervallo di clock cosicché $Y=Y+1$;
- abilitare il contatore C_x all'indietro fino a che $X=X_{min}$;
- incrementare il contatore C_y e ripetere i passi precedenti se $Y<Y_{max}$.



Traiettoria: 1

Affinché l'oggetto descriva nel piano X, Y la seconda traiettoria, occorre selezionare la modalità di conteggio in avanti anche per il contatore C_x ed eseguire ordinatamente i seguenti passi:

- abilitare il contatore C_x fino a che $X=X_{max}$;
- abilitare il contatore C_y fino a che $Y=Y_{max}$;
- incrementare il contatore C_x cosicché $X=X_{min}$;
- incrementare il contatore C_y cosicché $Y=Y_{min}$ (ritorno nell'origine).

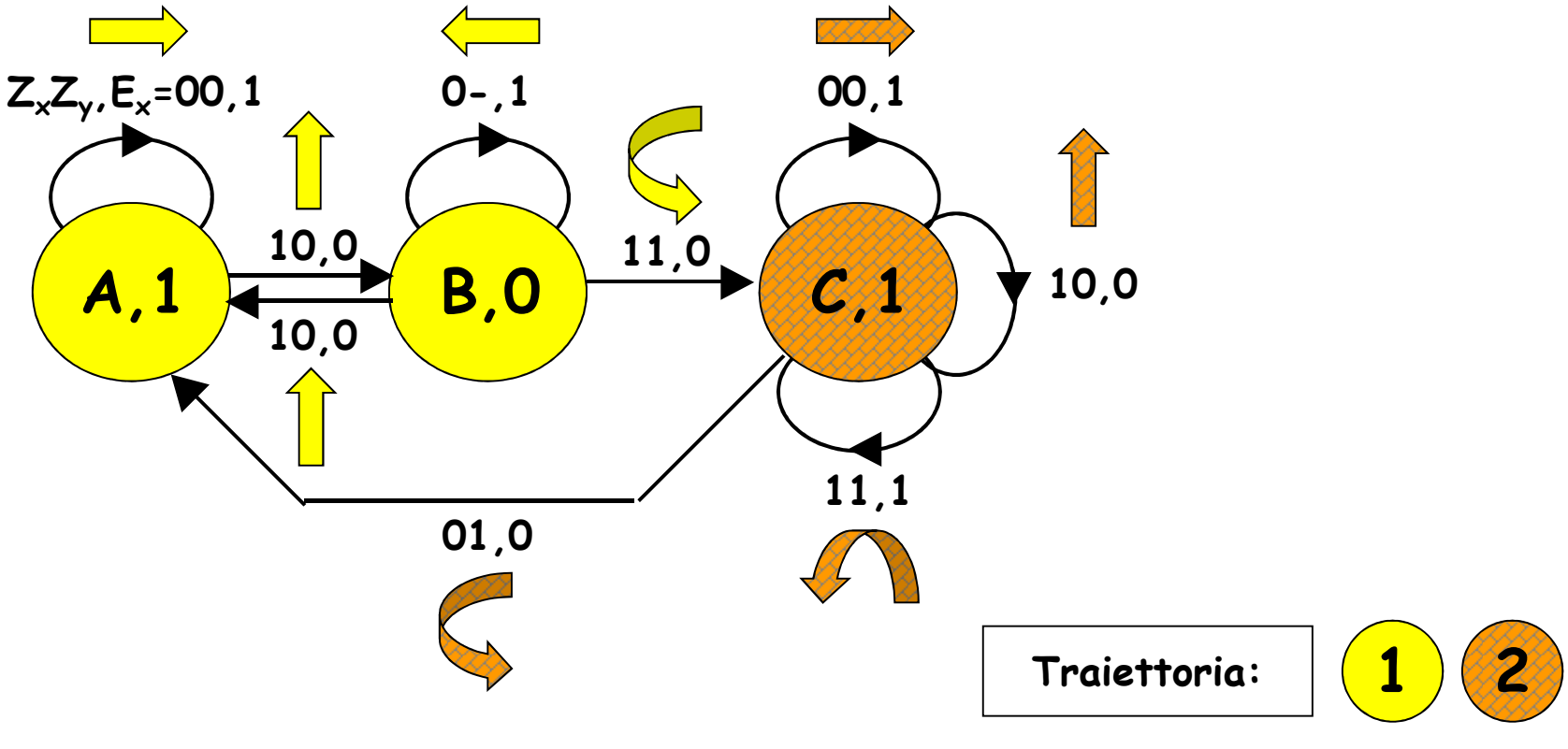


Tabella di flusso e tabella delle transizioni

Z_xZ_y

s.p.	00	01	11	10	U_x/D_x'
A	A,1	-,-	-,-	B,0	1
B	B,1	B,1	C,0	A,0	0
C	C,1	A,0	C,1	C,0	1

$s.f., E_x$

$(Z_xZ_y)^n$

$(y_1y_2)^n$	00	01	11	10	$(U_x/D_x')^n$
(A) 00	00,1	--,-	--,-	01,0	1
(B) 01	01,1	01,1	10,0	00,0	0
11	--,-	--,-	--,-	--,-	-
(C) 10	10,1	00,0	10,1	10,0	1

$(y_1y_2)^{n+1}, E_x^n$

Reti combinatorie di aggiornamento dello stato e di uscita

$T_1^n=(Z_x' Z_y y_1 + Z_x Z_y y_2)^n$

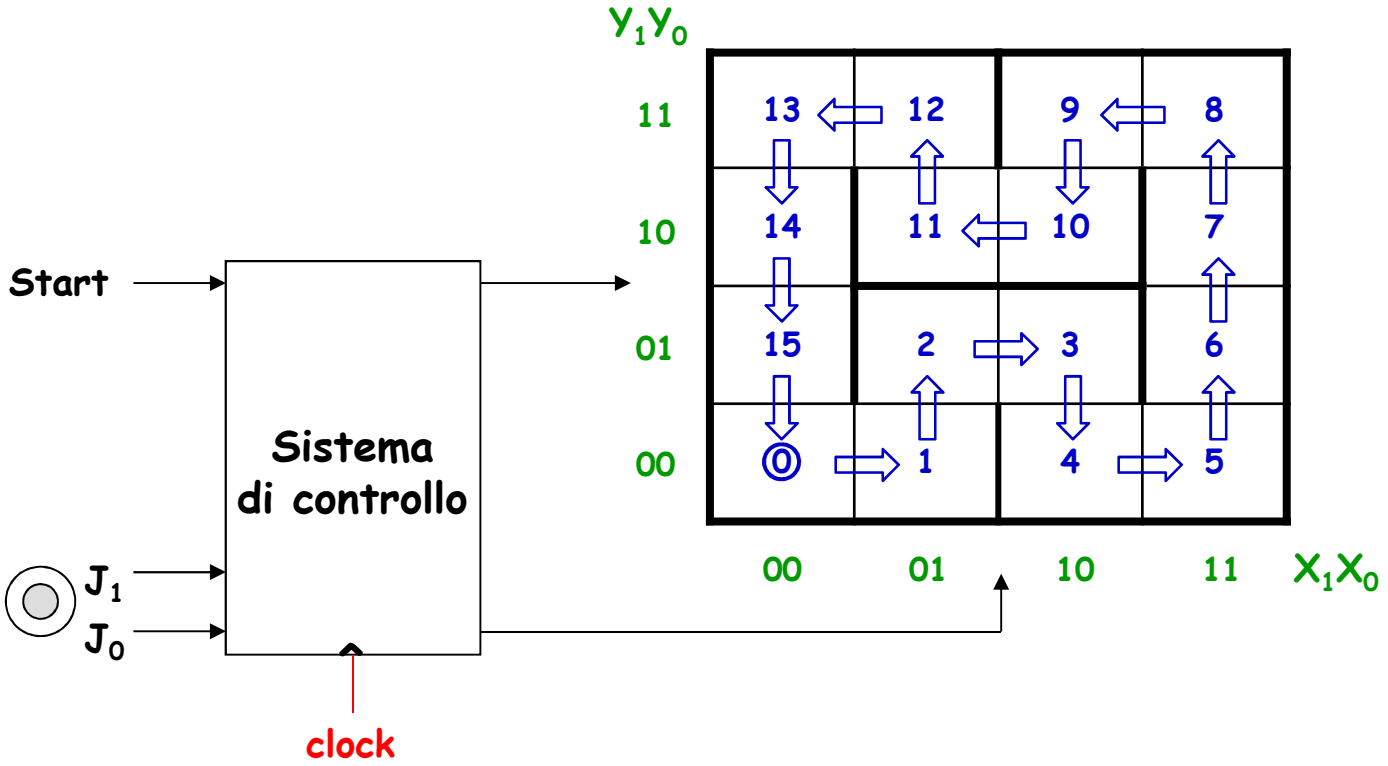
$T_2^n=(Z_x y_1')^n$

$E_x^n=(Z_x' Z_y' + Z_x' y_1' + Z_x Z_y y_1)^n$

$(U_x/D_x')^n=(y_2')^n$

Esercizio 3

Un videogioco prevede che un oggetto mobile su una griglia di sedici celle segua ciclicamente il percorso schematizzato in figura, senza arretrare o andare a sbattere contro le pareti. A gioco fermo l'oggetto permane nella cella 0 di coordinate $Y_1Y_0 = X_1X_0 = 00$. Il gioco ha inizio nell'intervallo di clock successivo a quello in cui il segnale sincrono Start assume il valore 1 (per ipotesi tale valore ha durata unitaria e si può presentare soltanto a gioco fermo in conseguenza dell'introduzione di un gettone). Corrispondentemente il sistema di controllo del videogioco prende atto dei comandi di spostamento dell'oggetto che il giocatore seleziona tramite un joystick. I comandi possibili sono 4, codificati tramite 2 bit J_1, J_0 come segue: 01 = spostamento a destra, 10 = spostamento a sinistra, 11 = spostamento in alto, 00 = spostamento in basso. Se il comando è corretto, l'oggetto è spostato nella cella successiva del percorso; se è errato, l'oggetto è riportato nella cella 0. Il gioco prosegue da tale cella nel caso che gli errori non siano stati più di due; in caso contrario il gioco è considerato concluso.



Il sistema di controllo del videogioco deve essere strutturato, come indicato in figura, in quattro unità:

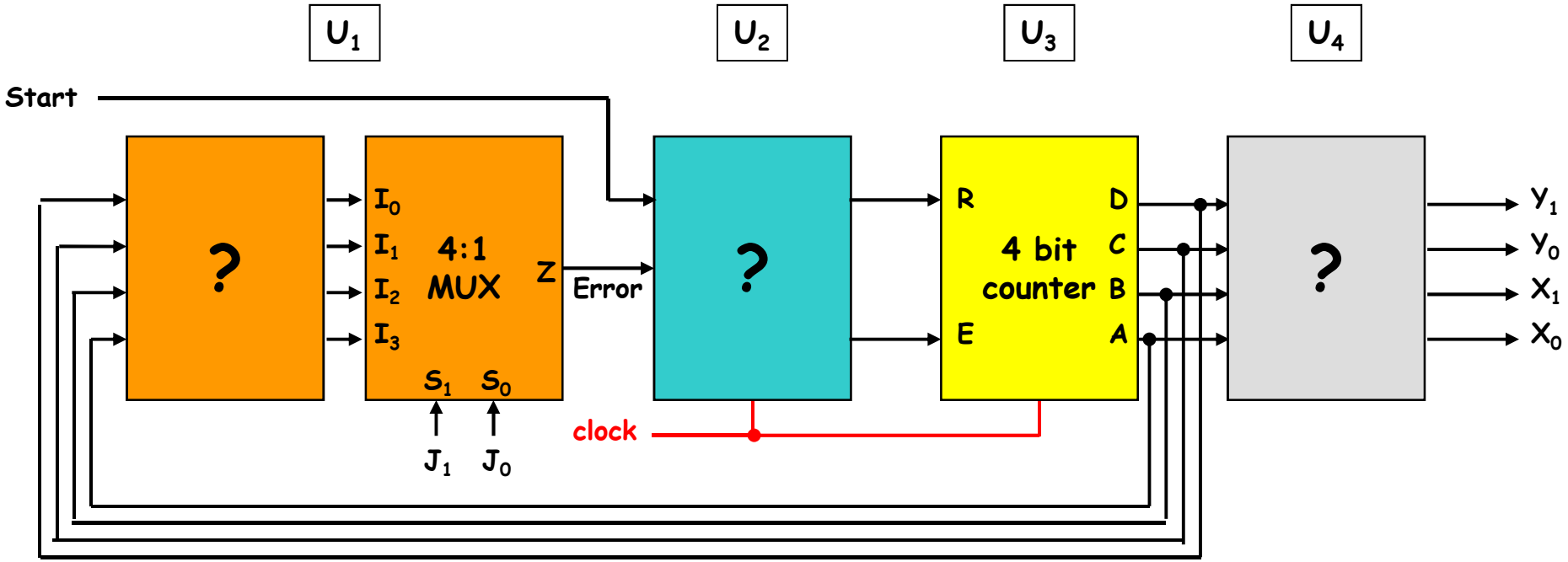
U_1 - unità di rilevazione di errore (spostamento selezionato non corretto);

U_2 - unità di conteggio degli errori;

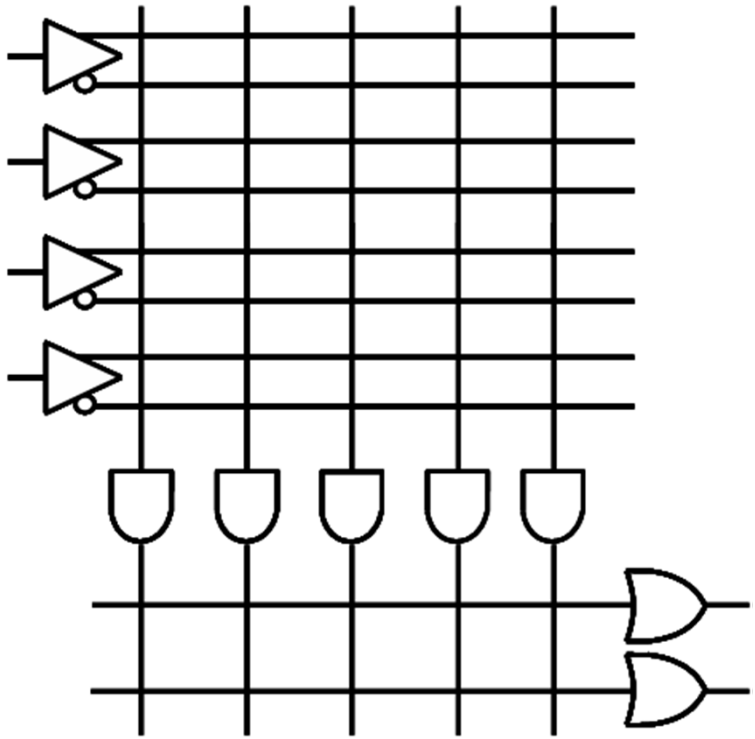
U_3 - unità di conteggio dei passi (numero di spostamenti correttamente eseguiti a partire dalla cella 0);

U_4 - unità di transcodifica delle coordinate dell'oggetto in base al numero di passi eseguiti.

L'unità combinatoria a due stadi U_1 ha il compito di evidenziare, tramite il segnale di uscita Error, la selezione da parte del giocatore di uno spostamento corretto (Error = 0) o errato (Error = 1), tenendo conto del passo corrente (indicato dai segnali di uscita D (MSB), C, B, A (LSB) di U_3) e del valore attuale dei segnali J_1, J_0. L'unità sequenziale U_2 ha il compito di elaborare i segnali Start e Error in modo da indicare all'unità U_3, tramite i comandi RESET (R) e ENABLE (E), se l'oggetto deve (a) restare nella cella 0 in attesa dell'evento Start = 1, (b) ritornare nella cella 0 in caso di spostamento selezionato errato, (c) avanzare di un passo lungo il percorso in caso di spostamento selezionato corretto. L'unità combinatoria U_4 ha il compito di generare le coordinate dell'oggetto in base al valore corrente dei segnali di uscita D, C, B, A di U_3.

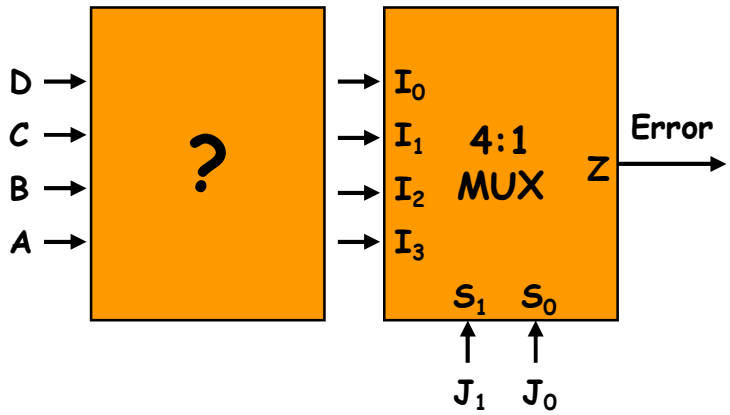


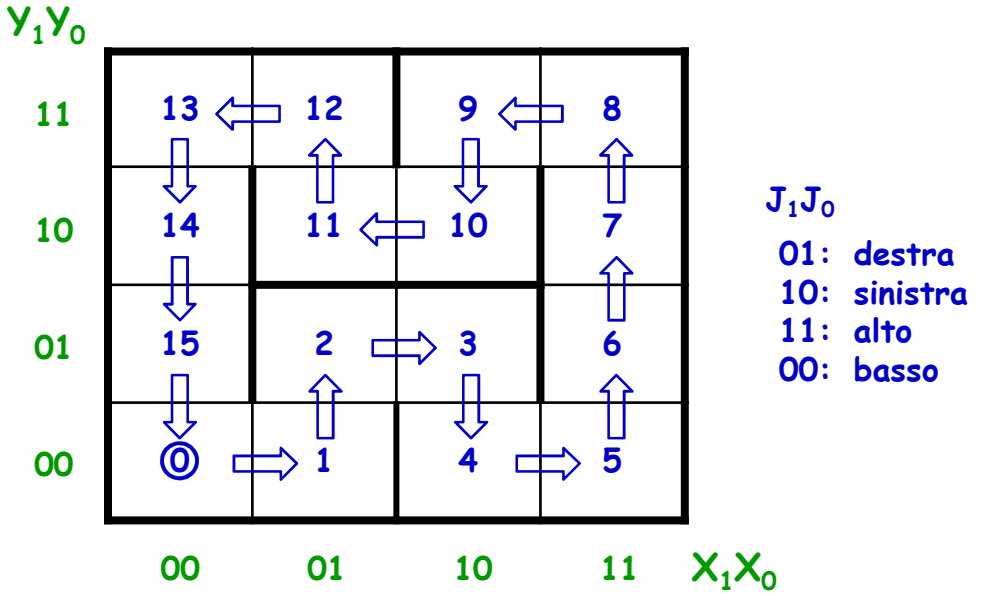
- Si completi il progetto dell'unità U_1.
- Si identifichi il grafo degli stati dell'unità U_2 ed una sua possibile realizzazione basata sull'impiego di un contatore binario dotato di comandi di RESET (R') e ENABLE (E').
- Si evidenzino le modifiche da apportare all'unità U_2 nell'ipotesi che il numero N di errori ammissibili prima della conclusione del gioco sia configurabile dall'esterno mediante tre bit n_2, n_1, n_0 ($0 \leq N \leq 7$).
- Si determini il numero di circuiti logici programmabili del tipo indicato in figura necessari per la sintesi dell'unità U_4.



U_1: unità di rilevazione di errore …

Sintesi del 1° stadio
tramite gate elementari





J_1J_0
01: destra
10: sinistra
11: alto
00: basso

BA

DC \ BA	00	01	11	10
00	0	1	1	0
01	0	1	1	1
11	1	1	1	1
10	1	1	1	1

J_1J_0 = 01 (destra)

OK in 0,2,4

I_1=D+A+CB

BA

DC \ BA	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	1	1	1
10	0	1	1	0

J_1J_0 = 10 (sinistra)

OK in 8,10,12

I_2=D'+A+CB

BA

DC \ BA	00	01	11	10
00	1	0	1	1
01	1	0	0	0
11	1	1	1	1
10	1	1	0	1

J_1J_0 = 11 (alto)

OK in 1,5,6,7,11

I_3=DC+DB'+DA'+B'A'+D'C'B

BA

DC \ BA	00	01	11	10
00	1	1	0	1
01	1	1	1	1
11	1	0	0	0
10	1	0	1	1

J_1J_0 = 00 (basso)

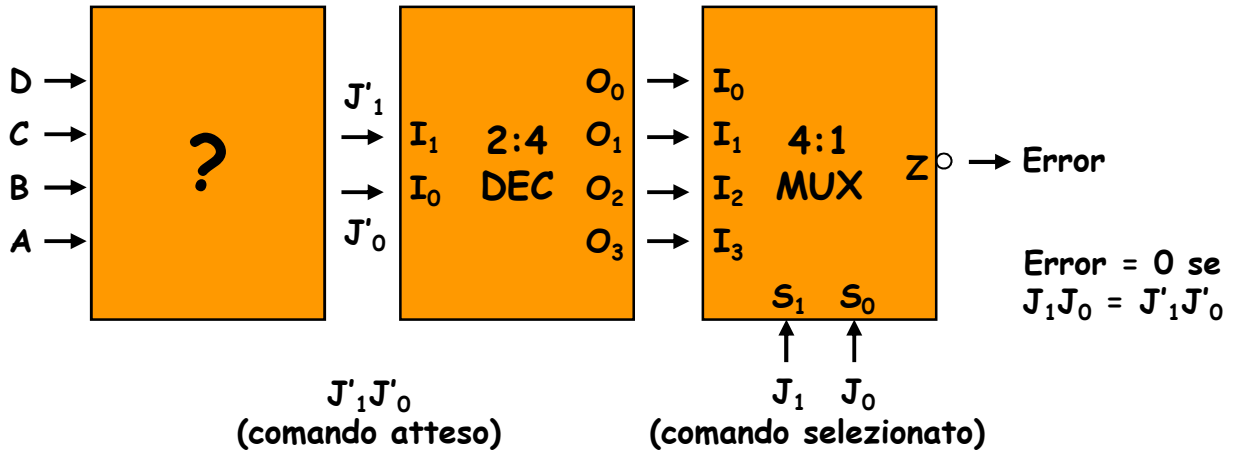
OK in 3,9,13,14,15

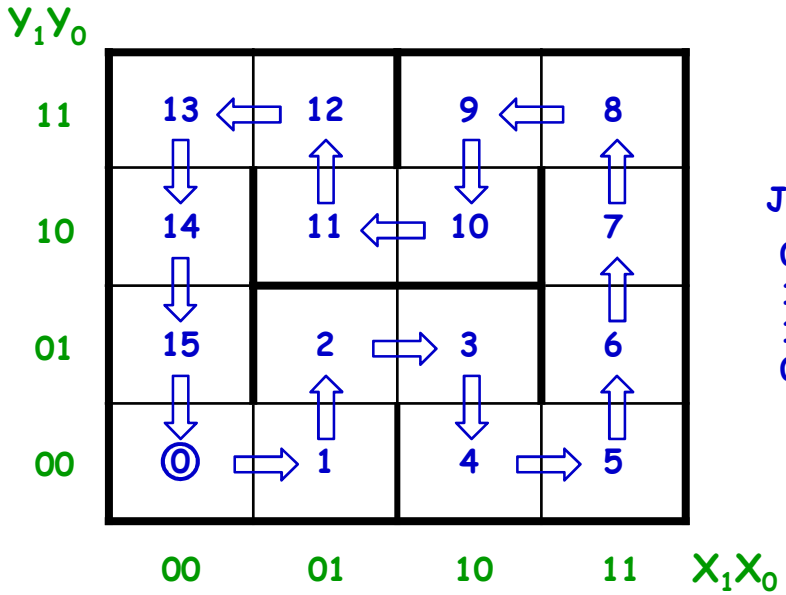
I_0=D'C+D'B'+D'A'+B'A'+DC'B

… U_1: unità di rilevazione di errore

oppure:

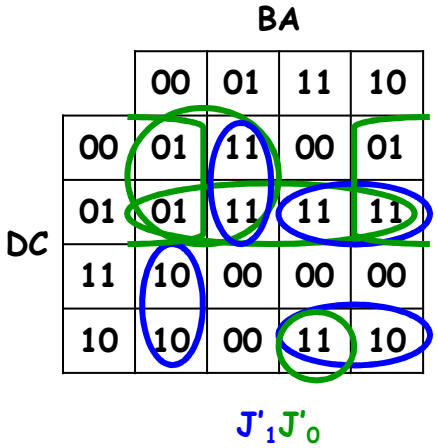
Sintesi del 1° stadio
con comparatore (DEC/MUX)
e (-) gate elementari





J_1J_0

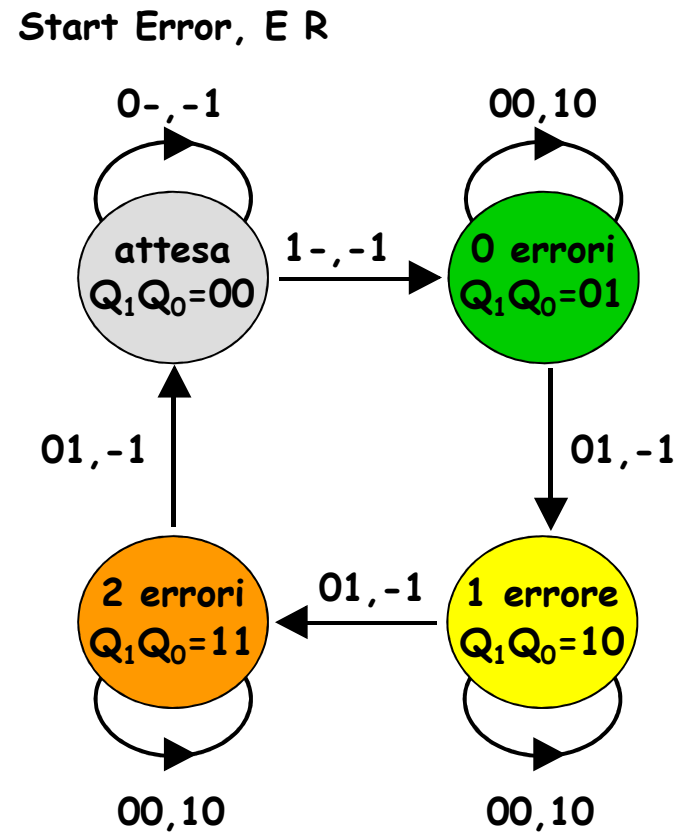
01: destra
10: sinistra
11: alto
00: basso



$J'_1 = DB'A' + D'B'A + D'CB + DC'B$

$J'_0 = D'B' + D'C + D'A' + DC'BA$

U_2: unità di conteggio degli errori



Estensione nel caso che siano ammissibili N errori ($0 \le N \le N_{max}$):

counter $\times(N_{max}+2)$

E, R, E': idem

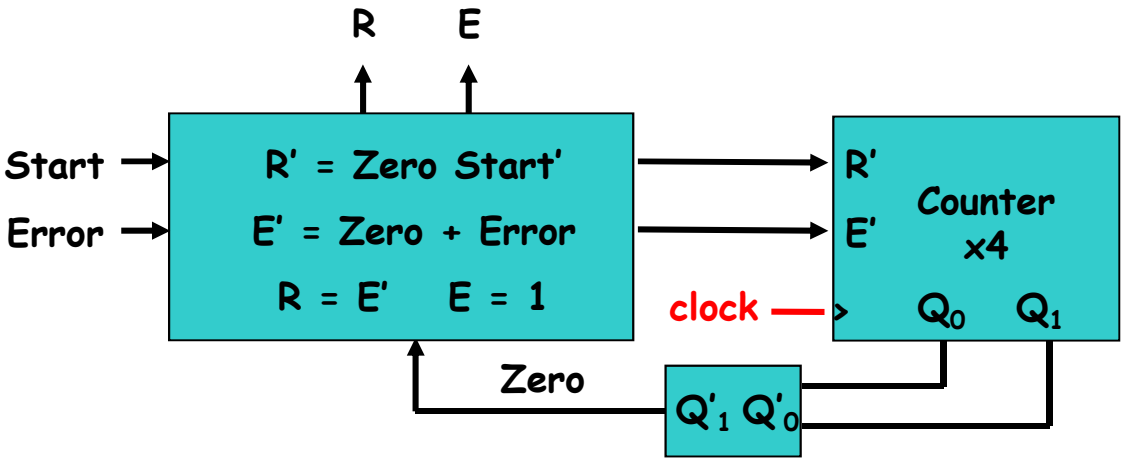
R' = idem + Error (Count = (N+1))

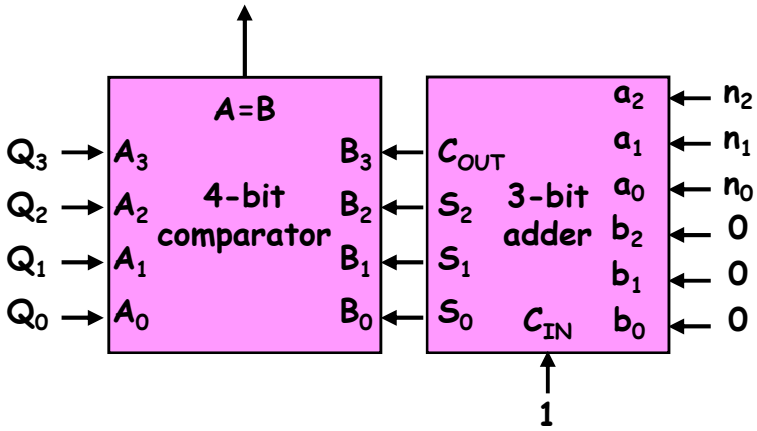
Se $0 \le N \equiv (n_2 n_1 n_0) \le 7$:

counter $\times 9$

Zero = $Q'_3\, Q'_2\, Q'_1\, Q'_0$

R' = Zero Start' + Error $(Q_3Q_2Q_1Q_0 = (n_2n_1n_0+1))$





U_4: unità di generazione delle coordinate dell'oggetto

DC \ BA	00	01	11	10
00	0	0	1	0
01	1	1	1	1
11	0	0	0	0
10	1	1	0	1

$X_1 = D'C + DC'B' + DC'(B)A' + D'C'BA$

$K = D'C'BA + DCBA$

DC \ BA	00	01	11	10
00	0	1	0	1
01	0	1	1	1
11	1	0	0	0
10	1	0	1	0

$X_0 = DB'A' + D'B'A + D'BA' + K'BA$

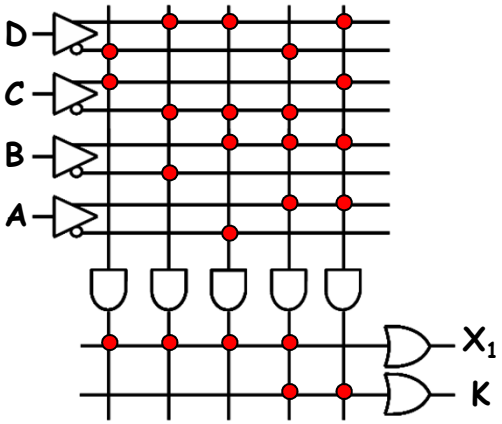
$W = D'BA' + K$

DC \ BA	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	1	1	0	1
10	1	1	1	1

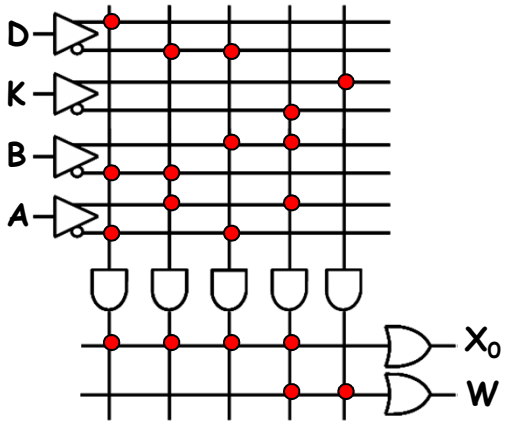
$Y_1 = DB' + W'B$

DC \ BA	00	01	11	10
00	0	0	1	1
01	0	0	0	1
11	1	1	1	0
10	1	1	0	0

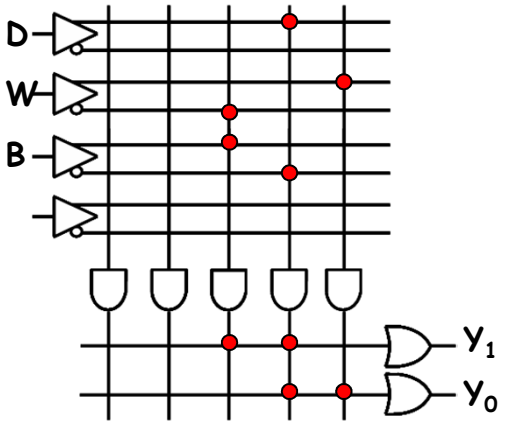
$Y_0 = DB' + W$



1°



2°



3°

Esercizio 4

In un sistema di comunicazione seriale vengono trasferite una di seguito all'altra parole di 4 bit (I_1, I_0, C_1, C_0) tramite un segnale binario sincrono X. In ogni parola i primi 2 bit I_1, I_0 (bit di informazione) possono assumere una configurazione qualsiasi, mentre gli ultimi 2 bit C_1, C_0 (bit di rilevazione di errore) sono calcolati dal trasmettitore in maniera tale da indicare, in binario, il numero di 1 presenti nell'ambito dei bit di informazione.

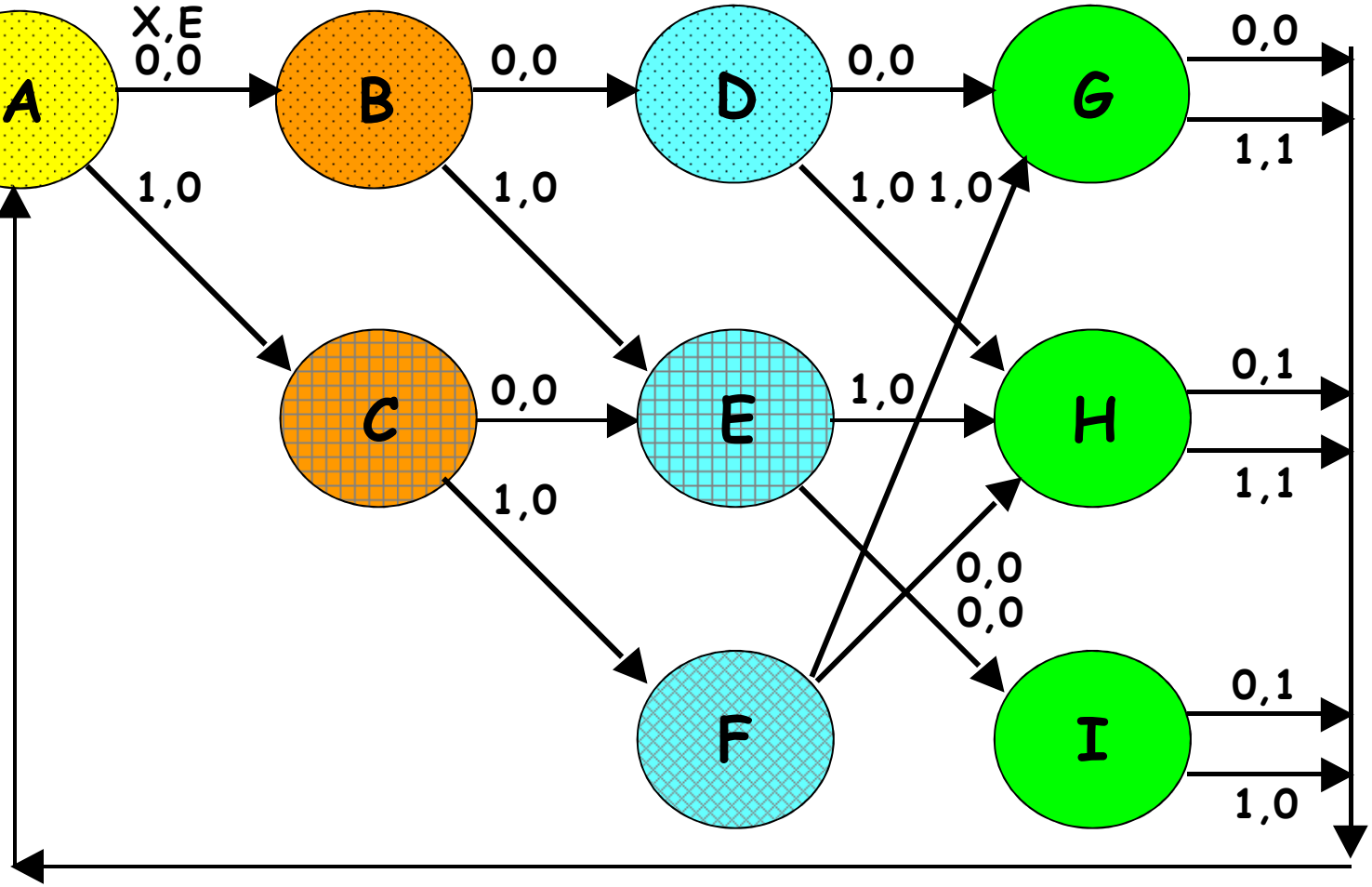
Si progetti una rete sequenziale sincrona ad un ingresso (X) ed una uscita (E), in grado di attivare il segnale E in corrispondenza dell'intervallo di ricezione dell'ultimo bit di ogni parola non correttamente ricevuta; in ogni altro intervallo di clock il segnale E deve assumere il valore 0.

Si identifichi:

- l'automa minimo della rete secondo il modello di Mealy;
- la realizzazione della rete mediante FF-JK e gate elementari.

parola	...	i-1 esima				i esima				i+1 esima				...
bit	...	I_1	I_0	C_1	C_0	I_1	I_0	C_1	C_0	I_1	I_0	C_1	C_0	...
X	...	0	1	1	0	1	0	0	1	1	1	0	1	...
E	...	0	0	0	1	0	0	0	0	0	0	0	1	...

Diagramma degli stati (modello di Mealy)



Numero di bit già ricevuti nell'ambito di una parola:	0	1	2	3
Numero di bit di informazione uguali a 1 già rilevati nell'ambito di una parola:	0	1	2	

Tabella di flusso

	X=0	X=1
A	B,0	C,0
B	D,0	E,0
C	E,0	F,0
D	G,0	H,0
E	I,0	H,0
F	H,0	G,0
G	A,0	A,1
H	A,1	A,1
I	A,1	A,0

non riducibile

Tabella delle transizioni

$(Xy_1y_2)^n$ \ $(y_3y_4)^n$

	00	01	11	10
000	0001,0	0010,0	0000,0	0011,0
001	----,-	0110,0	0000,1	1011,0
011	----,-	----,-	----,-	----,-
010	----,-	----,-	0000,1	0111,0
100	0101,0	0110,0	0000,1	0111,0
101	----,-	1010,0	0000,1	0111,0
111	----,-	----,-	----,-	----,-
110	----,-	----,-	0000,0	0011,0

$(y_1y_2y_3y_4)^{n+1}, Z^n$

Mappa di codifica

y_1y_2 \ y_3y_4

	00	01	11	10	
00	A	B	G	D	0
01	-	C	H	E	1
11	-	-	-	-	
10	-	-	I	F	2
	0	1	3	2	

Rete combinatoria di uscita

$$Z^n=(y_2y_3y_4+Xy_1'y_3y_4+X'y_1y_4)^n$$

Rete combinatoria di aggiornamento dello stato

$(Xy_1y_2)^n$ \ $(y_3y_4)^n$

	00	01	11	10
000	0	0	0	0
001	-	0	0	1
011	-	-	-	-
010	-	-	0	0
100	0	0	0	0
101	-	1	0	0
111	-	-	-	-
110	-	-	0	0

y_1^{n+1}

$(Xy_1y_2)^n$ \ $(y_3y_4)^n$

	00	01	11	10
000	0	0	0	0
001	-	1	0	0
011	-	-	-	-
010	-	-	0	1
100	1	1	0	1
101	-	0	0	1
111	-	-	-	-
110	-	-	0	0

y_2^{n+1}

$(Xy_1y_2)^n$ \ $(y_3y_4)^n$

	00	01	11	10
000	0	1	0	1
001	-	1	0	1
011	-	-	-	-
010	-	-	0	1
100	0	1	0	1
101	-	1	0	1
111	-	-	-	-
110	-	-	0	1

y_3^{n+1}

$(Xy_1y_2)^n$ \ $(y_3y_4)^n$

	00	01	11	10
000	1	0	0	1
001	-	0	0	1
011	-	-	-	-
010	-	-	0	1
100	1	0	0	1
101	-	0	0	1
111	-	-	-	-
110	-	-	0	1

y_4^{n+1}

$J_1^n=(X'y_2y_4'+Xy_2y_3')^n$

$K_1^n=1$

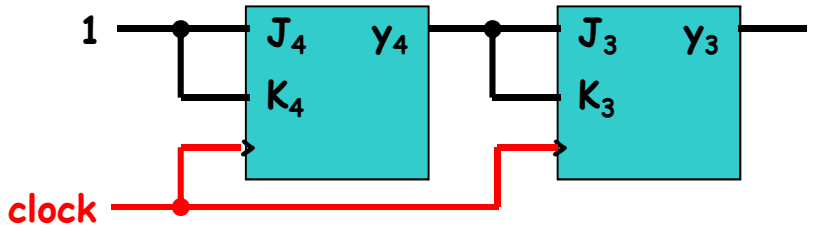
$J_2^n=(X'y_1y_4'+Xy_3'+Xy_1'y_4')^n$

$K_2^n=(X'y_3+Xy_4)^n$

$J_3^n=K_3^n=y_4^n$

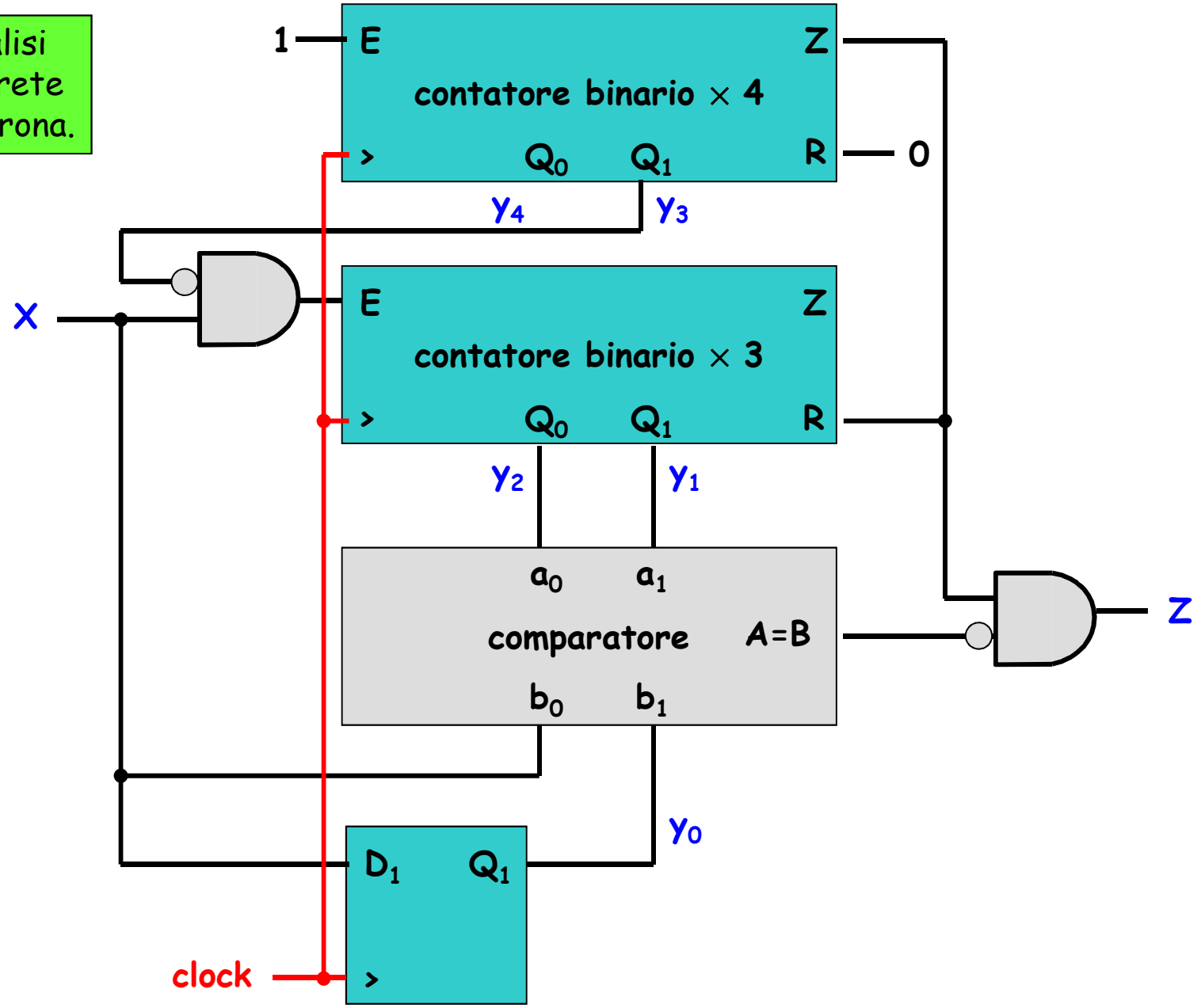
$J_4^n=K_4^n=1$

La codifica degli stati selezionata conduce ad una rete comprendente un contatore binario × 4, cui è delegato il compito di partizionare in parole i bit presentati in ingresso.



Esercizio 5

Si esegua l'analisi della seguente rete sequenziale sincrona.



Espressioni delle variabili di stato interno futuro e di uscita in funzione delle variabili di stato interno presente e di ingresso:

$y_0^{n+1} = X^n$

$y_1^{n+1} = ((y_1 \oplus (X\, y_3'\, y_2))\ (y_1\,(y_2 + X\, y_3') + y_3\, y_4)')^n$

$y_2^{n+1} = ((y_2 \oplus (X\, y_3'))\ (y_1\,(y_2 + X\, y_3') + y_3\, y_4)')^n$

$y_3^{n+1} = (y_3 \oplus y_4)^n$

$y_4^{n+1} = (y_4')^n$

$Z^n = (y_3\, y_4\, ((y_1 \oplus y_0) + (y_2 \oplus X))^n$

Tabella delle transizioni

$(y_0y_1y_2)^n$ \ $(Xy_3y_4)^n$	000	001	011	010	100	101	111	110
000	00001,0	00010,0	00000,0	00011,0	10101,0	10110,0	10000,1	10011,0
001	00101,0	00110,0	00000,1	00111,0	11001,0	11010,0	10000,0	10111,0
011	00001,0	00010,0	00000,1	00011,0	10001,0	10010,0	10000,1	10011,0
010	01001,0	01010,0	00000,1	01011,0	10001,0	10010,0	10000,1	11011,0
100	00001,0	00010,0	00000,1	00011,0	10101,0	10110,0	10000,1	10011,0
101	00101,0	00110,0	00000,1	00111,0	11001,0	11010,0	10000,1	10111,0
111	00001,0	00010,0	00000,1	00011,0	10001,0	10010,0	10000,0	10011,0
110	01001,0	01010,0	00000,0	01011,0	10001,0	10010,0	10000,1	11011,0

$(y_0y_1y_2y_3y_4)^{n+1}, Z^n$

Tabella di flusso

(in una forma, date le dimensioni, non del tutto convenzionale per motivi di disegno)

ciascuno stato è identificato con il valore decimale (0, 1, …, 31)
corrispondente alla relativa rappresentazione binaria (00000, 00001, …, 11111):

	X = 0	X = 1
0	1,0	21,0
1	2,0	22,0
2	3,0	19,0
3	0,0	16,1
4	5,0	25,0
5	6,0	26,0
6	7,0	23,0
7	0,1	16,0

	X = 0	X = 1
8	9,0	17,0
9	10,0	18,0
10	11,0	27,0
11	0,1	16,1
12	1,0	17,0
13	2,0	18,0
14	3,0	19,0
15	0,1	16,1

	X = 0	X = 1
16	1,0	21,0
17	2,0	22,0
18	3,0	19,0
19	0,1	16,1
20	5,0	25,0
21	6,0	26,0
22	7,0	23,0
23	0,1	16,1

	X = 0	X = 1
24	9,0	17,0
25	10,0	18,0
26	11,0	27,0
27	0,0	16,1
28	1,0	17,0
29	2,0	18,0
30	3,0	19,0
31	0,1	16,0

Eventuali stati che contraddistinguono il comportamento della rete in transitorio devono essere rimossi dalla tabella di flusso *prima* di applicare l'algoritmo di riduzione.

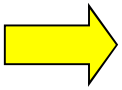
stati irraggiungibili: 4, 8, 12, 13, 14, 15, 20, 24, 28, 29, 30, 31

Tabella di flusso ridotta

	X = 0	X = 1
0	1,0	21,0
1	2,0	22,0
2	3,0	19,0
3	0,0	16,1
5	6,0	26,0
6	7,0	23,0
7	0,1	16,0
9	10,0	18,0
10	11,0	27,0
11	0,1	16,1
16	1,0	21,0
17	2,0	22,0
18	3,0	19,0
19	0,1	16,1
21	6,0	26,0
22	7,0	23,0
23	0,1	16,1
25	10,0	18,0
26	11,0	27,0
27	0,0	16,1

stati ora irraggiungibili:

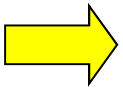
5 9 17 25



	X = 0	X = 1
0	1,0	21,0
1	2,0	22,0
2	3,0	19,0
3	0,0	16,1
6	7,0	23,0
7	0,1	16,0
10	11,0	27,0
11	0,1	16,1
16	1,0	21,0
18	3,0	19,0
19	0,1	16,1
21	6,0	26,0
22	7,0	23,0
23	0,1	16,1
26	11,0	27,0
27	0,0	16,1

stati ora irraggiungibili:

10 18



	X = 0	X = 1
0	1,0	21,0
1	2,0	22,0
2	3,0	19,0
3	0,0	16,1
6	7,0	23,0
7	0,1	16,0
11	0,1	16,1
16	1,0	21,0
19	0,1	16,1
21	6,0	26,0
22	7,0	23,0
23	0,1	16,1
26	11,0	27,0
27	0,0	16,1

Tabella di flusso ridotta

	X = 0	X = 1
0	1,0	21,0
1	2,0	22,0
2	3,0	19,0
3	0,0	16,1
6	7,0	23,0
7	0,1	16,0
11	0,1	16,1
16	1,0	21,0
19	0,1	16,1
21	6,0	26,0
22	7,0	23,0
23	0,1	16,1
26	11,0	27,0
27	0,0	16,1

Tabella triangolare

È possibile ridurre ulteriormente e direttamente la tabella osservando che alcune **righe** sono **identiche**:

{0,16}, {3,27}, {6,22}, {11,19,23}

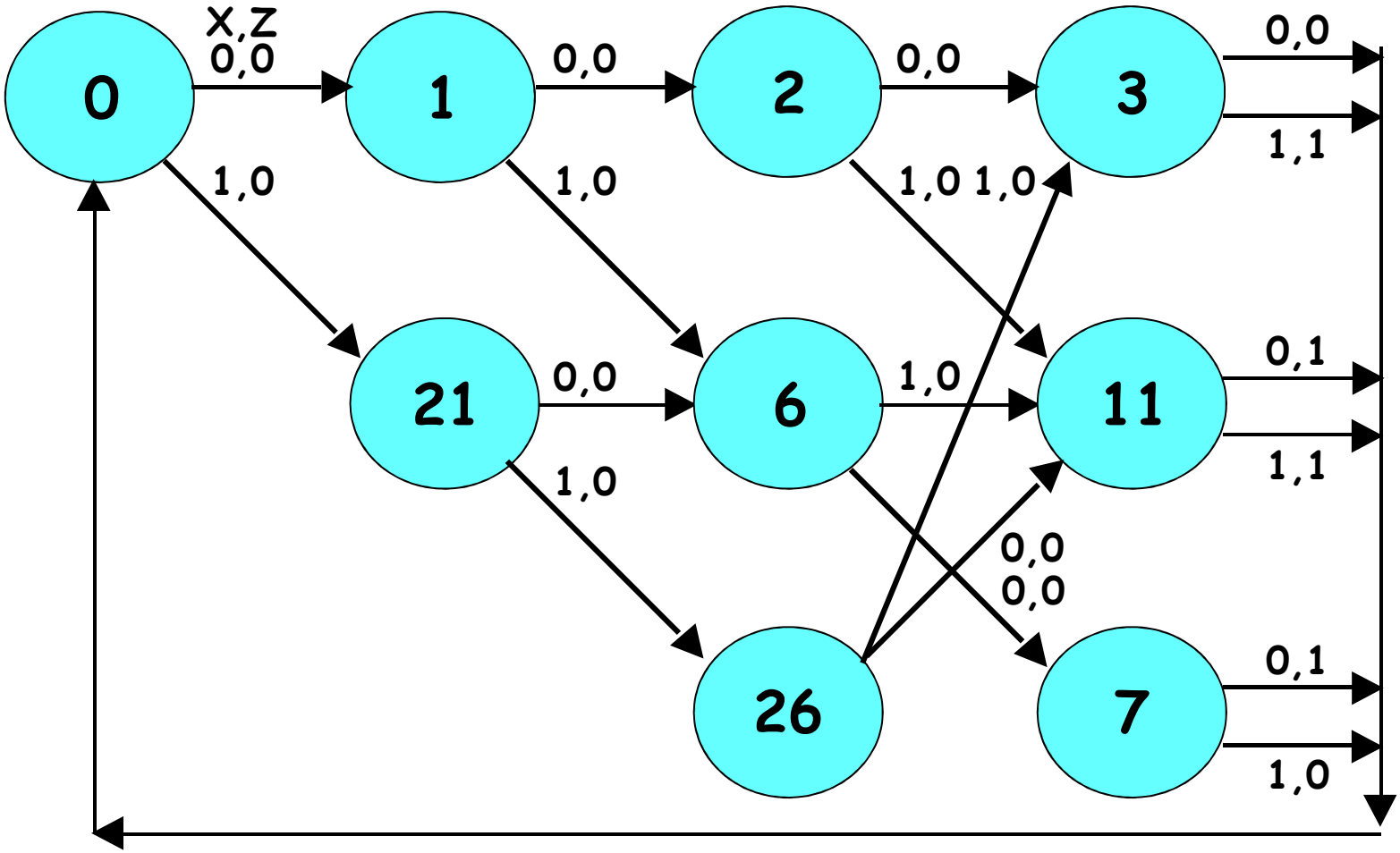
Gli **stati** corrispondenti sono pertanto **indistinguibili**.

	0	1	2	3	6	7	11	21
1	1,2 21,6							
2	1,3 21,11	2,3 6,11						
3								
6	1,7 21,11	2,7 6,11	3,7					
7								
11								
21	1,6 21,26	2,6 6,26	3,6 11,26		7,6 11,26			
26	1,11 21,3	2,11 6,3	3,11		7,11 11,3			6,11 26,3

	X = 0	X = 1
0	1,0	21,0
1	2,0	6,0
2	3,0	11,0
3	0,0	0,1
6	7,0	11,0
7	0,1	0,0
11	0,1	0,1
21	6,0	26,0
26	11,0	3,0

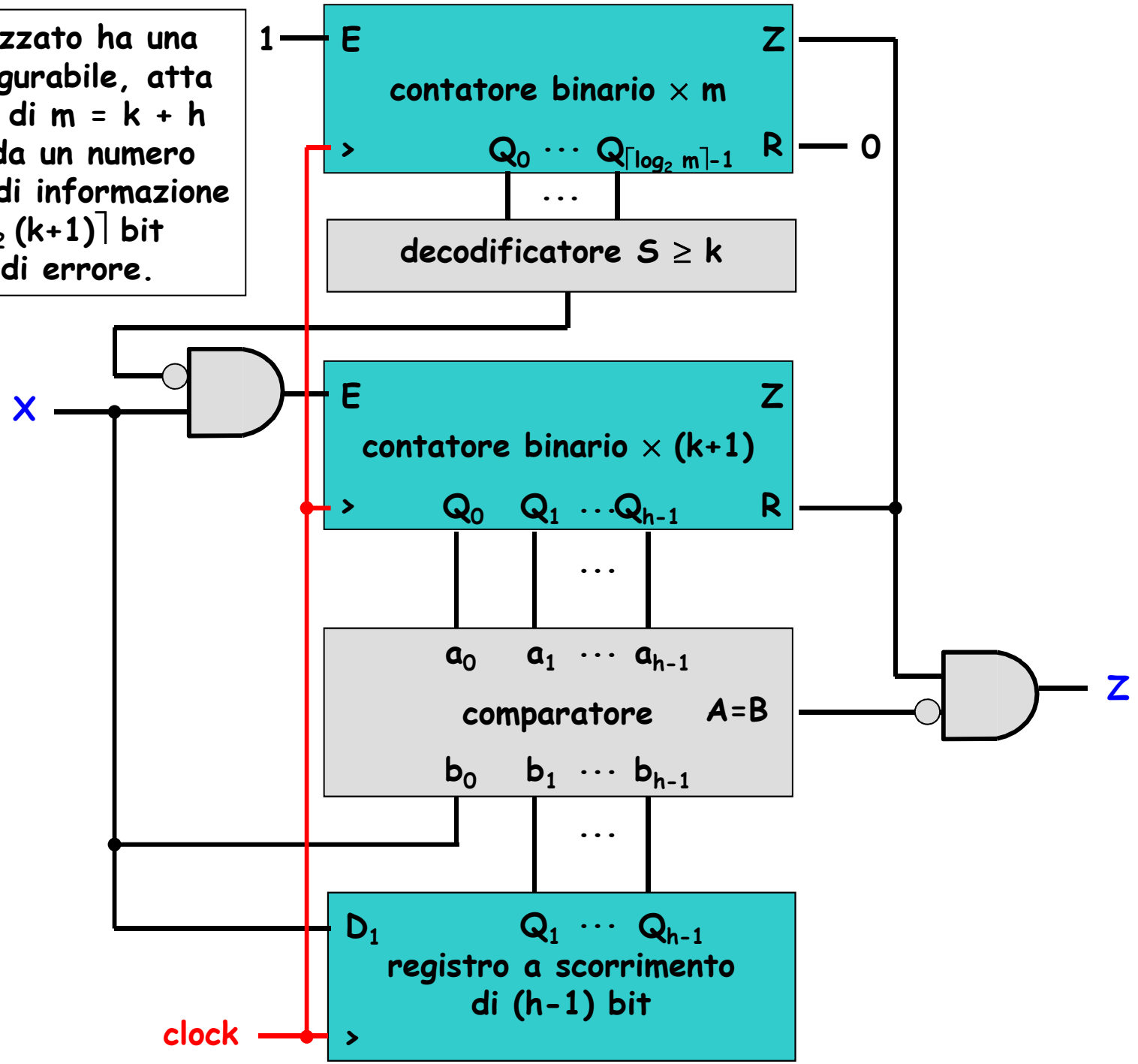
Tabella di flusso non riducibile

Diagramma degli stati



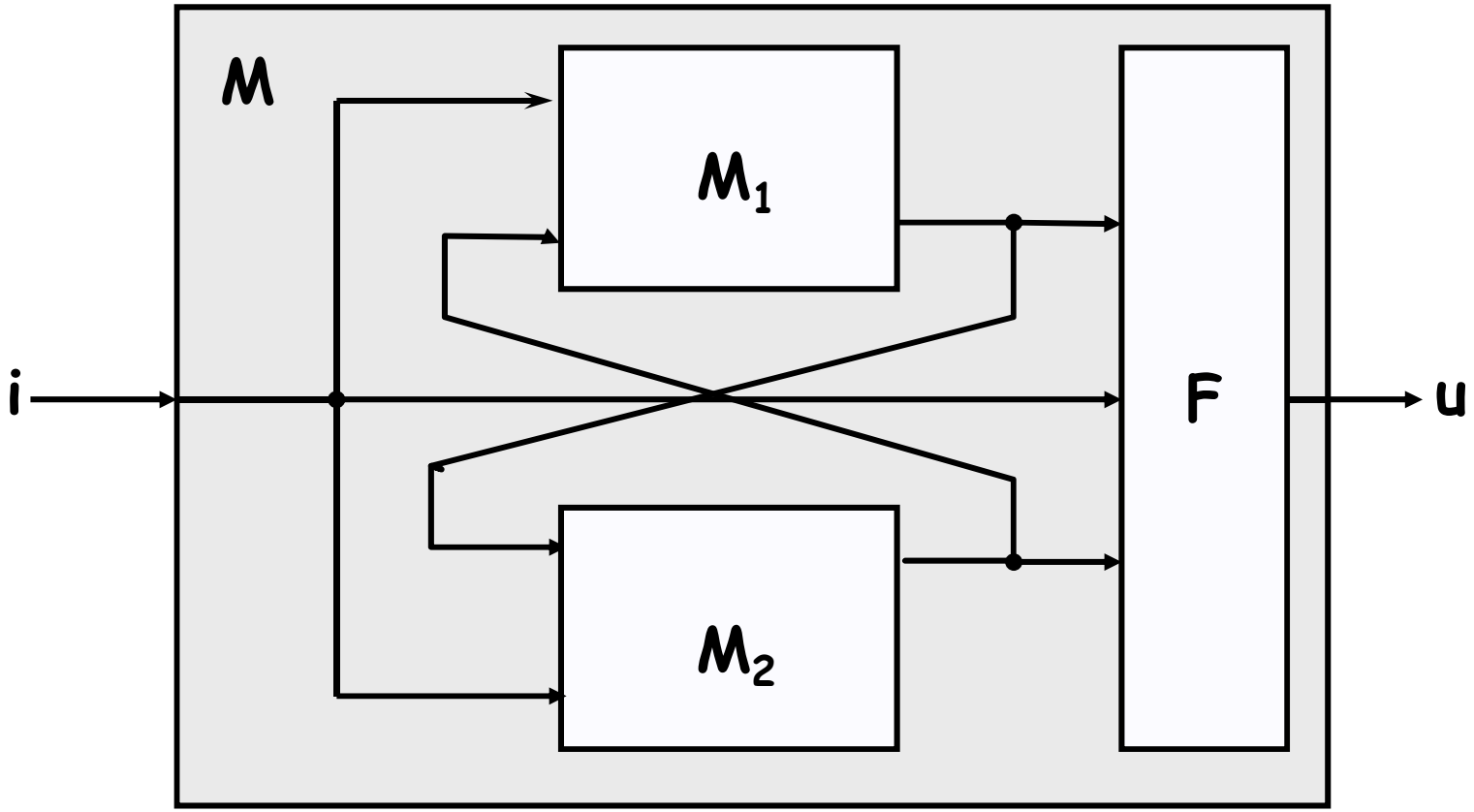
La rete in esame presenta, a regime, lo stesso comportamento di quella progettata nell'esercizio precedente !

Lo schema analizzato ha una struttura riconfigurabile, atta a gestire parole di m = k + h bit, costituite da un numero qualsiasi k di bit di informazione e da h = $\lceil \log_2 (k+1) \rceil$ bit di rilevazione di errore.




Il principio di
decomposizione

Principio di decomposizione:
una macchina sequenziale M con N stati può essere decomposta in due macchine più semplici M_1 e M_2, contraddistinte da un numero di stati N_1 e N_2, rispettivamente, tali che $N_1 < N$, $N_2 < N$, $N_1 \times N_2 \geq N$.



Esercizio 1

Una rete sequenziale sincrona è caratterizzata da due segnali di ingresso (E, X) e da un segnale di uscita (Z), tutti sincroni rispetto al clock della rete stessa.

Attraverso l'ingresso X la rete riceve serialmente parole di k bit. Il segnale E, attivo (valore logico 1) per k intervalli di clock, segnala la fase di ricezione di ciascuna parola.

L'uscita Z della rete può assumere il valore logico 1 soltanto in corrispondenza dell'intervallo di clock immediatamente successivo a quello di ricezione dell'ultimo bit di ciascuna parola, e ciò se la parola comprende almeno tre 1 consecutivi, ma non due 0 consecutivi.

Si identifichi:

- l'automa minimo della rete secondo il modello di Mealy;
- la realizzazione della rete con FF-JK e gate elementari.

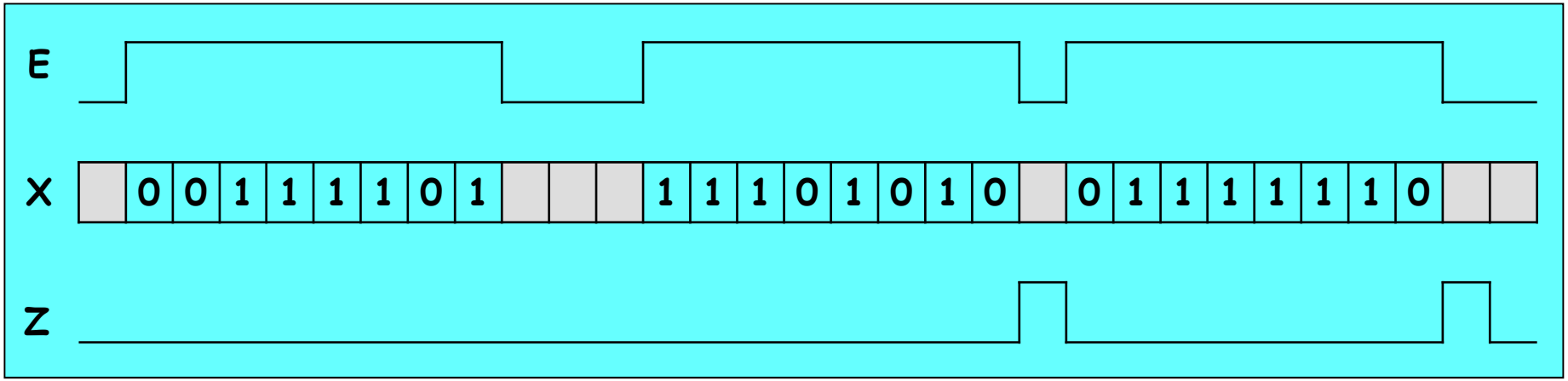
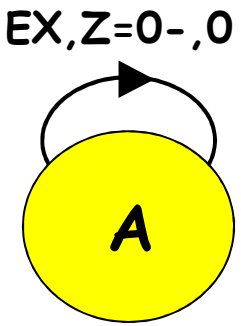
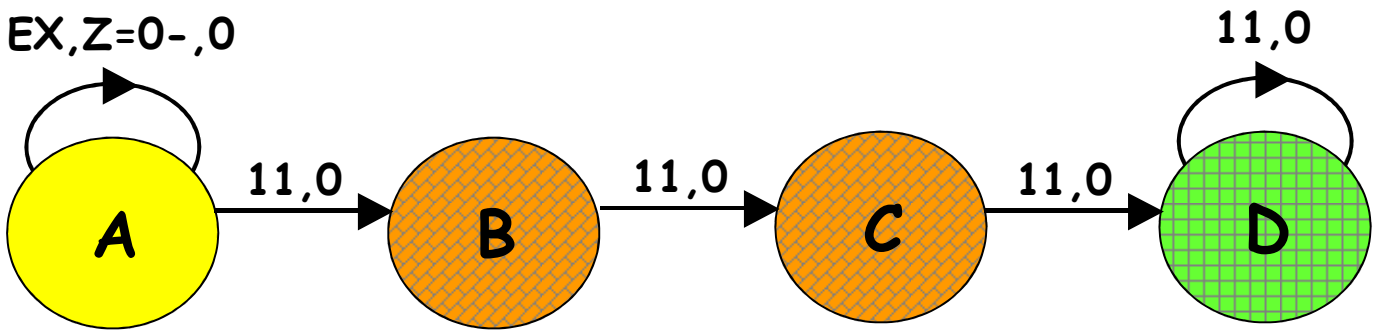


Diagramma degli stati (modello di Mealy)

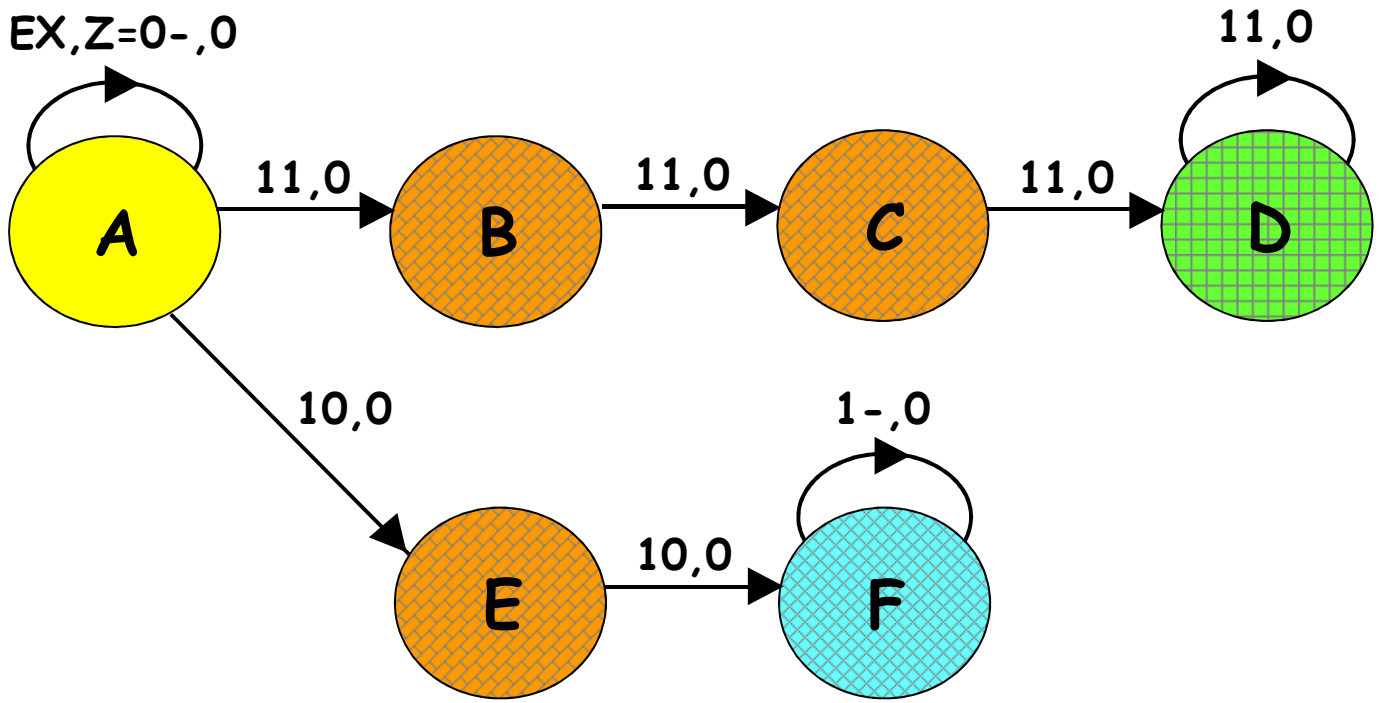
Il diagramma degli stati può essere costruito a partire dalla situazione in cui si trova ad operare la rete allorché in attesa di una parola da elaborare (E=0, X=-, Z=0).



Supponiamo ora che, quando E si attiva, la parola in ingresso presenti subito la sequenza desiderata, ovvero che X assuma il valore 1 per (almeno) tre intervalli di clock consecutivi.



Supponiamo ora che, quando E si attiva, la parola in ingresso presenti subito la sequenza indesiderata, ovvero che X assuma il valore 0 per due intervalli di clock consecutivi. In tal caso i successivi valori di X nell'ambito di una parola risultano ininfluenti.



attesa

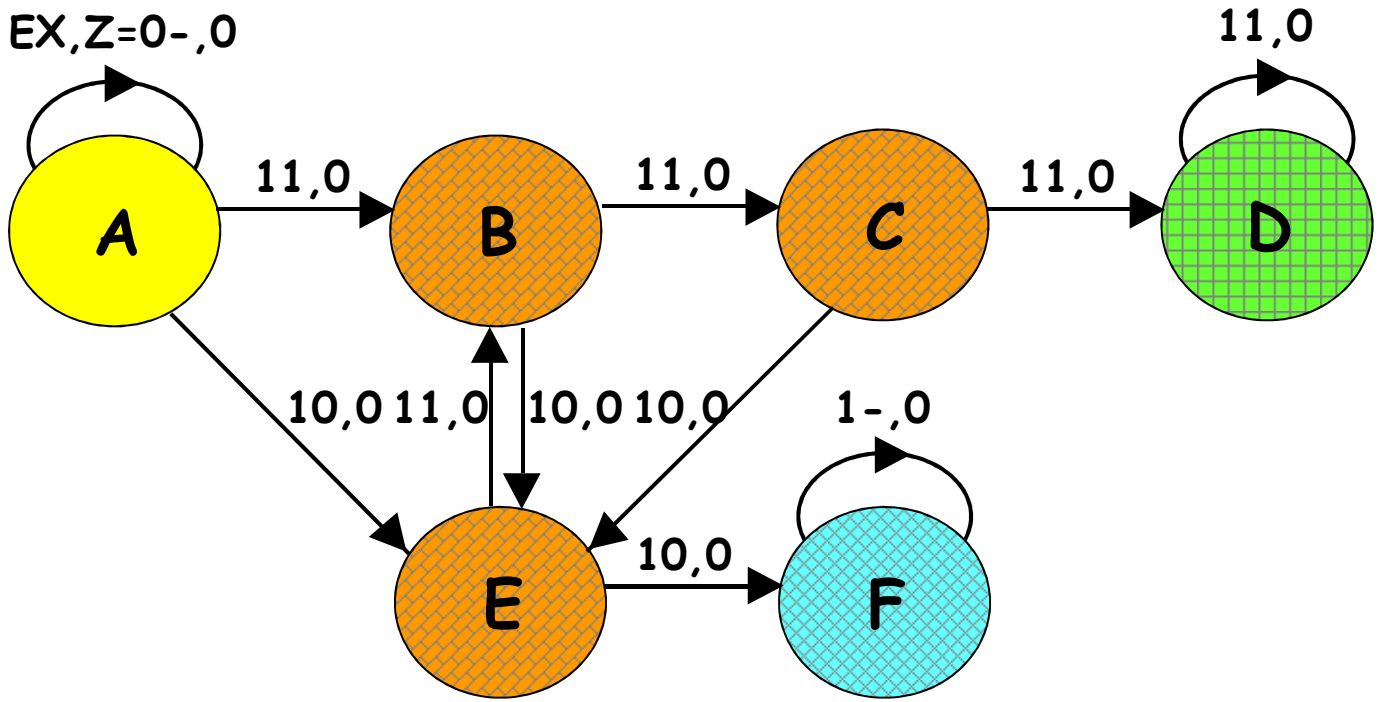
Presenza della sequenza desiderata/indesiderata:

no
no

si
no

no
si

Supponiamo ora che nella parola in corso di elaborazione non sia stata ancora rilevata né la sequenza desiderata, né quella indesiderata, e che tale situazione venga confermata dai successivi simboli applicati in ingresso. Se dopo uno (stato B) o due 1 consecutivi (stato C) si presenta il simbolo 0, da tali stati occorre transitare nello stato E. Se dopo uno 0 (stato E) si presenta il simbolo 1, da tale stato occorre transitare nello stato B.



attesa

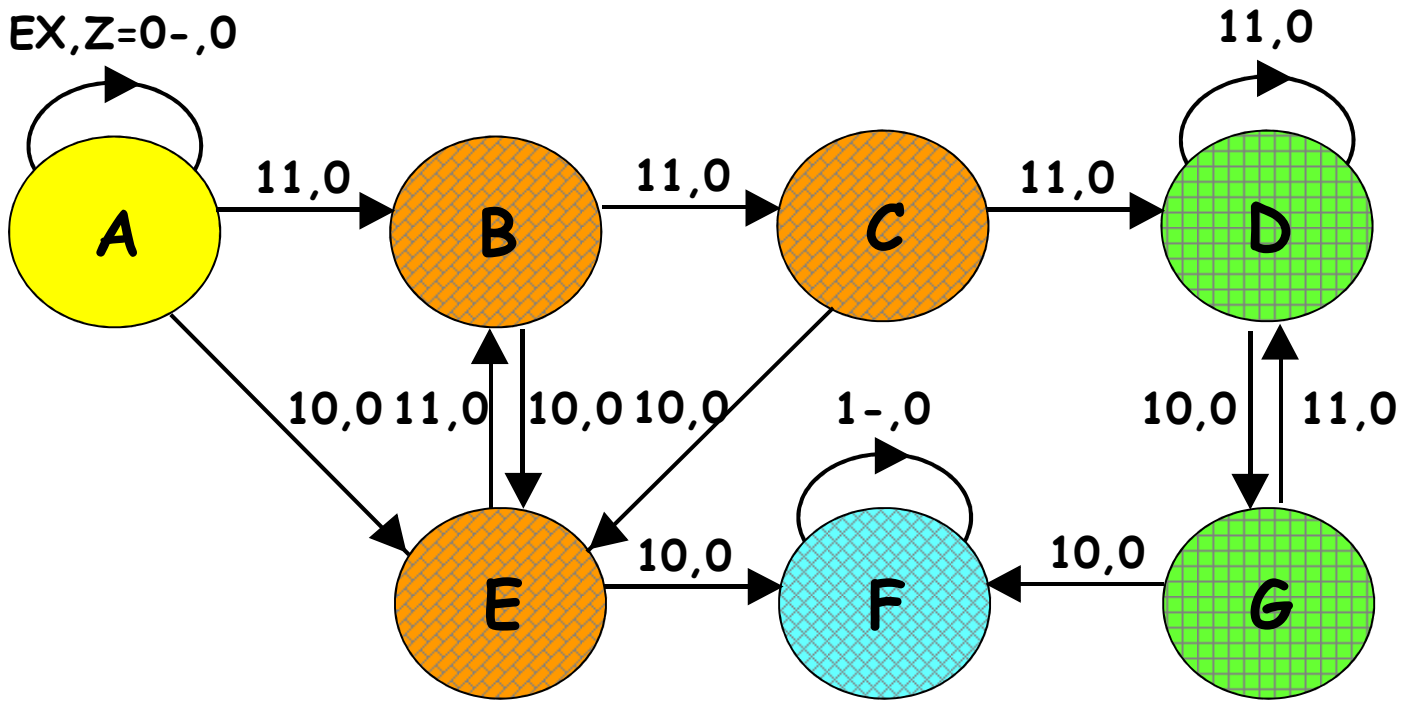
Presenza della sequenza desiderata/indesiderata:

no no

si no

no si

Supponiamo ora che nella parola in corso di elaborazione sia stata rilevata la sequenza desiderata, ma non ancora quella indesiderata. A fronte di un primo 0, dallo stato D occorre transitare in un nuovo stato G (non in E !). Da G si transiterà poi in D o in F a seconda che il successivo simbolo sia 1 o ancora 0, rispettivamente.



attesa

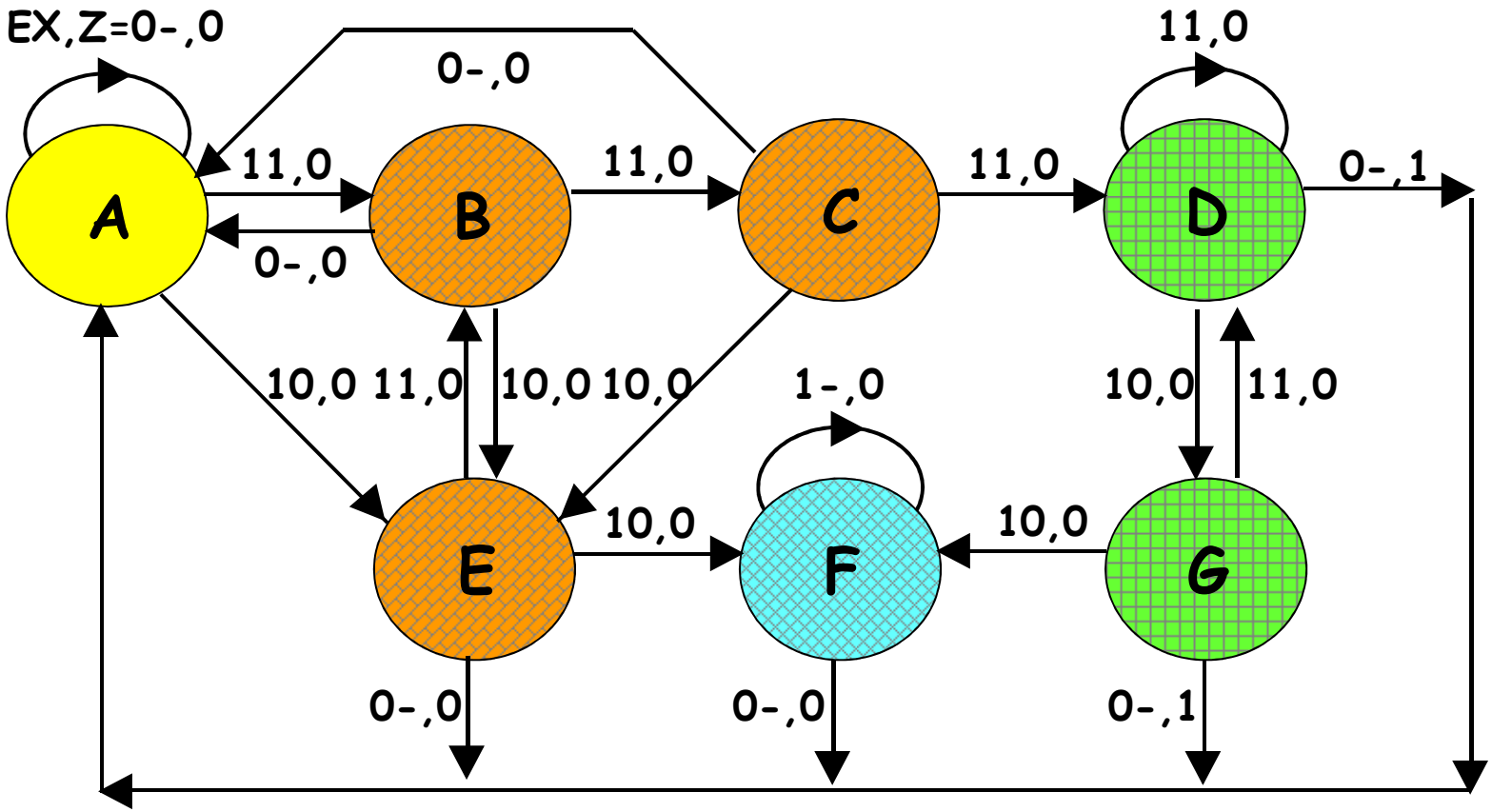
Presenza della sequenza desiderata/indesiderata:

no
no

si
no

~~no~~
si

Occorre infine prevedere, quando E si disattiva, il rientro nello stato di attesa A. Il valore da attribuire corrispondentemente all'uscita Z discende immediatamente, per ogni stato, dal significato associato allo stato stesso.



attesa

Presenza della sequenza desiderata/indesiderata:

no	si	-
no	no	si

Tabella di flusso

	E X = 00	01	11	10
A	A,0	A,0	B,0	E,0
B	A,0	A,0	C,0	E,0
C	A,0	A,0	D,0	E,0
D	A,1	A,1	D,0	G,0
E	A,0	A,0	B,0	F,0
F	A,0	A,0	F,0	F,0
G	A,1	A,1	D,0	F,0

non riducibile

Tabella delle transizioni

$(y_1y_2y_3)^n$ \ $(E\,X)^n$	00	01	11	10
(A) 000	000,0	000,0	001,0	100,0
(B) 001	000,0	000,0	010,0	100,0
(D) 011	000,1	000,1	011,0	111,0
(C) 010	000,0	000,0	011,0	100,0
(E) 100	000,0	000,0	001,0	101,0
(F) 101	000,0	000,0	101,0	101,0
(G) 111	000,1	000,1	011,0	101,0
110	---,-	---,-	---,-	---,-

$(y_1y_2y_3)^{n+1}, Z^n$

Rete combinatoria di uscita e di aggiornamento dello stato

$$Z^n = (y_2\ y_3\ E')^n$$

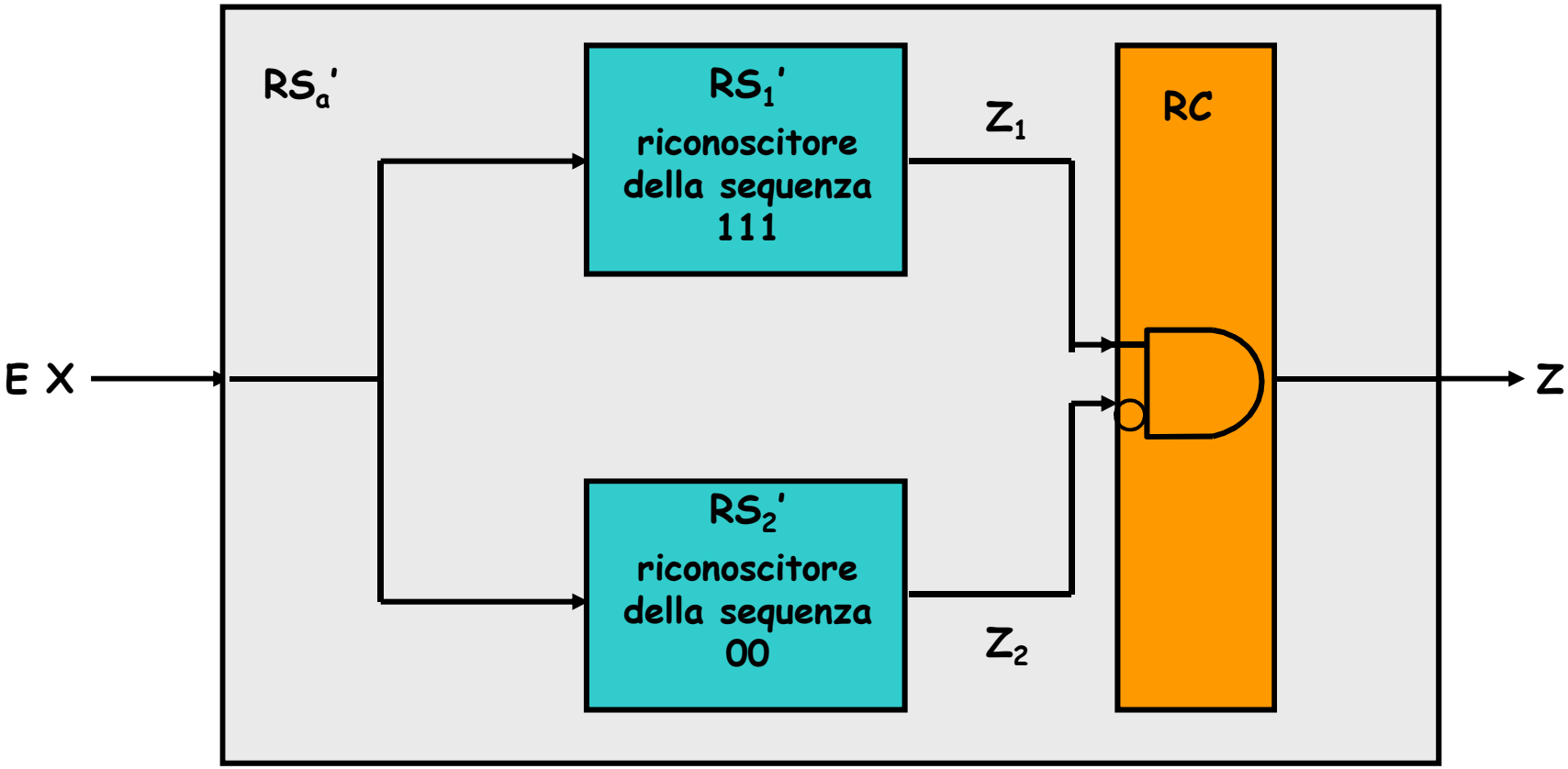
$$J_1^n = (E\ X')^n$$

$$K_1^n = (E' + y_3'\ X + y_2\ X)^n$$

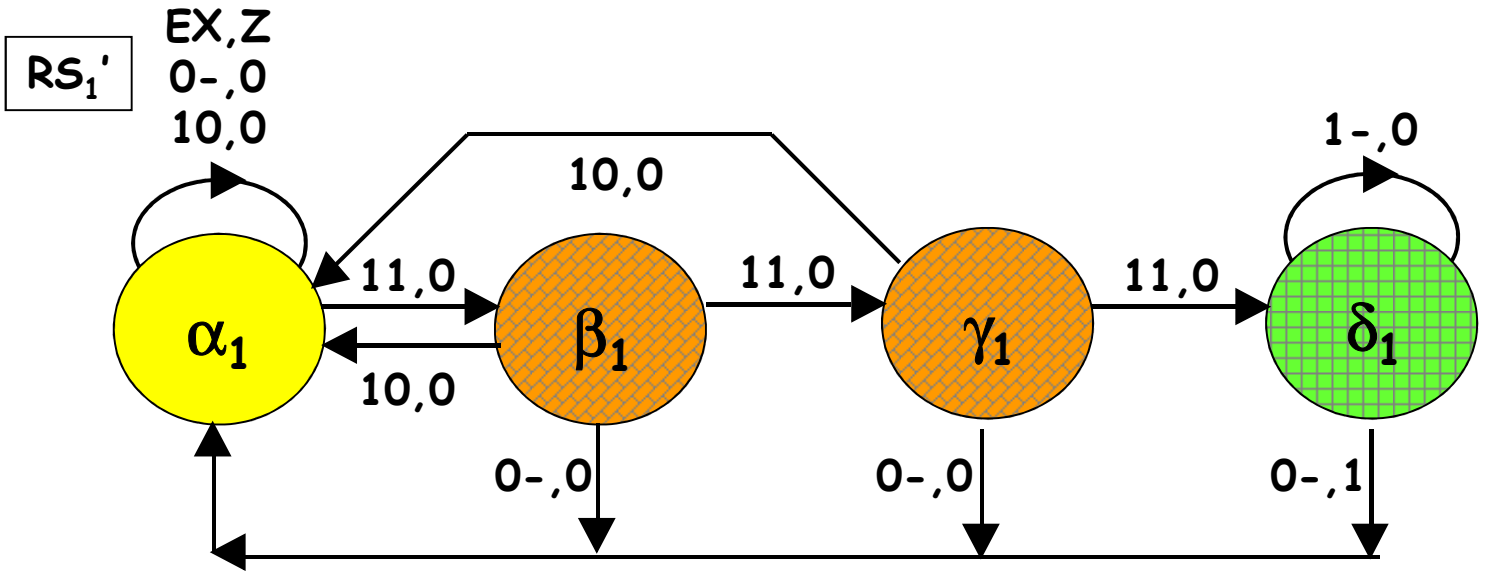
similmente
J_2^n, K_2^n, J_3^n, K_3^n

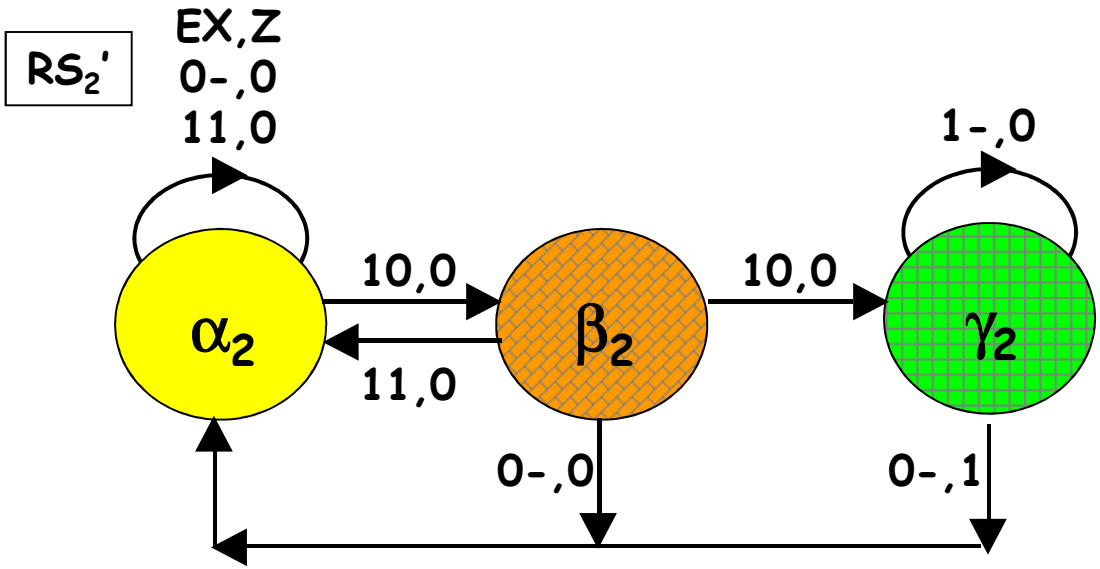
Alternativa alla realizzazione precedente (RS_a), è la seguente (RS_a') che deriva dall'applicazione del:

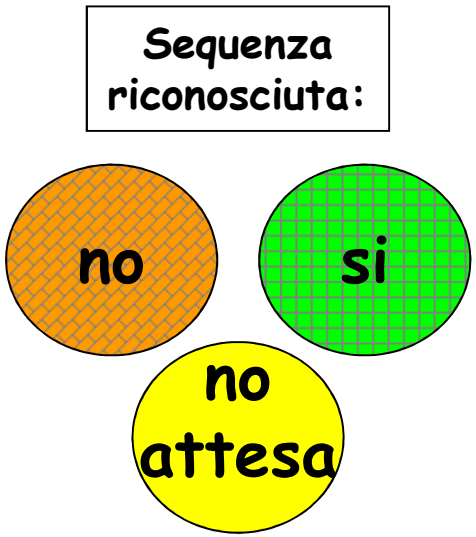
Principio di decomposizione:
una macchina sequenziale M con N stati può essere decomposta in due macchine più semplici M_1 e M_2, contraddistinte da un numero di stati N_1 e N_2, rispettivamente, tali che $N_1 < N$, $N_2 < N$, $N_1 \times N_2 \geq N$.



Il diagramma degli stati dei 2 riconoscitori di sequenza per la realizzazione della rete secondo il modello RS_a'

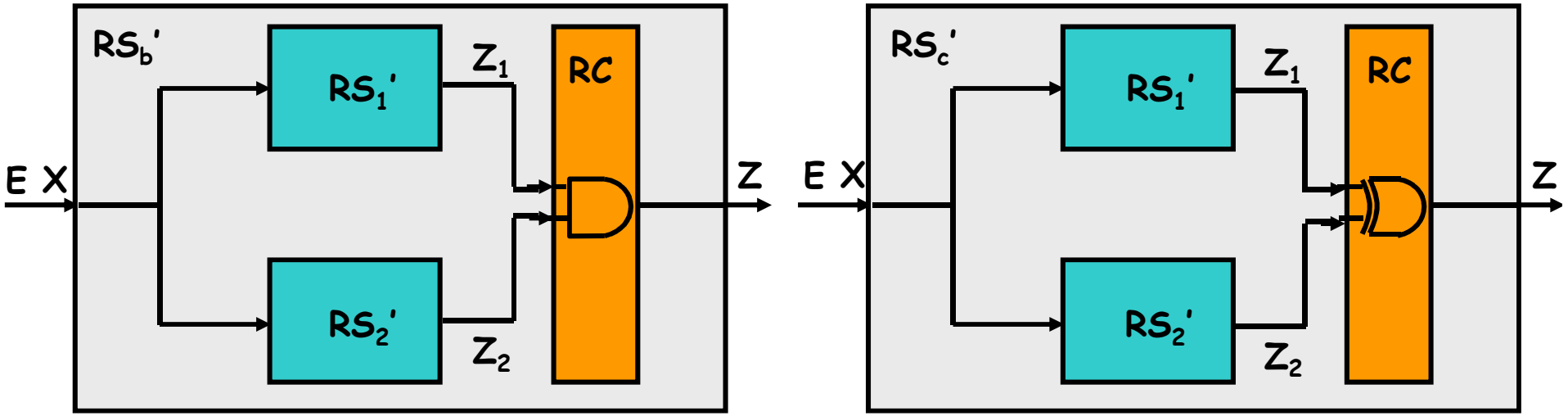






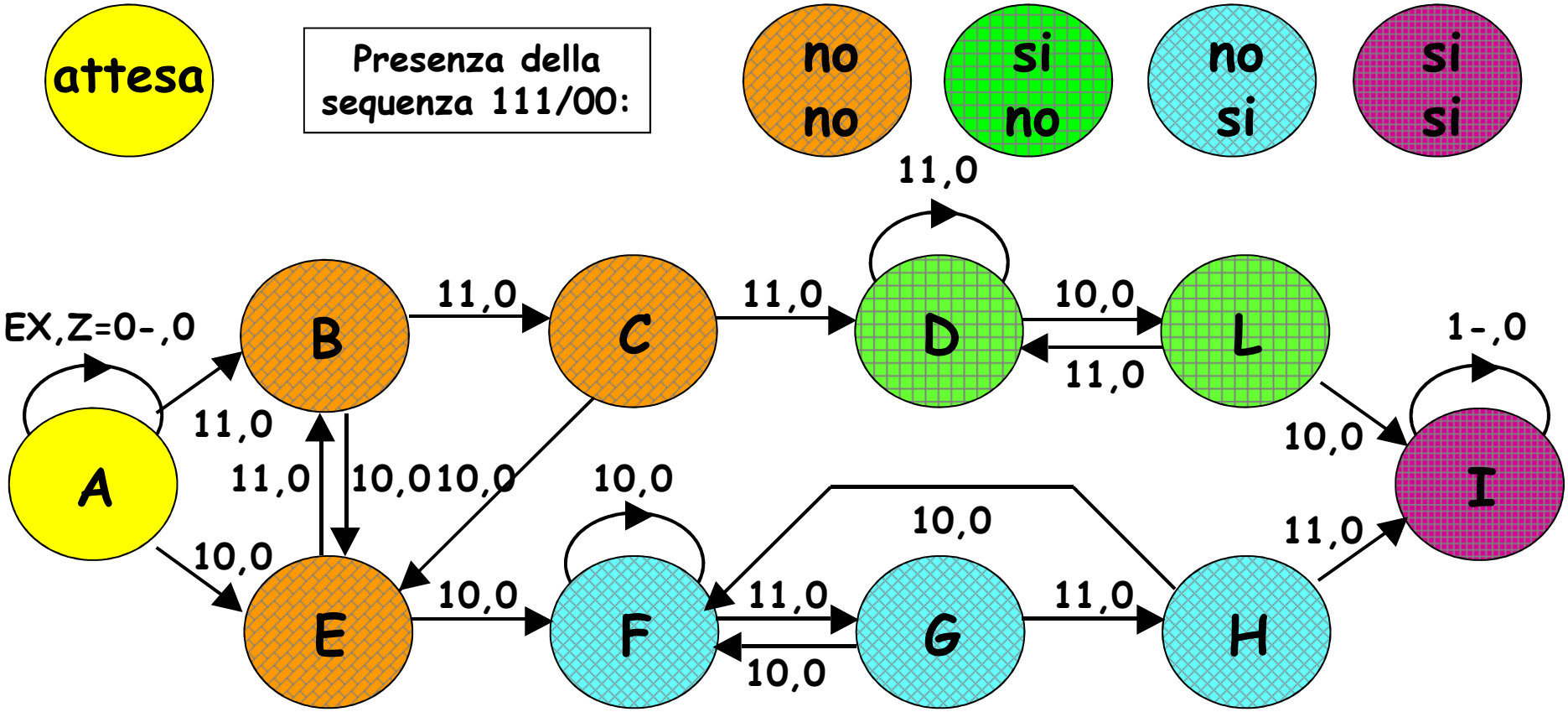
L'applicazione del principio di decomposizione consente non solo di semplificare e meglio strutturare il progetto, ma anche di riutilizzarne parti a fronte di eventuali variazioni delle specifiche.

Se, ad esempio, la specifica di progetto dovesse variare, prevedendo che l'uscita Z si attivi qualora una parola contenga non più (a) solo la sequenza 111 e non la sequenza 00, bensì (b) entrambe le sequenze, oppure (c) una qualunque delle due, la soluzione RS_a' precedentemente identificata applicando il principio di decomposizione sarebbe ampiamente riutilizzabile, essendo sufficiente modificare soltanto la rete combinatoria di uscita:



L'approccio convenzionale, al contrario, non consente il riutilizzo della soluzione RS_a ai fini della realizzazione delle nuove reti RS_b e RS_c.

I diagrammi degli stati per le reti RS_b e RS_c (... e RS_a)



Occorre infine prevedere la transizione in A da ogni altro stato per EX=0-.
Il valore da attribuire corrispondentemente a Z dipende dalla specifica di progetto, ovvero Z=1 se:

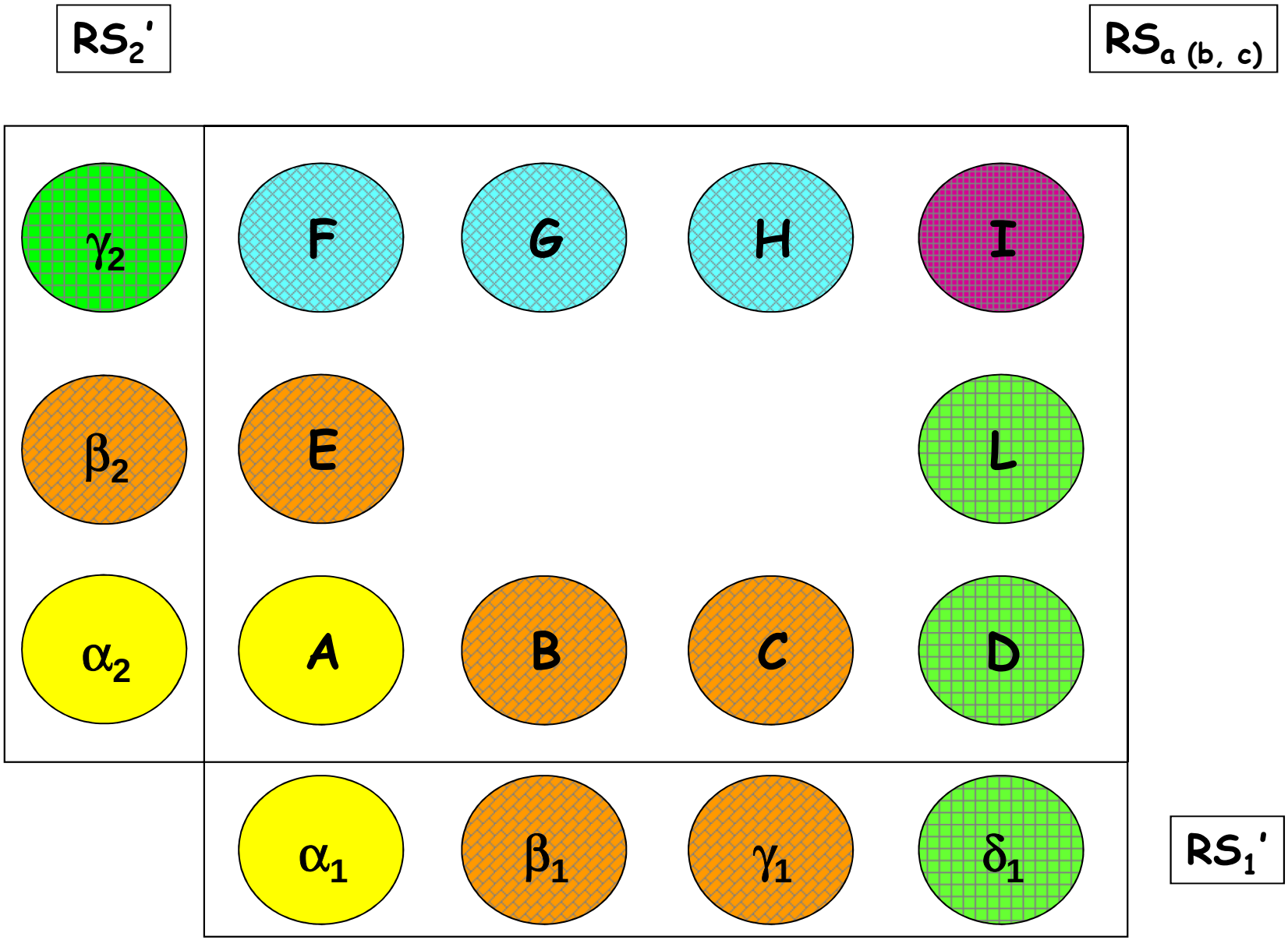
(RS_b)

(RS_c)

Il diagramma può essere utilizzato anche per la sintesi di RS_a.
In tal caso Z=1 se:

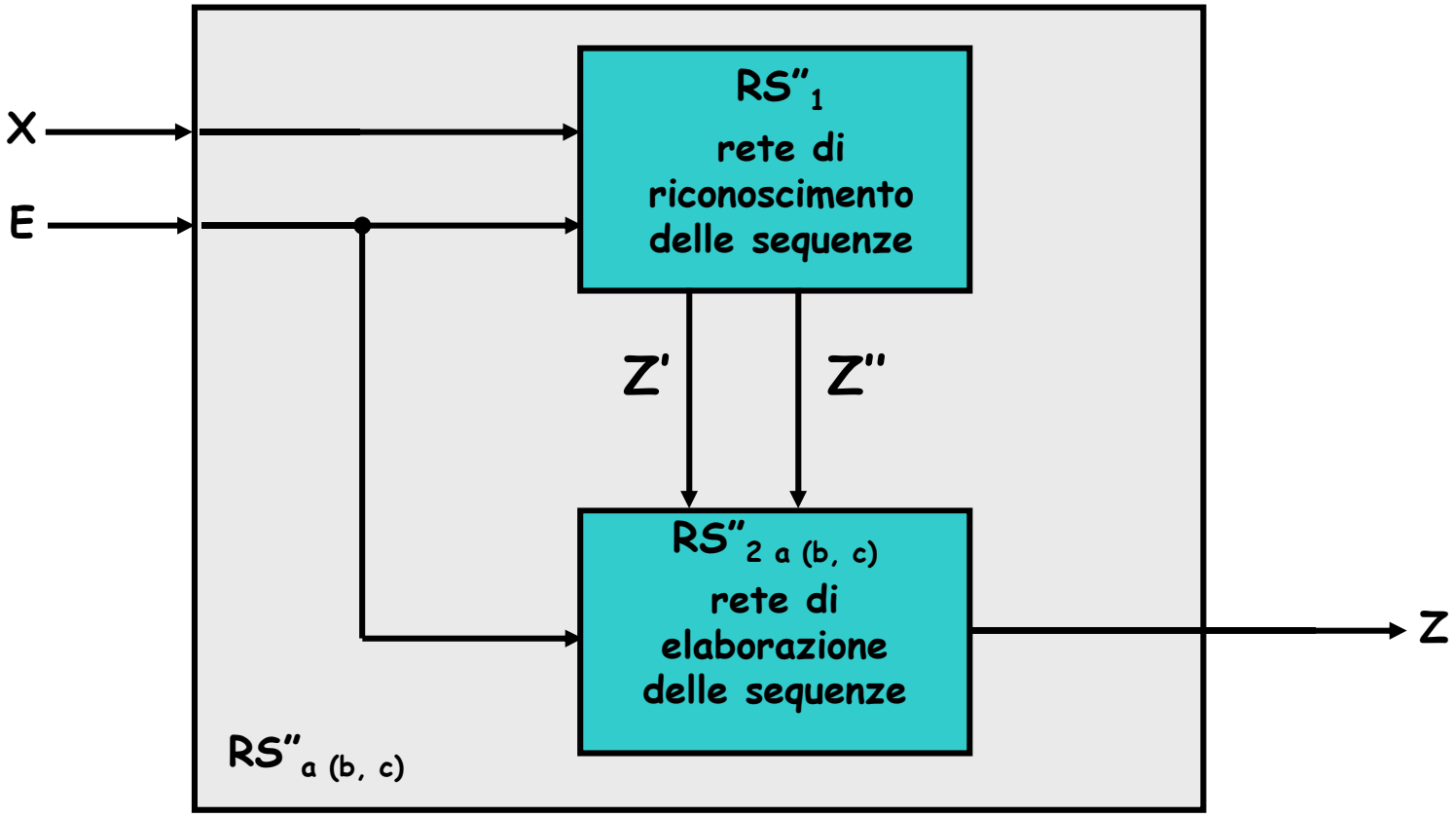
(RS_a)

La corrispondenza tra gli stati di RS_1' e RS_2' e gli stati di RS_a (o di RS_b o di RS_c)



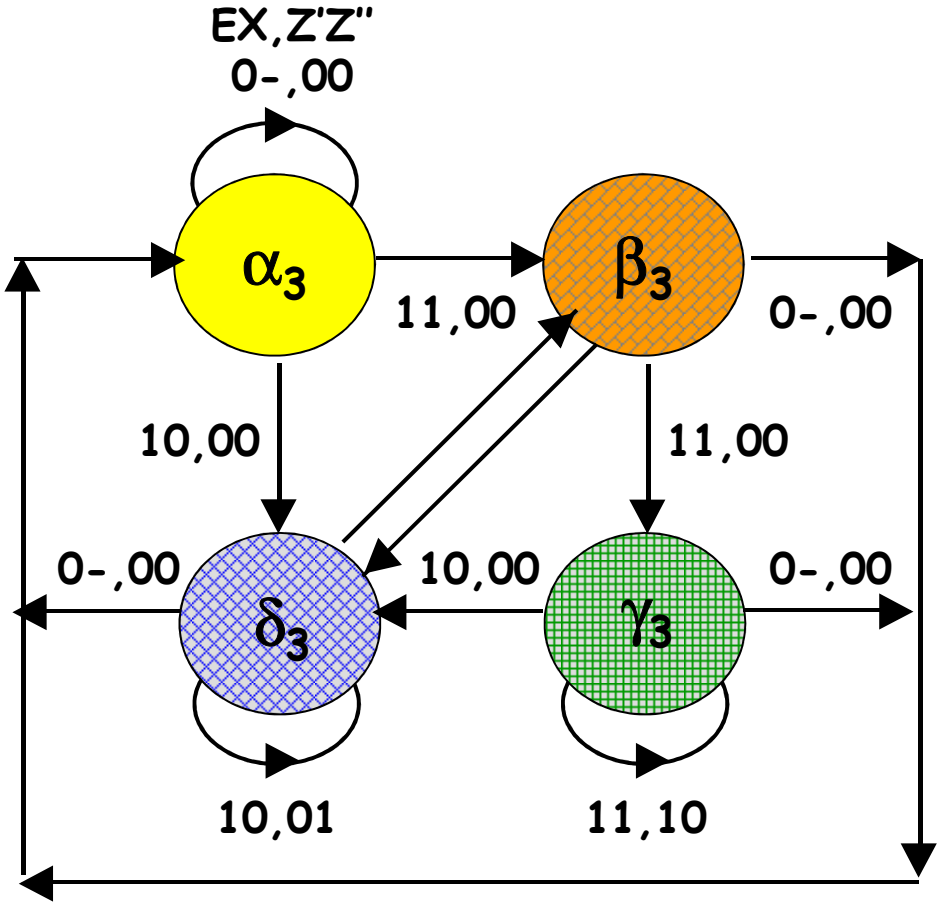
L'applicazione del principio di decomposizione, ovvero la strutturazione di una rete in sottoreti più semplici opportunamente interconnesse, in generale non è univoca.

Si potrebbe infatti adottare anche il seguente modello, che prevede due sottoreti sequenziali contraddistinte da funzionalità più ortogonali rispetto al modello precedente:



La rete di riconoscimento delle sequenze (RS''_1)

Tale rete ha il compito di attivare Z' o Z'' ogni qualvolta rileva in una parola la stringa 111 o 00, rispettivamente.
Il comportamento della rete è invariante rispetto alla regola (a, b, c) di elaborazione delle sequenze selezionate.



attesa

numero di 1 consecutivi in precedenza rilevati:

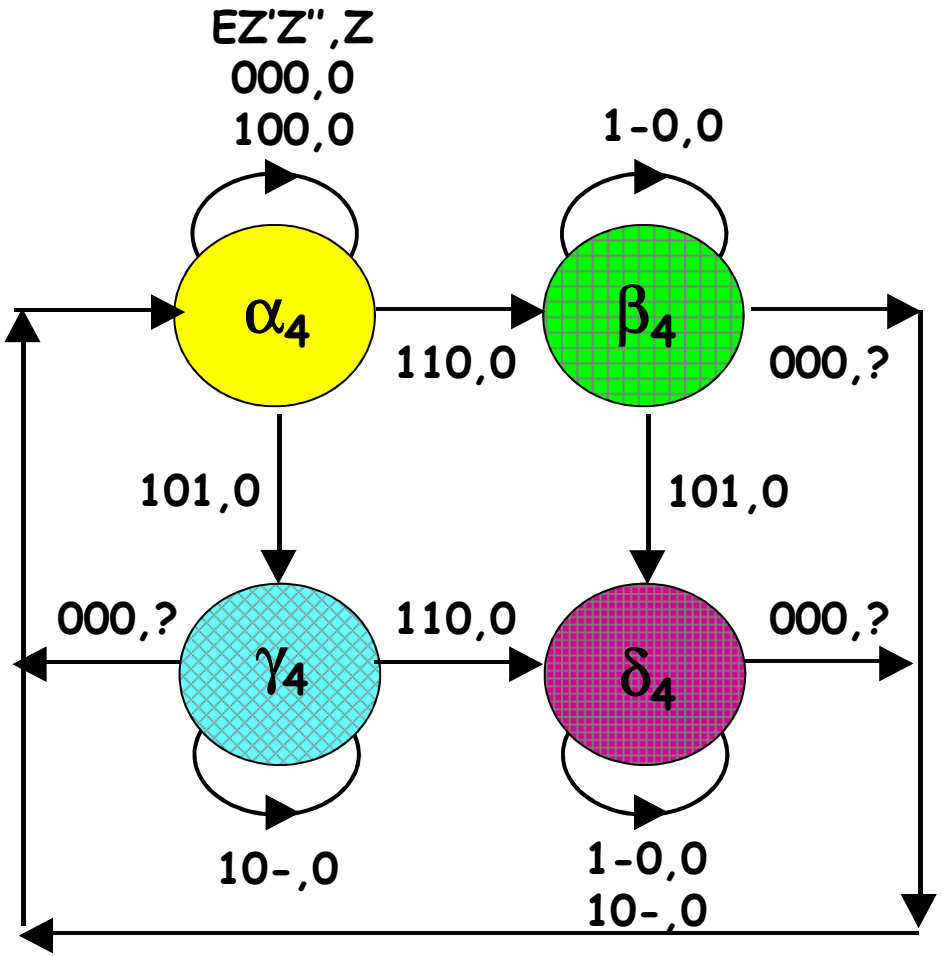
1 ≥2

numero di 0 consecutivi in precedenza rilevati:

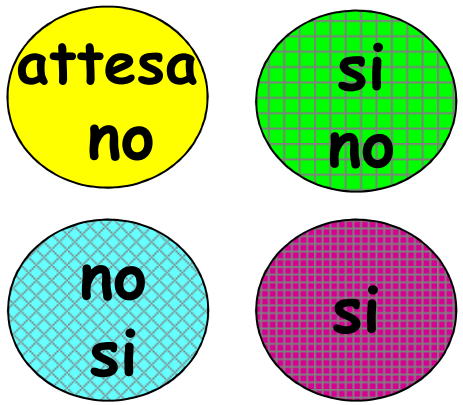
≥1

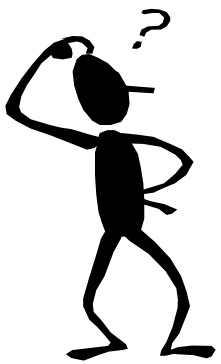
Le reti di elaborazione delle sequenze (RS"$_{2a\ (b,\ c)}$)

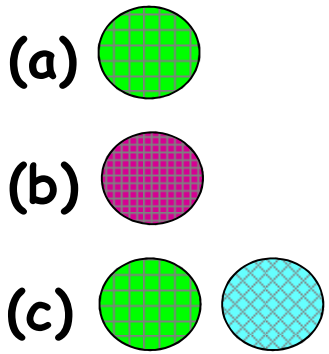
Tali reti hanno il compito di generare il segnale di uscita Z
in accordo alla assegnata regola di gestione delle due sequenze.
Il comportamento delle reti non dipende dalle sequenze selezionate.



Presenza delle sequenze 111/00:

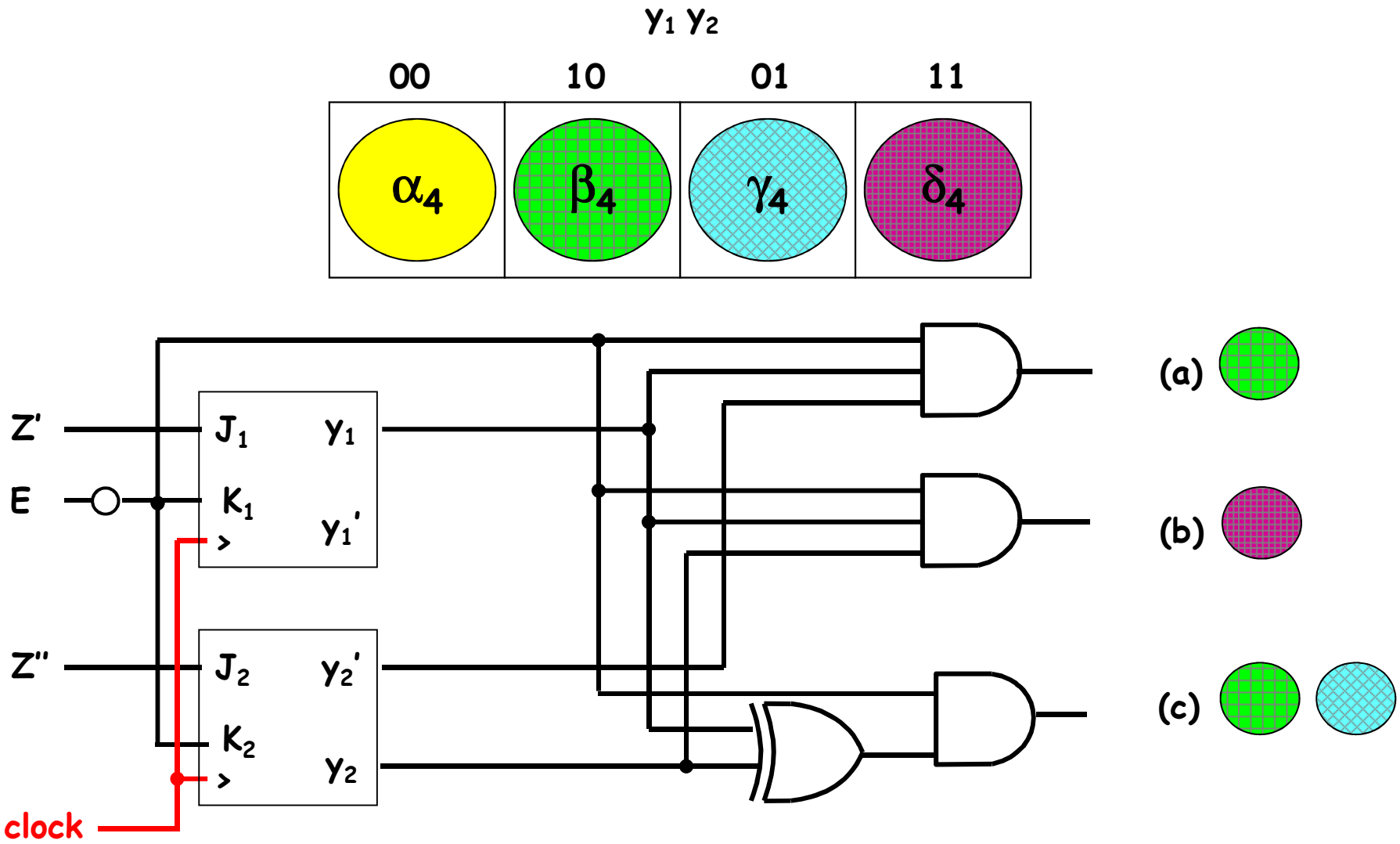




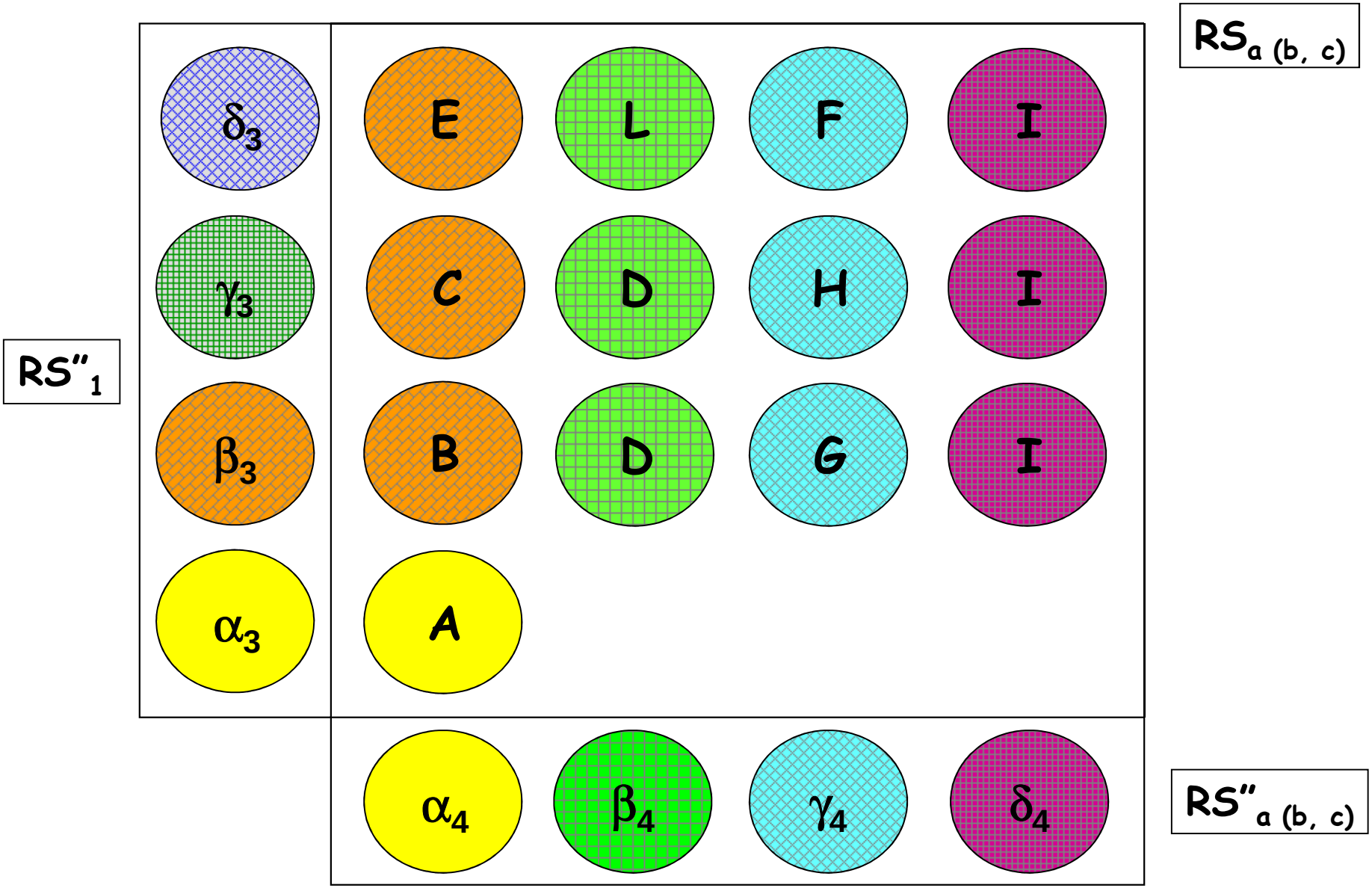


Le reti di elaborazione delle sequenze (RS"$_{2a\ (b,\ c)}$)

Schema logico



La corrispondenza tra gli stati di RS''_1 e RS''_{2a} (o RS''_{2b} o RS''_{2c}) e gli stati di RS_a (o RS_{2b} o RS_{2c})



Esercizio 2

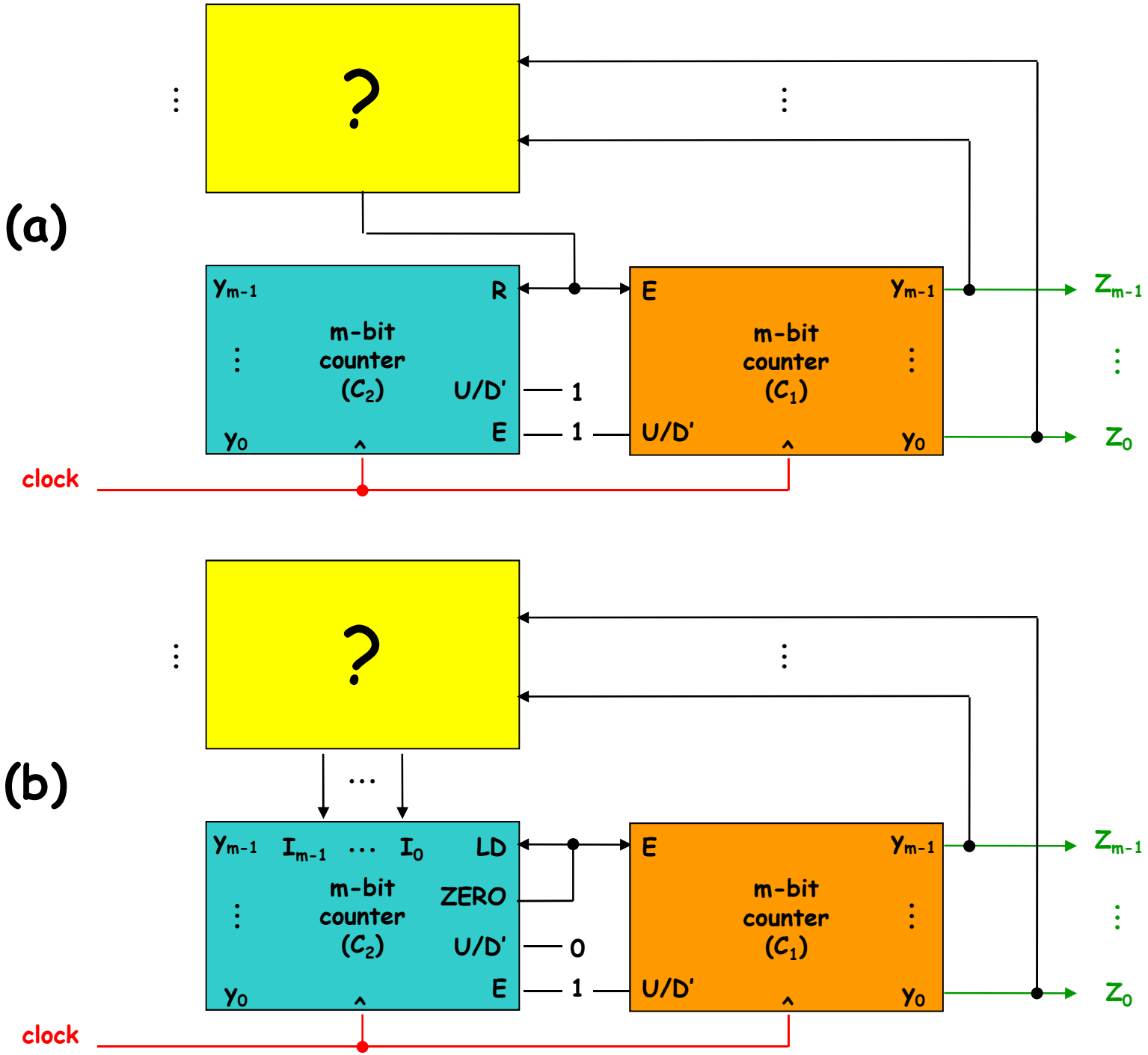
Un sistema digitale, caratterizzato da m segnali di uscita Z_{m-1} (MSB), …, Z_1, Z_0 (LSB), deve ciclicamente generare in uscita le 2^m configurazioni di m bit nell'ordine previsto dal sistema di numerazione binario, replicando consecutivamente ciascuna di esse per un numero di volte pari al corrispondente valore decimale. Con riferimento ad esempio al caso m = 3, la sequenza di configurazioni (sinteticamente rappresentate in notazione decimale) che il sistema deve ciclicamente fornire in uscita è:

$$(Z_2Z_1Z_0)_{dieci} = 0\ 1\ 1\ 2\ 2\ 2\ 3\ 3\ 3\ 3\ 4\ 4\ 4\ 4\ 4\ 5\ 5\ 5\ 5\ 5\ 6\ 6\ 6\ 6\ 6\ 6\ 7\ 7\ 7\ 7\ 7\ 7\ 7\ 7$$

1. Si calcoli, in funzione di m, il numero di stati dell'automa che modella il comportamento del sistema.

Il sistema deve essere progettato in conformità al "principio di decomposizione", secondo il quale una macchina sequenziale M con N stati può essere decomposta in due macchine più semplici M_1 e M_2, contraddistinte rispettivamente da N_1 e N_2 stati, con $N_1 < N$, $N_2 < N$, $N_1 \times N_2 \geq N$. In tale ottica il sistema può essere convenientemente strutturato in un contatore binario in avanti a m bit (C_1), le cui variabili di stato y_{m-1}, …, y_1, y_0 ordinatamente coincidono con i segnali di uscita Z_{m-1}, …, Z_1, Z_0, e nella relativa unità di controllo (CU), cui è demandato il compito di gestire opportunamente il segnale di abilitazione al conteggio di C_1 in modo tale che da esso si ottenga direttamente in uscita la desiderata sequenza di configurazioni.

2. Si identifichino due alternative realizzazioni per CU, basate entrambe, come indicato in figura, su un contatore binario unidirezionale a m bit (C_2), rispettivamente del tipo:
   b. in avanti, con comando di reset (R) sincrono;
   a. all'indietro, con comando di load (LD) sincrono e annessi ingressi I_{m-1}, …, I_1, I_0.


(a)
?
y_{m-1}
R
m-bit
counter
(C_2)
U/D'
1
E
1
y_0
E
y_{m-1}
m-bit
counter
(C_1)
U/D'
y_0
z_{m-1}
z_0
clock
(b)
?
...
y_{m-1} I_{m-1} ... I_0
LD
m-bit
counter
(C_2)
ZERO
U/D'
0
E
1
y_0
E
y_{m-1}
m-bit
counter
(C_1)
U/D'
y_0
z_{m-1}
z_0
clock

Per entrambe le realizzazioni si evidenzi esplicitamente sia il ruolo svolto da C_2 durante il processo di generazione delle configurazioni di uscita (ovvero il significato associato a ciascuno suo stato interno), sia, con riferimento al caso m = 3, la sequenza degli stati interni di C_2 secondo la seguente notazione:

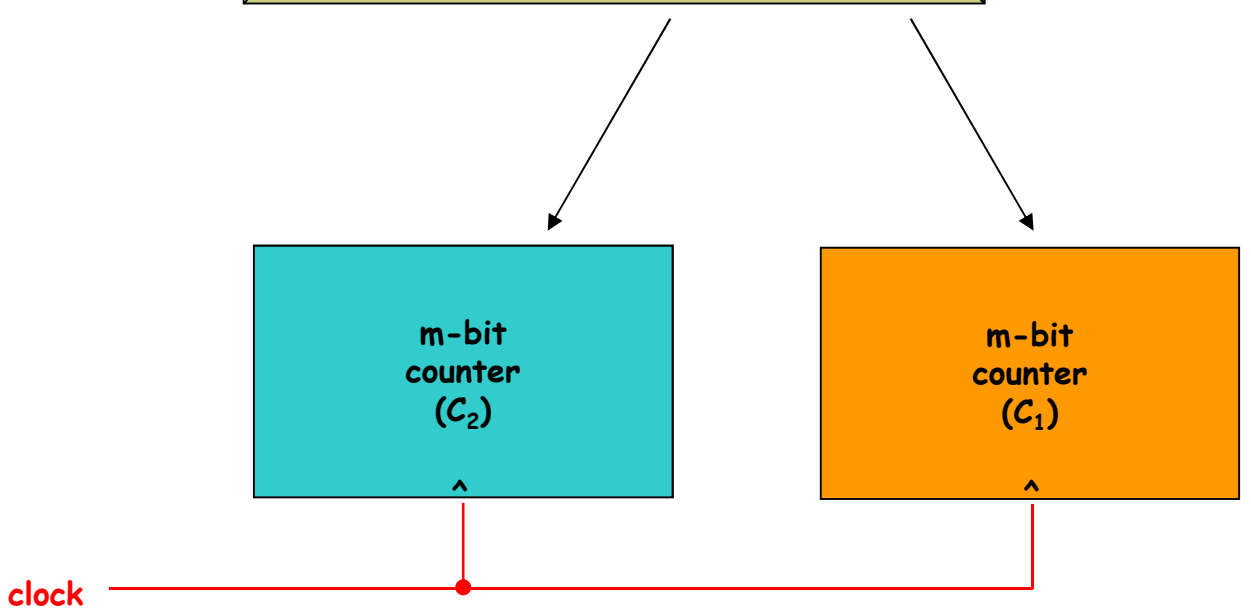
C_1	0	1	1	2	2	2	3	3	3	3	4	4	4	4	4	5	5	5	5	5	5	6	6	6	6	6	6	6	7	7	7	7	7	7	7	7
C_2																																				a
																																				b

3. Si individuino le estensioni da apportare alle due precedenti realizzazioni (avvalendosi degli ulteriori ingressi di controllo dei contatori C_1 e C_2 ed usando i componenti ritenuti più idonei allo scopo) nell'ipotesi che le specifiche di progetto prevedano che il sistema generi, sempre secondo le stesse modalità, non necessariamente tutte le 2^m configurazioni di uscita, ma soltanto le prime k (ovvero le configurazioni 0, 1, …, k-1) con $k \in \{1, 2, \ldots, 2^m\}$, essendo l'ultima configurazione della sequenza desiderata codificata in binario e staticamente definita tramite m segnali X_{m-1} (MSB), …, X_1, X_0 (LSB) applicati in ingresso al sistema. A titolo esemplificativo, le possibili sequenze di configurazioni che il sistema deve pertanto essere in grado di generare in uscita con riferimento al caso m = 3 sono:

$X_2X_1X_0$	k	
000	1	$(Z_2Z_1Z_0)_{dieci}$ = 0
001	2	$(Z_2Z_1Z_0)_{dieci}$ = 0 1 1
010	3	$(Z_2Z_1Z_0)_{dieci}$ = 0 1 1 2 2 2
011	4	$(Z_2Z_1Z_0)_{dieci}$ = 0 1 1 2 2 2 3 3 3 3
100	5	$(Z_2Z_1Z_0)_{dieci}$ = 0 1 1 2 2 2 3 3 3 3 4 4 4 4 4
101	6	$(Z_2Z_1Z_0)_{dieci}$ = 0 1 1 2 2 2 3 3 3 3 4 4 4 4 4 5 5 5 5 5
110	7	$(Z_2Z_1Z_0)_{dieci}$ = 0 1 1 2 2 2 3 3 3 3 4 4 4 4 4 5 5 5 5 5 6 6 6 6 6 6
111	8	$(Z_2Z_1Z_0)_{dieci}$ = 0 1 1 2 2 2 3 3 3 3 4 4 4 4 4 5 5 5 5 5 6 6 6 6 6 6 7 7 7 7 7 7 7

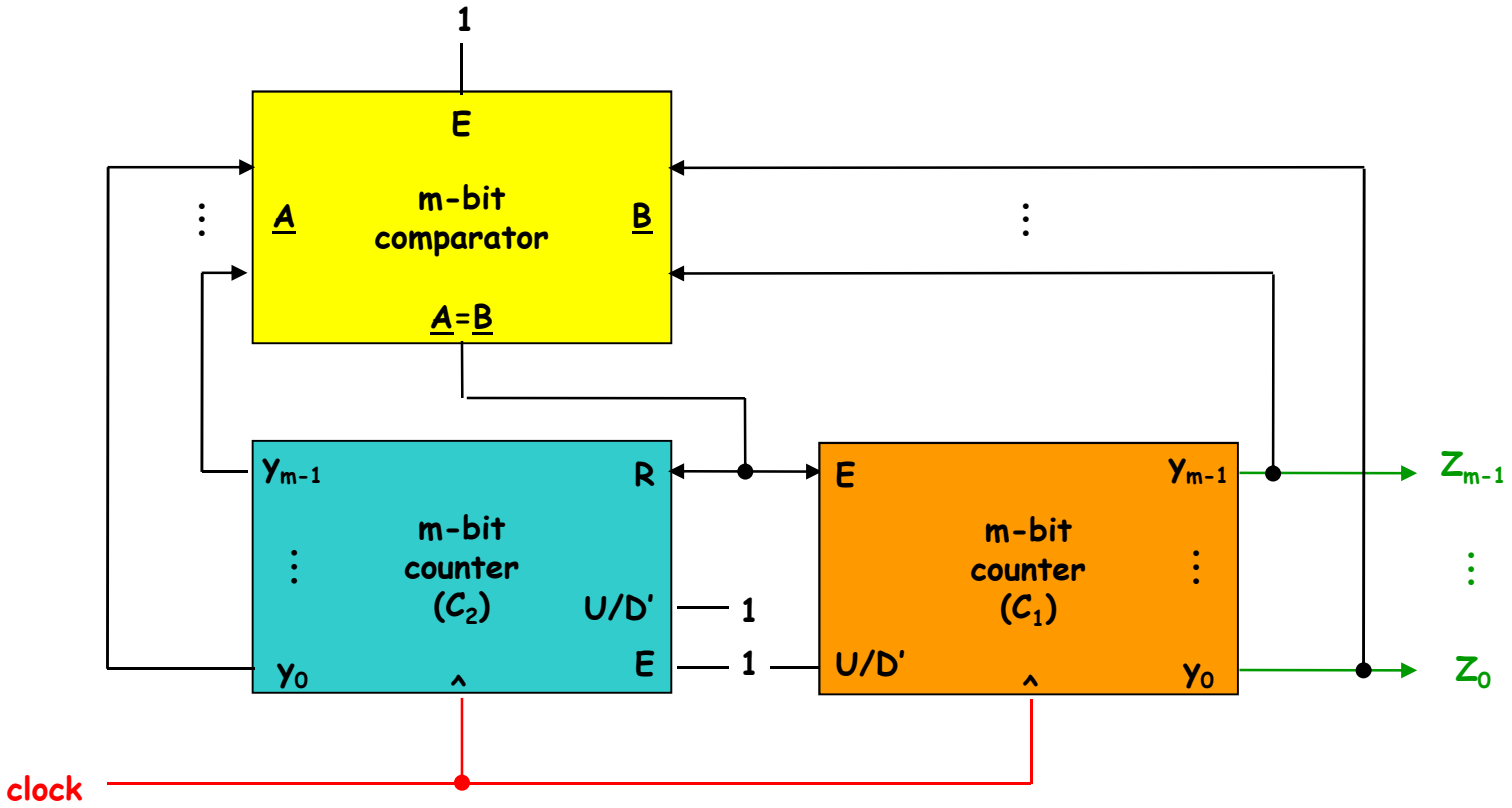
Calcolo del numero degli stati dell'automa che modella il comportamento del sistema

$$N = \sum_{k=0}^{2^m-1} (1+k) = \left(\frac{2^m+1}{2} \right) 2^m$$



Schema realizzativo (a)

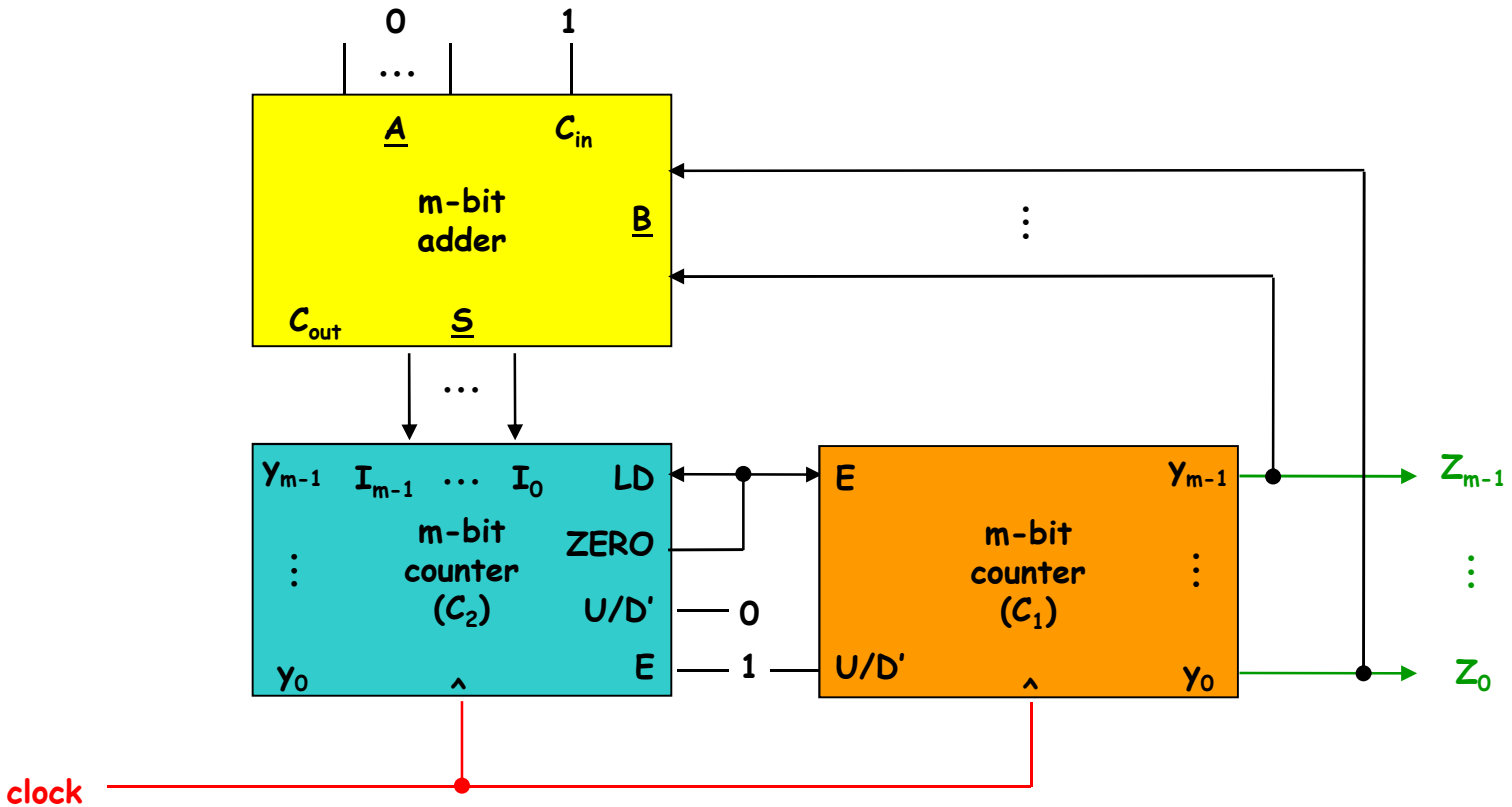
Ciascuno stato interno di C_2 indica quante volte è stata replicata la configurazione correntemente fornita in uscita da C_1.



C_1	0	1	1	2	2	2	3	3	3	3	4	4	4	4	4	5	5	5	5	5	5	6	6	6	6	6	6	6	7	7	7	7	7	7	7	7
C_2	0	0	1	0	1	2	0	1	2	3	0	1	2	3	4	0	1	2	3	4	5	0	1	2	3	4	5	6	0	1	2	3	4	5	6	7

Schema realizzativo (b)

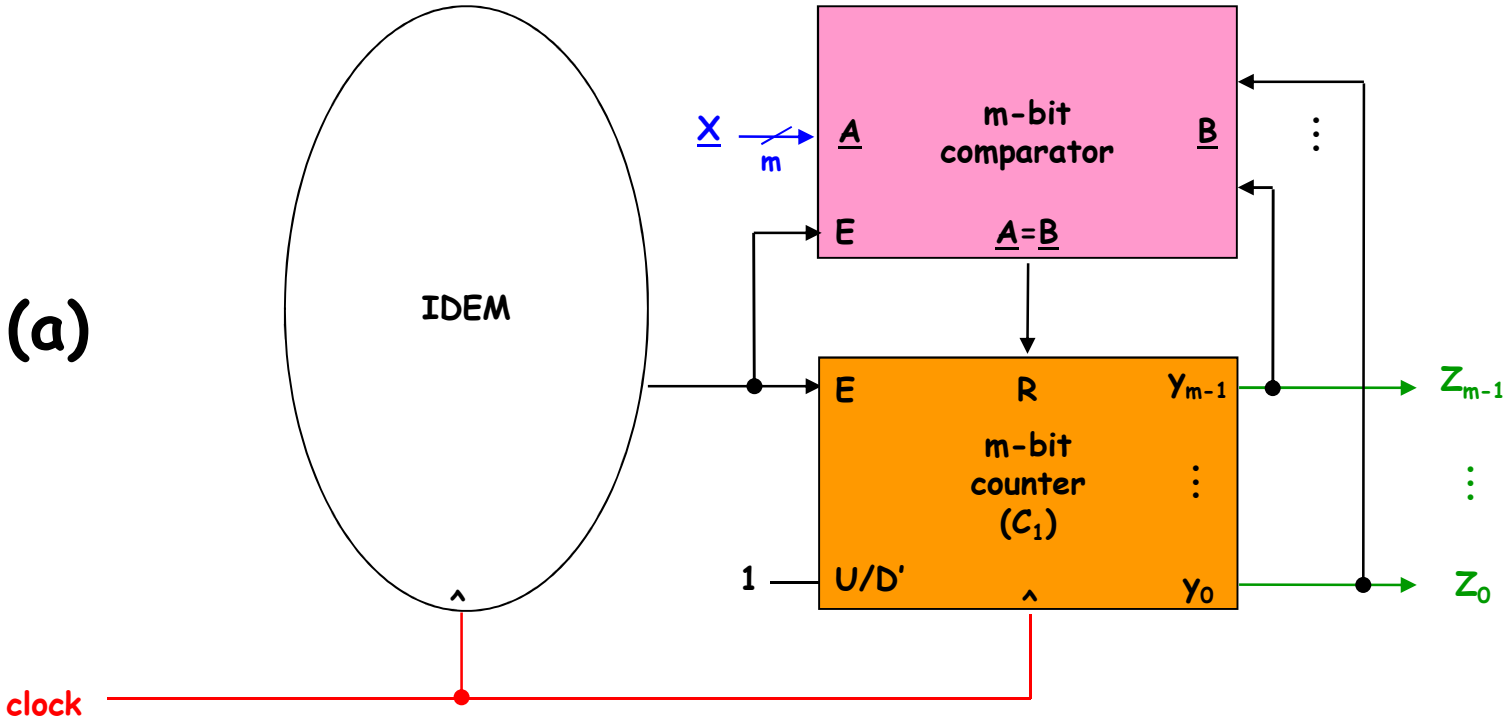
Ciascuno stato interno di C_2 indica quante volte dovrà essere replicata la configurazione correntemente fornita in uscita da C_1.



C_1	0	1	1	2	2	2	3	3	3	3	4	4	4	4	4	5	5	5	5	5	5	6	6	6	6	6	6	6	7	7	7	7	7	7	7	7
C_2	0	1	0	2	1	0	3	2	1	0	4	3	2	1	0	5	4	3	2	1	0	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0

Estensione schemi realizzativi

(a)



(b)

Unica variante: il segnale di uscita del comparatore è applicato anche all'ingresso di Reset di C_2.


Macchine
con ampiezza
di memoria
finita

Esercizio 1

Una rete sequenziale sincrona è caratterizzata da un unico segnale di ingresso (X) e da un unico segnale di uscita (Z), entrambi sincroni.

In ogni intervallo di clock l'uscita Z deve assumere il valore logico 1 soltanto se gli ultimi quattro valori di X costituiscono una palindrome.

Si identifichi:

- l'automa minimo della rete secondo il modello di Mealy;
- una possibile realizzazione della rete con FF-D e gate elementari.

X	0	1	1	0	1	0	0	1	1	1	1	1	…
Z				1	0	0	0	1	0	0	1	1	…

Diagramma degli stati (modello di Mealy)

Occorre prevedere otto distinti stati, ciascuno dei quali identifica una ben precisa configurazione dei valori assunti da X nei tre precedenti intervalli di clock.
Denotando gli stati direttamente con tali configurazioni binarie ed assumendo che in esse i bit siano ordinati in modo tale che l'ultimo a destra si riferisca all'ultimo valore assunto da X, è immediato costruire il diagramma degli stati:

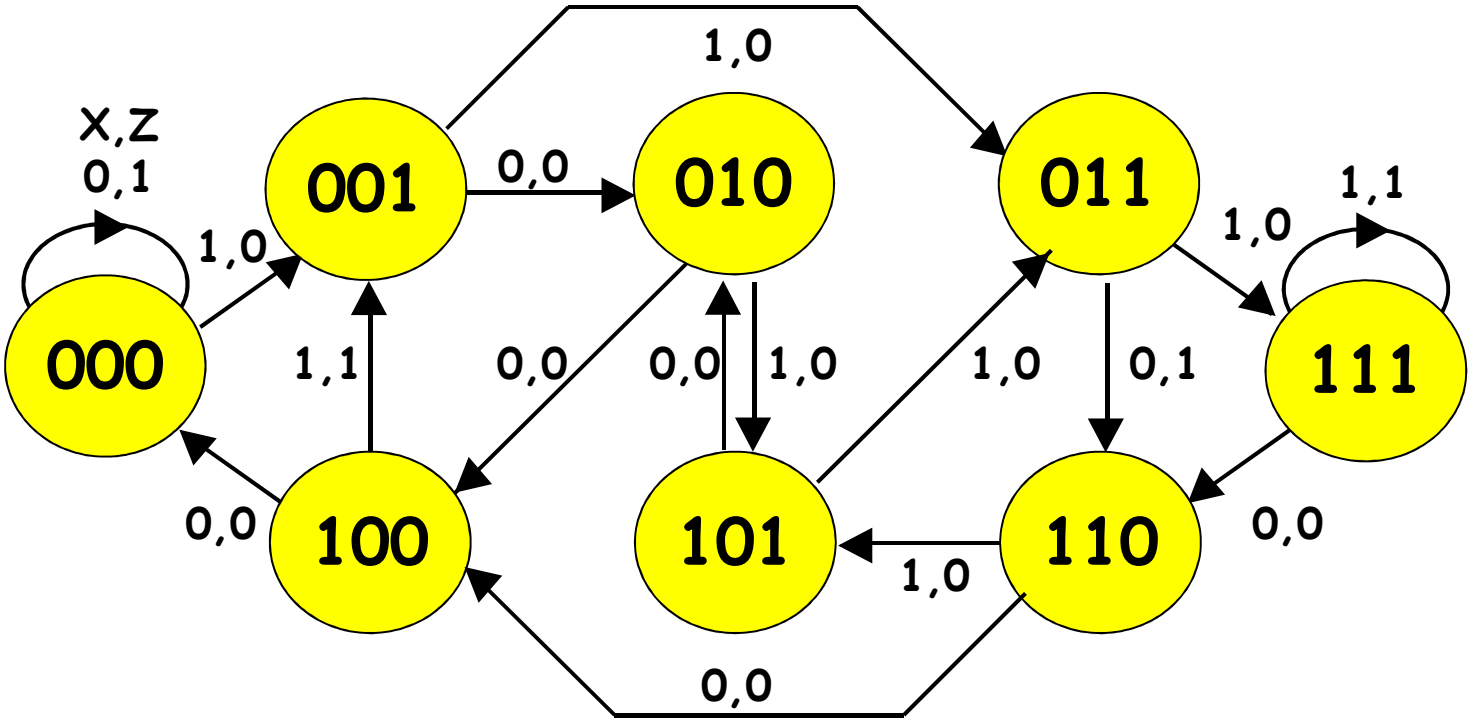
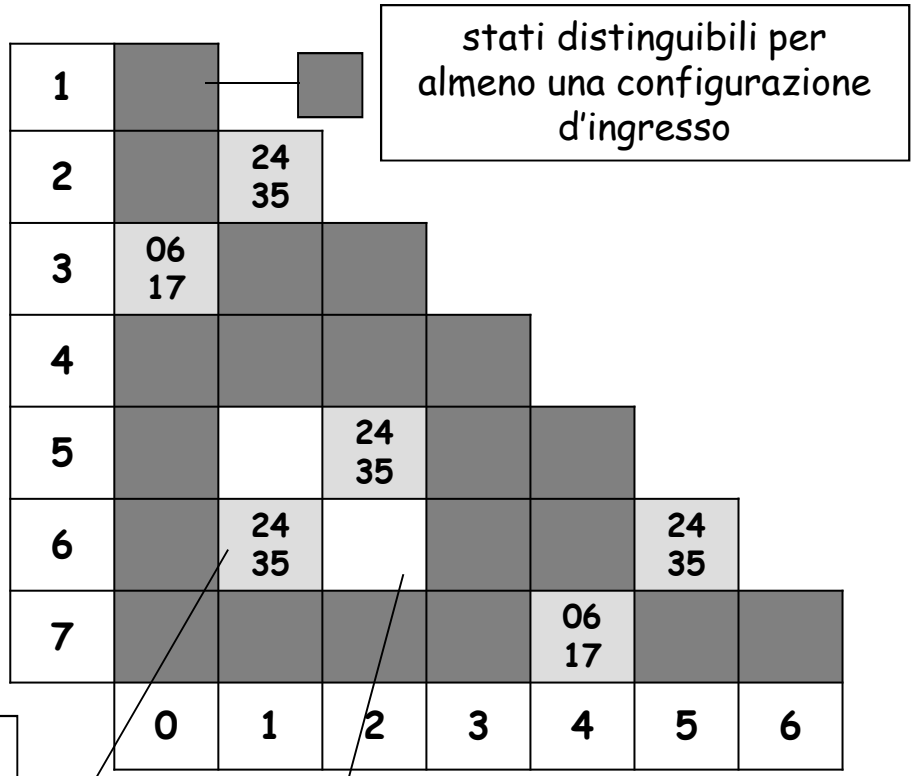


Tabella di flusso e tabella triangolare

	X = 0	X = 1
0	0,1	1,0
1	2,0	3,0
2	4,0	5,0
3	6,1	7,0
4	0,0	1,1
5	2,0	3,0
6	4,0	5,0
7	6,0	7,1



Classi di indistinguibilità {0}, {1,5}, {2,6}, {3}, {4}, {7}

Tabella di flusso minima

		X = 0	X = 1
	0	0,1	1,0
{1,5}	1	2,0	3,0
{2,6}	2	4,0	1,0
	3	2,1	7,0
	4	0,0	1,1
	7	2,0	7,1

Tabella delle transizioni

$(y_3y_2y_1)^n$ \ X^n	0	1
000	000,1	001,0
001	010,0	011,0
011	010,1	111,0
010	100,0	001,0
100	000,0	001,1
101	---,-	---,-
111	010,0	111,1
110	---,-	---,-

$(y_3y_2y_1)^{n+1}, Z^n$

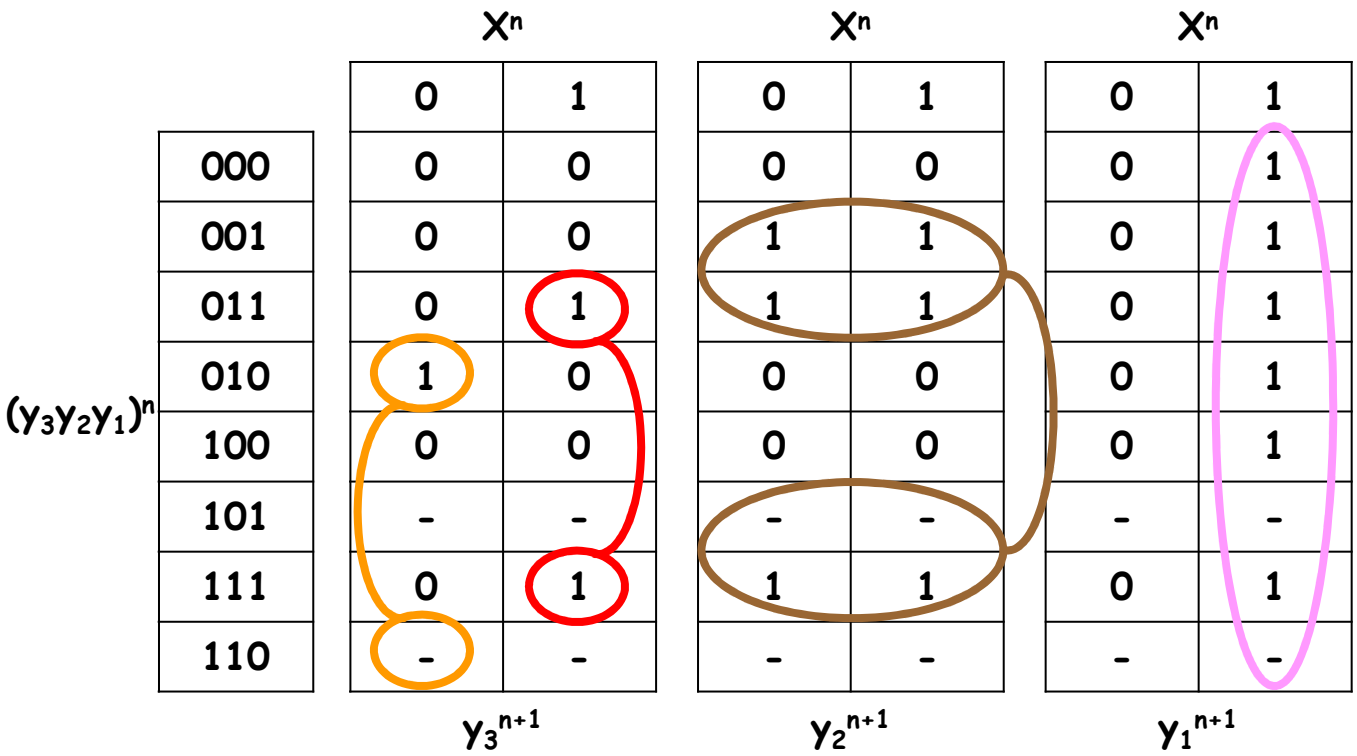
Mappa di codifica

y_3 \ y_2y_1	00	01	11	10
0	0	1	3	2
1	4	-	7	-

Rete combinatoria di uscita

$$Z^n = (X'y_1'y_2'y_3' + X'y_1y_2y_3' + Xy_3)^n$$

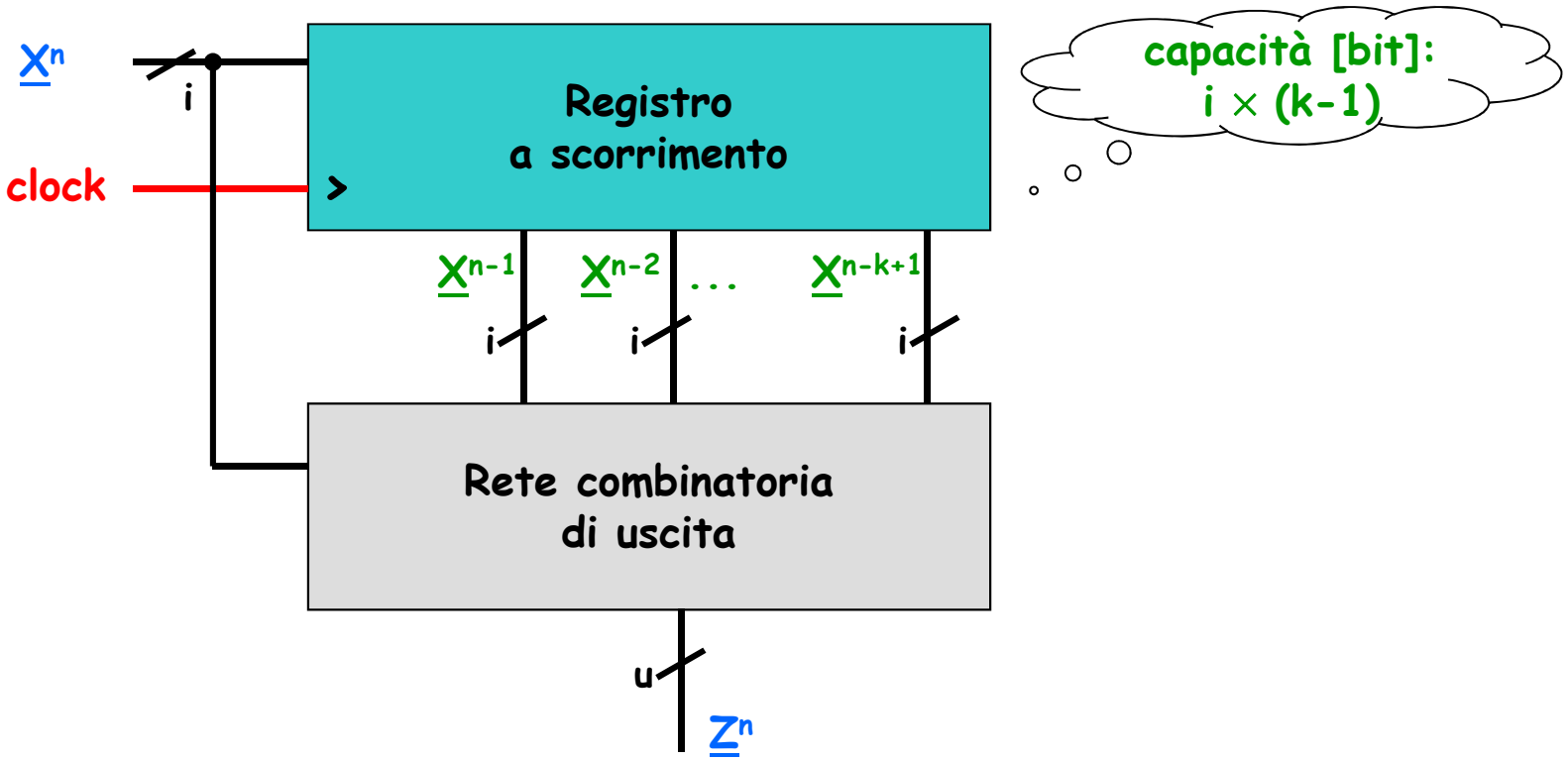
Rete combinatoria di aggiornamento dello stato



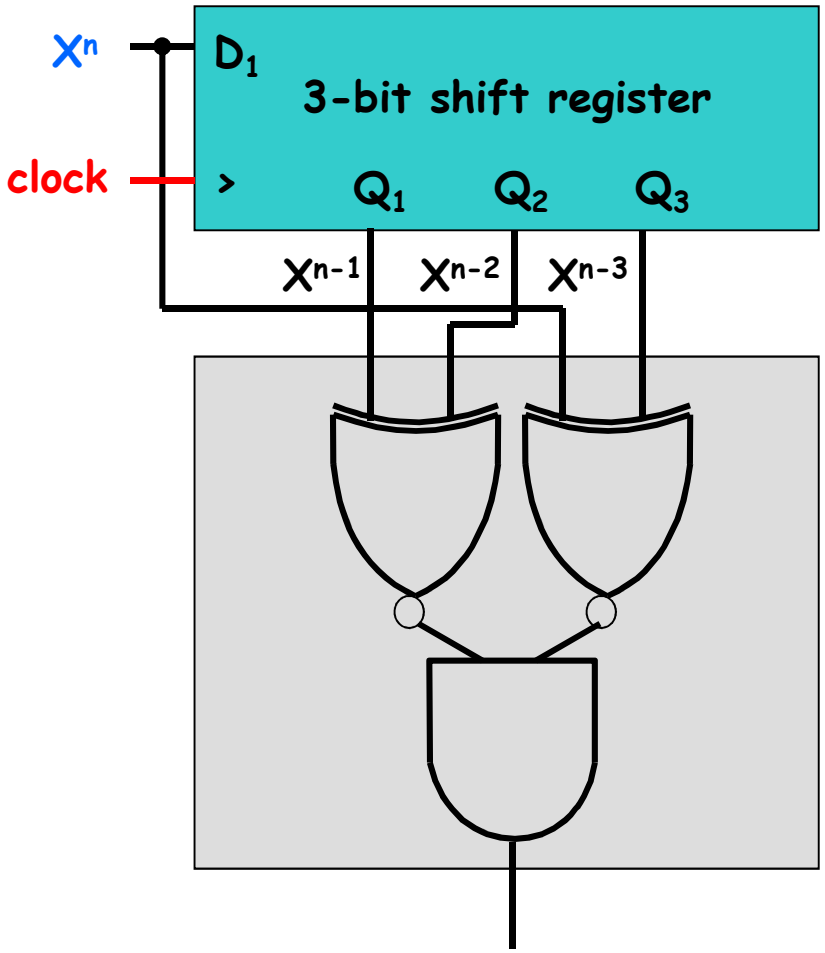
$D_3^n = y_3^{n+1} = (X'y_1'y_2 + Xy_1y_2)^n$ $D_2^n = y_2^{n+1} = y_1^n$ $D_1^n = y_1^{n+1} = X^n$

Quando i segnali di uscita $\underline{Z} = \{Z_1, Z_2, \ldots, Z_u\}$ di una rete sequenziale sincrona dipendono in ogni intervallo di clock direttamente dalle ultime k configurazioni dei segnali di ingresso $\underline{X} = \{X_1, X_2, \ldots, X_i\}$, è possibile eseguire il progetto in maniera rapida, strutturata e flessibile applicando il seguente

1° modello di riferimento per macchine con ampiezza di memoria finita



Applicazione del modello al problema in esame (i=u=1, k=4):

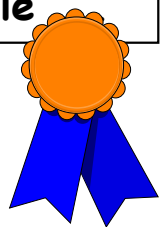


Unità di memoria

Unità di elaborazione

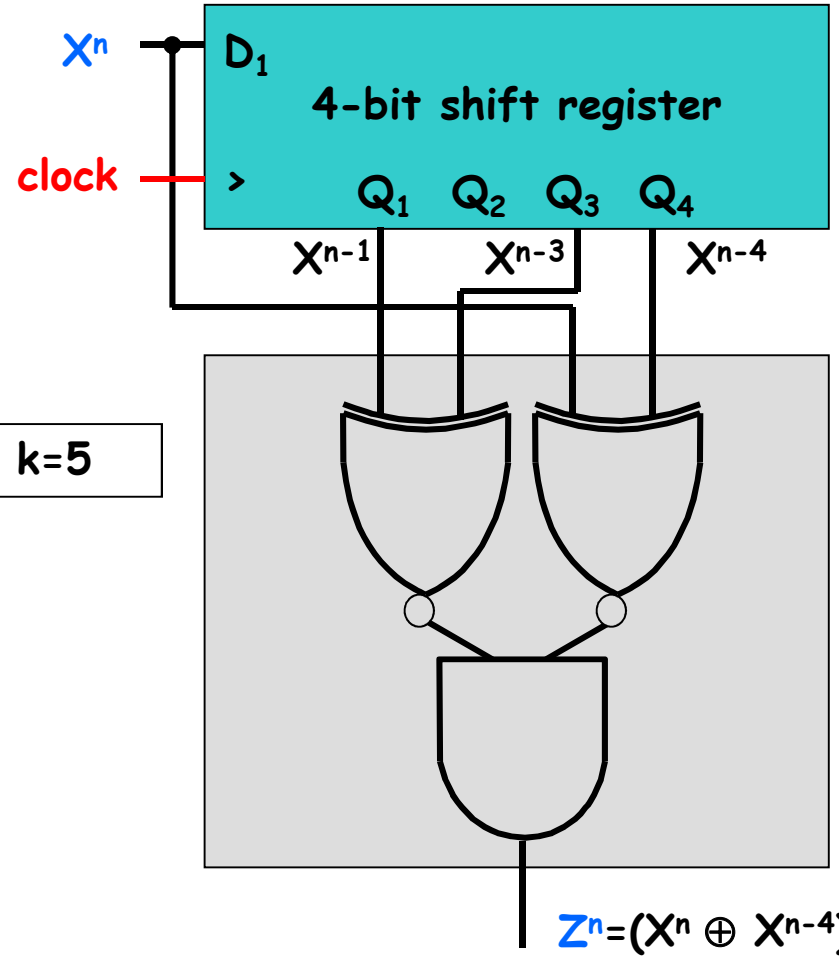
$$Z^n=(X^n \oplus X^{n-3})' \cdot (X^{n-1} \oplus X^{n-2})'$$

Rispetto alla realizzazione precedentemente identificata seguendo l'approccio convenzionale, questa soluzione è non solo più immediata ed economica, ma anche più flessibile



La flessibilità del modello

Se infatti la specifica di progetto richiedesse di attivare Z se non più gli ultimi 4, bensì gli ultimi k ($\forall$ k) valori di X costituiscono una palindrome, basterebbe ricorrere ad un registro a scorrimento di k-1 bit ed adattare coerentemente il comparatore in uscita:



Unità di memoria

Esempio: k=5

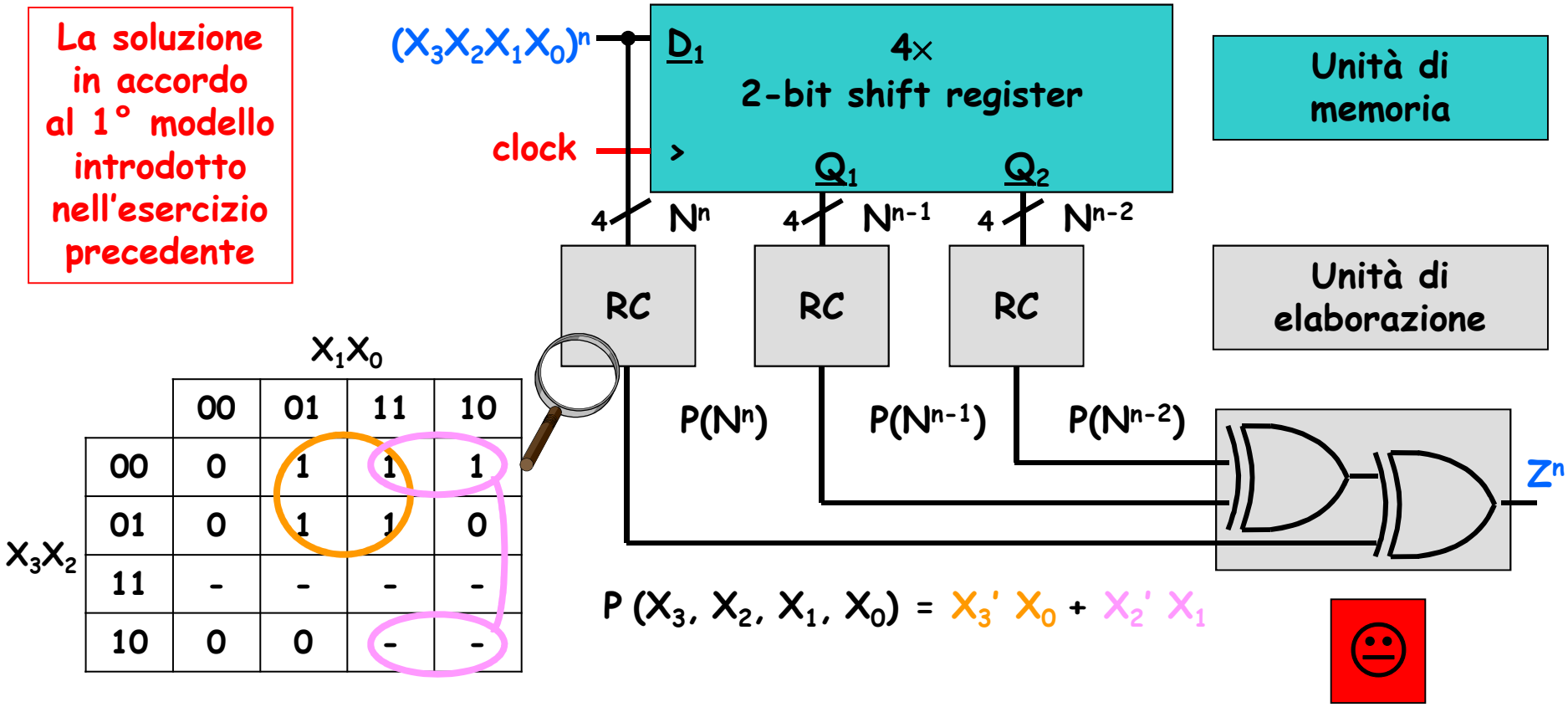
Unità di elaborazione

$Z^n=(X^n \oplus X^{n-4})' \cdot (X^{n-1} \oplus X^{n-3})'$

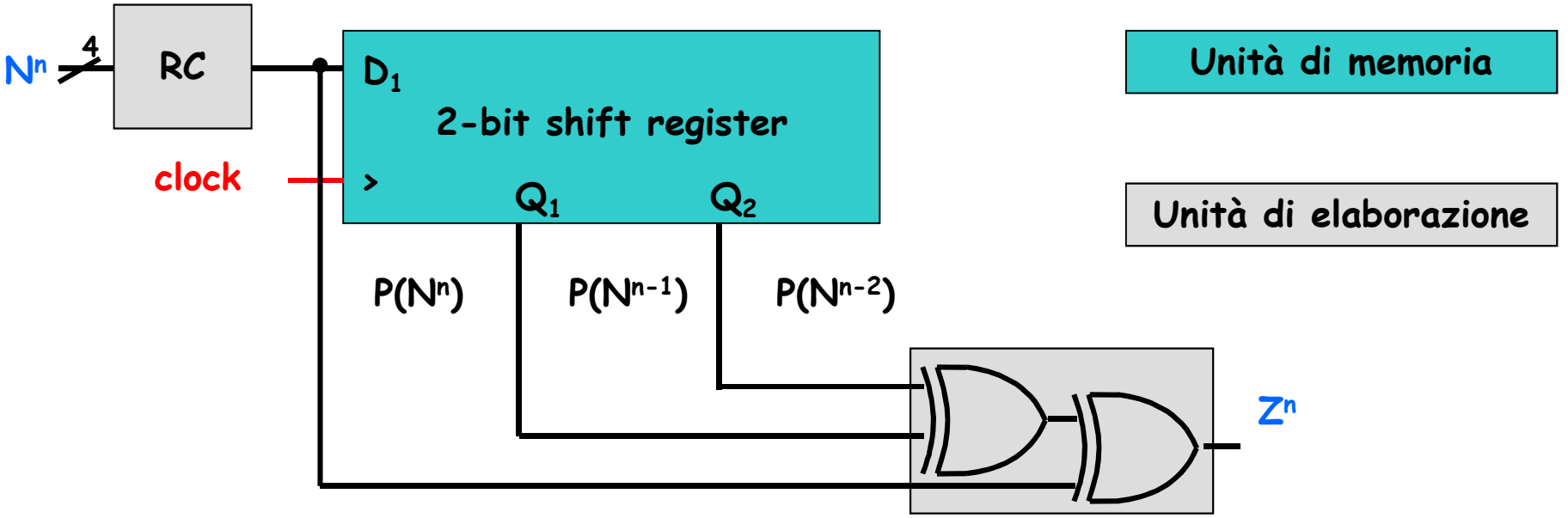
Esercizio 2

Una rete sequenziale sincrona è caratterizzata da quattro segnali di ingresso (X_3, X_2, X_1, X_0) e da un segnale di uscita (Z), tutti sincroni. In ogni intervallo di clock la rete riceve in ingresso una cifra decimale codificata in BCD (X_3 rappresenta il bit più significativo, X_0 il bit meno significativo). L'uscita della rete nel generico n-esimo intervallo di clock (Z^n) dipende dalle ultime tre cifre applicate in ingresso (N^n, N^{n-1}, N^{n-2}). In particolare Z^n=1 se una soltanto o tutte e tre le cifre sono prime. In caso contrario Z^n=0.

La soluzione in accordo al 1° modello introdotto nell'esercizio precedente



In realtà, ai fini della generazione del segnale di uscita Z, non è necessario memorizzare le cifre applicate in ingresso e quindi procedere alla loro elaborazione. Si può più convenientemente preelaborare ciascuna cifra, verificando se soddisfa o meno la proprietà richiesta (ovvero se è prima), e memorizzare solo il risultato di tale verifica:

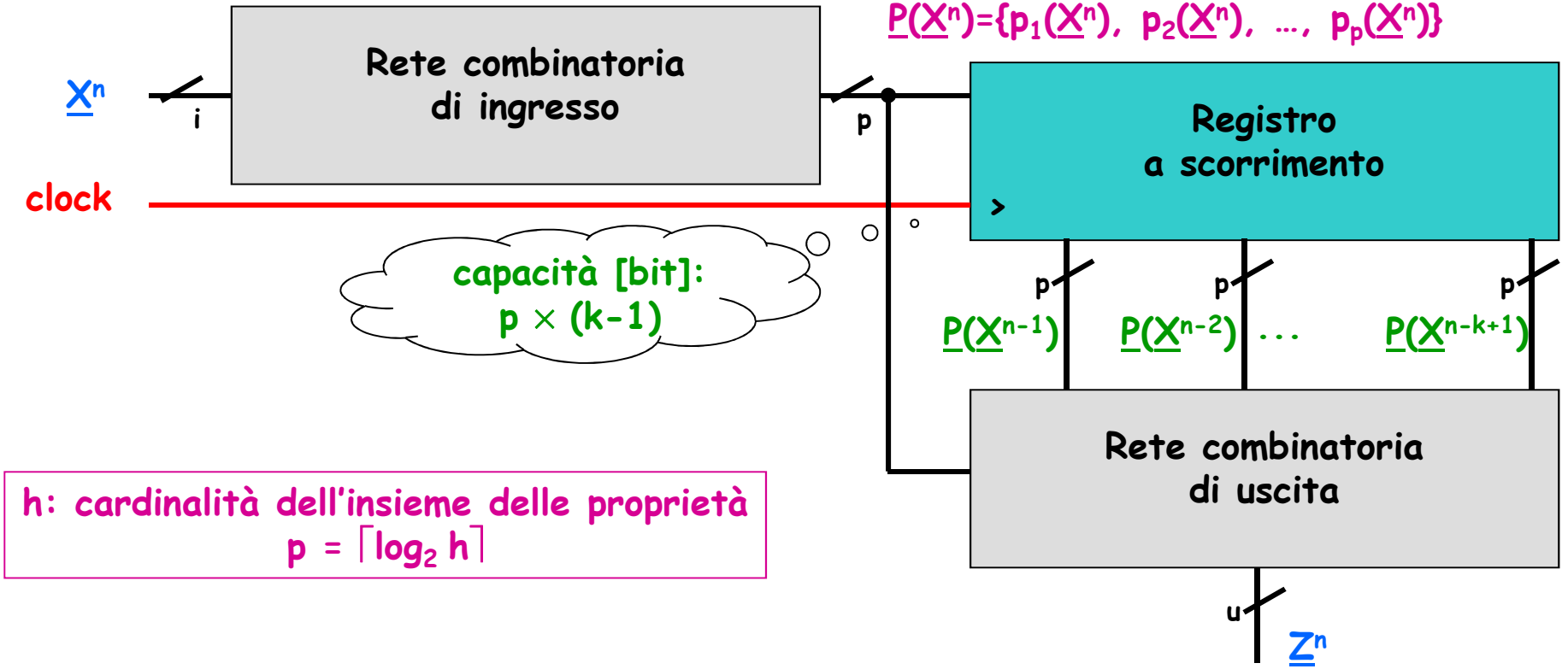


La capacità di memoria ora richiesta è pari al 25%.
La rete combinatoria RC non è più inutilmente replicata.

La soluzione è immediatamente configurabile per qualunque valore di k (basta ricorrere ad un registro a scorrimento di k-1 bit ed adattare coerentemente il solo generatore di parità).

Quando i segnali di uscita $\underline{Z}=\{Z_1, Z_2, \ldots, Z_u\}$ di una rete sequenziale sincrona dipendono in ogni intervallo di clock da una qualche proprietà associata alle ultime k configurazioni dei segnali di ingresso $\underline{X}=\{X_1, X_2, \ldots, X_i\}$, è possibile eseguire il progetto in maniera rapida, strutturata e flessibile applicando il seguente

2° modello di riferimento per macchine con ampiezza di memoria finita

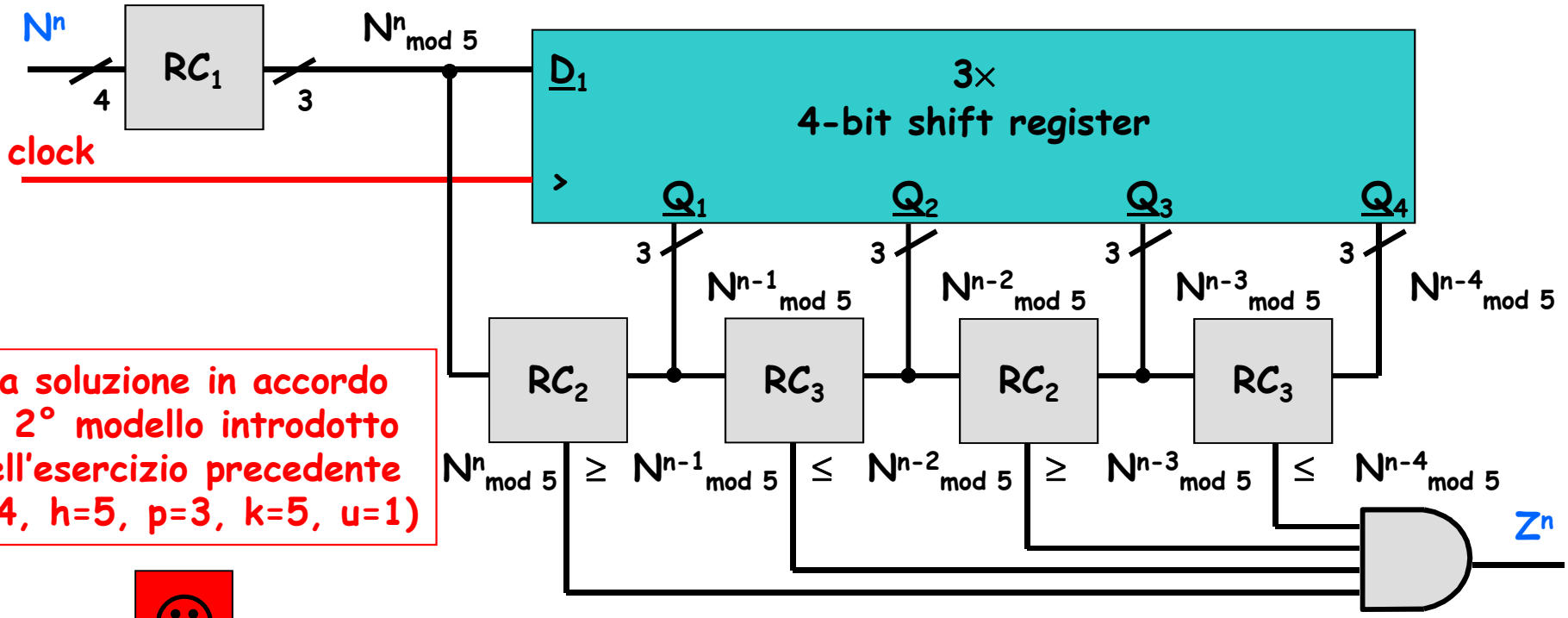


h: cardinalità dell'insieme delle proprietà

$p = \lceil \log_2 h \rceil$

Esercizio 3

Una rete sequenziale sincrona è caratterizzata da quattro segnali di ingresso (X_3, X_2, X_1, X_0) e da un segnale di uscita (Z), tutti sincroni. In ogni intervallo di clock la rete riceve in ingresso una cifra decimale codificata in BCD (X_3 rappresenta il bit più significativo, X_0 il bit meno significativo). L'uscita della rete nel generico n-esimo intervallo di clock (Z^n) dipende dalle ultime cinque cifre applicate in ingresso (N^n, N^{n-1}, N^{n-2}, N^{n-3}, N^{n-4}). In particolare Z^n=1 se e soltanto se $N^{n-j+1}{}_{mod\ 5} \geq N^{n-j}{}_{mod\ 5}$, j = 1, 3, $N^{n-j+1}{}_{mod\ 5} \leq N^{n-j}{}_{mod\ 5}$, j = 2, 4 (ovvero se $N^{n}{}_{mod\ 5} \geq N^{n-1}{}_{mod\ 5} \leq N^{n-2}{}_{mod\ 5} \geq N^{n-3}{}_{mod\ 5} \leq N^{n-4}{}_{mod\ 5}$).



La soluzione in accordo al 2° modello introdotto nell'esercizio precedente (i=4, h=5, p=3, k=5, u=1)

$(X_3\ X_2\ X_1\ X_0)^n$ — 4 — RC_1 — 3 — $(p_2\ p_1\ p_0)^n$

$(p_2\ p_1\ p_0)^{n-j+1}$ — 3 —
$(p_2\ p_1\ p_0)^{n-j}$ — 3 — RC_2 — $(r_j)^n = ((p_2\ p_1\ p_0)^{n-j+1} \geq (p_2\ p_1\ p_0)^{n-j})$

j dispari (1, 3)

$(p_2\ p_1\ p_0)^{n-j+1}$ — 3 —
$(p_2\ p_1\ p_0)^{n-j}$ — 3 — RC_3 — $(r_j)^n = ((p_2\ p_1\ p_0)^{n-j+1} \leq (p_2\ p_1\ p_0)^{n-j})$

j pari (2, 4)

RC_1

$(X_1X_0)^n$

$(X_3X_2)^n$

	00	01	11	10
00	000	001	011	010
01	100	000	010	001
11	-	-	-	-
10	011	100	-	-

$(p_2p_1p_0)^n$

$p_2^n = (X_2\ X_1'\ X_0' + X_3\ X_0)^n$

$p_1^n = (X_1\ X_0 + X_2'\ X_1 + X_3\ X_0')^n$

$p_0^n = (X_3'\ X_2'\ X_0 + X_2\ X_1\ X_0' + X_3\ X_0')^n$

RC_2

$(p_2p_1p_0)^{n-j}$

$(p_2p_1p_0)^{n-j+1}$	000	001	011	010	100	101	111	110
000	1	0	0	0	0	-	-	-
001	1	1	0	0	0	-	-	-
011	1	1	1	1	0	-	-	-
010	1	1	0	1	0	-	-	-
100	1	1	1	1	1	-	-	-
101	-	-	-	-	-	-	-	-
111	-	-	-	-	-	-	-	-
110	-	-	-	-	-	-	-	-

$r_j^n = p_2^{n-j+1} + (p_2')^{n-j}\,(\ldots)$

$r_j^n = (p_2p_1p_0)^{n-j+1} \geq (p_2p_1p_0)^{n-j}$

	00	01	11	10
00	1	0	0	0
01	1	1	0	0
11	1	1	1	1
10	1	1	0	1

$(\ldots) = (p_1\ p_0)^{n-j+1}$

$+ (p_1)^{n-j+1}\,(p_1')^{n-j}$

$+ (p_1)^{n-j+1}\,(p_0')^{n-j}$

$+ (p_0)^{n-j+1}\,(p_1')^{n-j}$

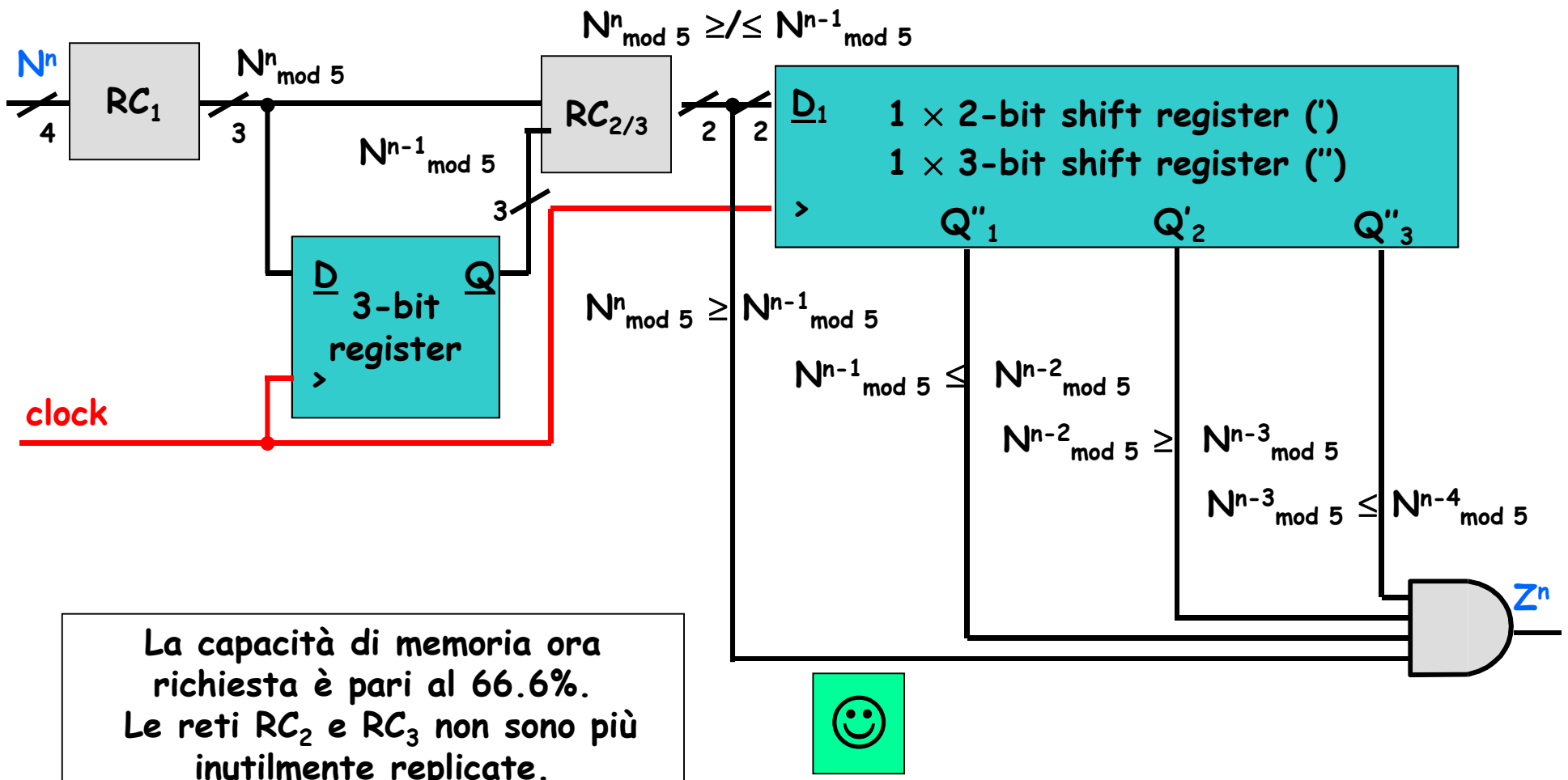
$+ (p_1'\ p_0')^{n-j}$

$(p_2\ p_1\ p_0)^{n-j+1}$ (3) $(p_2\ p_1\ p_0)^{n-j}$ (3) — RC_3 — $\leq$

$\equiv$

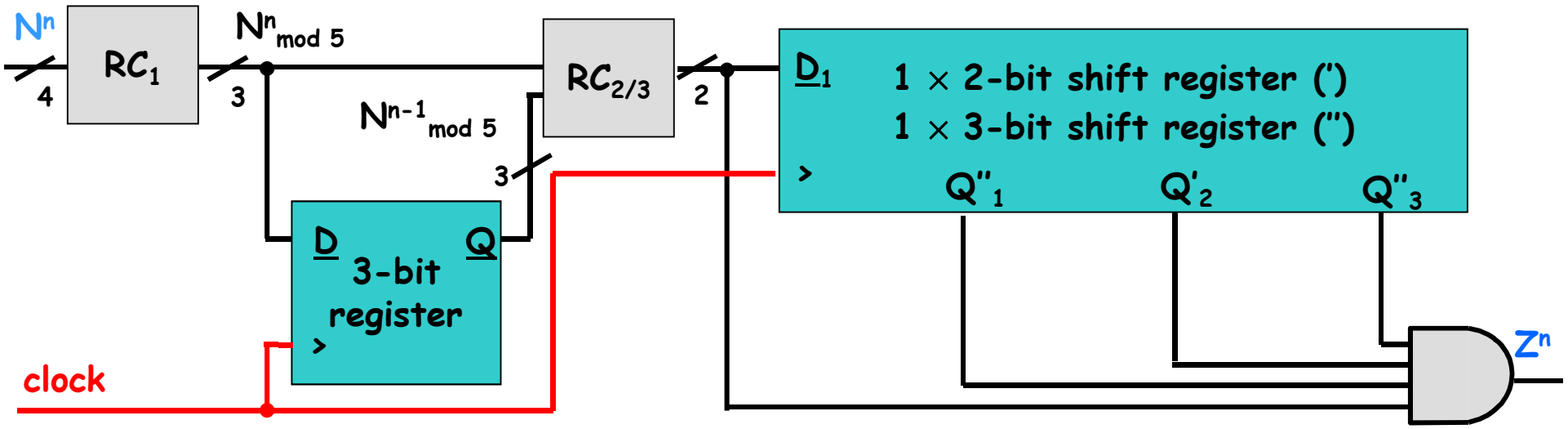
$(p_2\ p_1\ p_0)^{n-j}$ (3) $(p_2\ p_1\ p_0)^{n-j+1}$ (3) — RC_2 — $\geq$

Ai fini della generazione del segnale di uscita non è necessario memorizzare via via i bit rappresentativi della proprietà (il resto della divisione per cinque) associata alle 4 cifre precedentemente applicate in ingresso e quindi procedere alle comparazioni.
Si può più convenientemente memorizzare i bit relativi soltanto all'ultima cifra, confrontarli con quelli associati alla cifra attualmente in ingresso, e generare Z in base all'esito di tale confronto, unitamente all'esito dei 3 precedenti analoghi confronti:



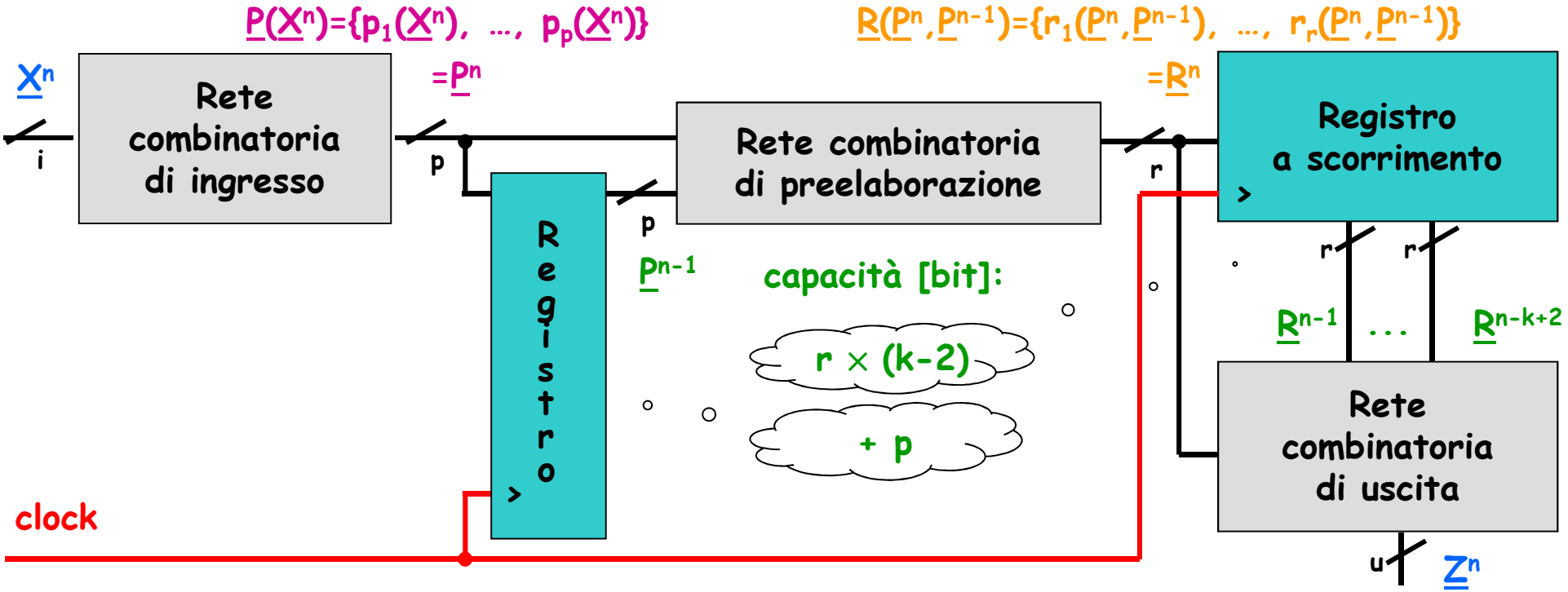
La capacità di memoria ora richiesta è pari al 66.6%.
Le reti RC_2 e RC_3 non sono più inutilmente replicate.

N^n	...	1	7	1	5	0	5	0	2	2	0	0	9	?
$N^n_{\text{mod } 5}$	D^n	...	1	2	1	0	0	0	0	2	2	0	0	4	?
$N^{n-1}_{\text{mod } 5}$	Q^n	1	2	1	0	0	0	0	2	2	0	0	4	?	?
$N^n_{\text{mod } 5} \geq N^{n-1}_{\text{mod } 5}$	$D'_1{}^n$	...	0	1	1	1	1	1	0	1	1	1	0	?	?
$N^{n-1}_{\text{mod } 5} \geq N^{n-2}_{\text{mod } 5}$	$Q'_1{}^n$	0	1	1	1	1	1	0	1	1	1	0	?	?	?
$N^{n-2}_{\text{mod } 5} \geq N^{n-3}_{\text{mod } 5}$	$Q'_2{}^n$	1	1	1	1	1	0	1	1	1	0	?	?	?	?
$N^n_{\text{mod } 5} \leq N^{n-1}_{\text{mod } 5}$	$D''_1{}^n$	...	1	0	0	1	1	1	1	1	0	1	1	?	?
$N^{n-1}_{\text{mod } 5} \leq N^{n-2}_{\text{mod } 5}$	$Q''_1{}^n$	1	0	0	1	1	1	1	1	0	1	1	?	?	?
$N^{n-2}_{\text{mod } 5} \leq N^{n-3}_{\text{mod } 5}$	$Q''_2{}^n$	0	0	1	1	1	1	1	0	1	1	?	?	?	?
$N^{n-3}_{\text{mod } 5} \leq N^{n-4}_{\text{mod } 5}$	$Q''_3{}^n$	0	1	1	1	1	1	0	1	1	?	?	?	?	?
$D'_1{}^n$ $Q''_1{}^n$ $Q'_2{}^n$ $Q''_3{}^n$	Z^n	0	0	0	1	1	0	0	0	0	0	?	0	?	?



Quando i segnali di uscita $\underline{Z} = \{Z_1, Z_2, \ldots, Z_u\}$ di una rete sequenziale sincrona dipendono in ogni intervallo di clock dalle (o da qualche proprietà delle) ultime k configurazioni dei segnali di ingresso $\underline{X} = \{X_1, X_2, \ldots, X_i\}$ secondo relazioni che coinvolgono iterativamente ciascuna delle k-1 coppie di (proprietà di) configurazioni consecutive, è possibile eseguire il progetto in maniera rapida, strutturata e flessibile applicando il seguente

3° modello di riferimento per macchine con ampiezza di memoria finita

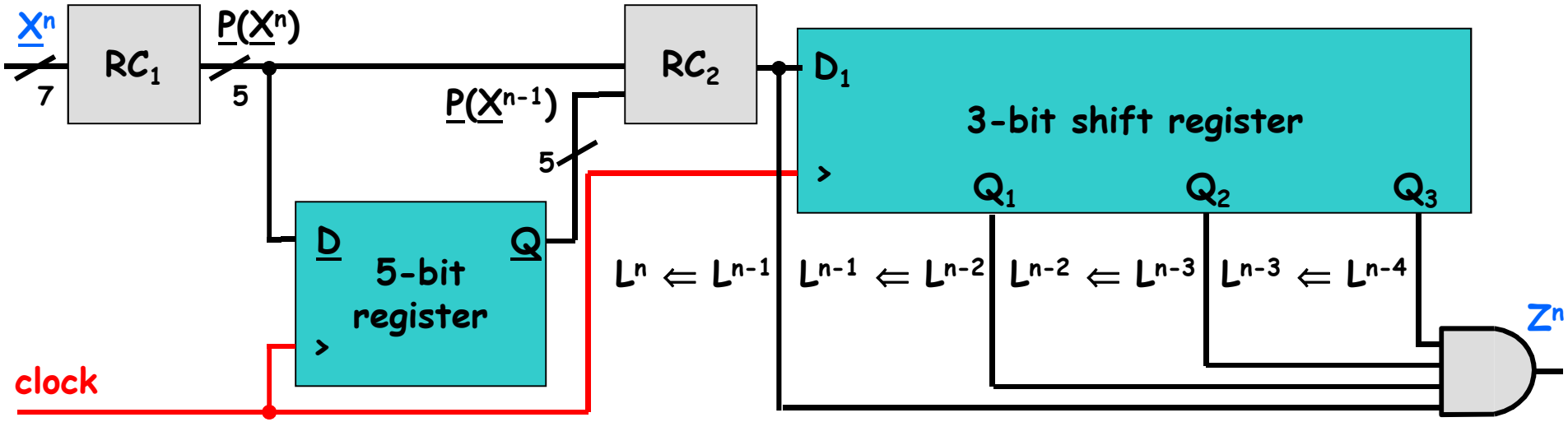


Esercizio 4

Una rete sequenziale sincrona è caratterizzata da sette segnali di ingresso (X_6, X_5 , X_4, X_3, X_2, X_1, X_0) e da un segnale di uscita (Z), tutti sincroni.

In ogni intervallo di clock la rete riceve in ingresso un simbolo alfanumerico (0, 1, …, 9, A, B, …, Z, a, b, …, z) rappresentato secondo il codice ASCII (X_6 rappresenta il bit più significativo, X_0 il bit meno significativo).

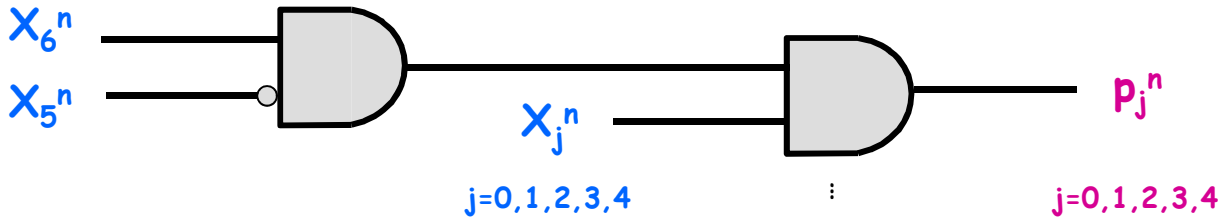
L'uscita della rete nel generico n-esimo intervallo di clock (Z^n) dipende dagli ultimi 5 simboli applicati in ingresso ($\underline{X}^n$, $\underline{X}^{n-1}$, $\underline{X}^{n-2}$, $\underline{X}^{n-3}$, $\underline{X}^{n-4}$). In particolare Z^n deve valere 1 solo se tali simboli corrispondono a lettere maiuscole, diverse fra loro, ed in ordine alfabetico.



La soluzione in accordo al 3° modello introdotto nell'esercizio precedente (i=7, h=27, p=5, r=1, k=5, u=1)

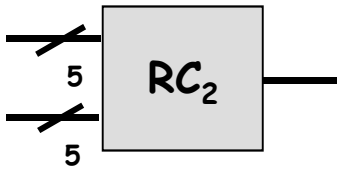
$(X_6\ X_5\ X_4\ X_3\ X_2\ X_1\ X_0)^n$ → 7 → **RC1** → 5 → $(p_4\ p_3\ p_2\ p_1\ p_0)^n$

	X_6	X_5	X_4	X_3	X_2	X_1	X_0	p_4	p_3	p_2	p_1	p_0
0	0	1	1	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	1	0	0	0	0	0
…	0	1	1	…	…	…	…	0	0	0	0	0
9	0	1	1	1	0	0	1	0	0	0	0	0
A	1	0	0	0	0	0	1	0	0	0	0	1
B	1	0	0	0	0	1	0	0	0	0	1	0
…	1	0	…	…	…	…	…	…	…	…	…	…
Z	1	0	1	1	0	1	0	1	1	0	1	0
a	1	1	0	0	0	0	1	0	0	0	0	0
b	1	1	0	0	0	1	0	0	0	0	0	0
…	1	1	…	…	…	…	…	0	0	0	0	0
z	1	1	1	1	0	1	0	0	0	0	0	0



$\underline{P}^n \equiv (p_4\ p_3\ p_2\ p_1\ p_0)^n$

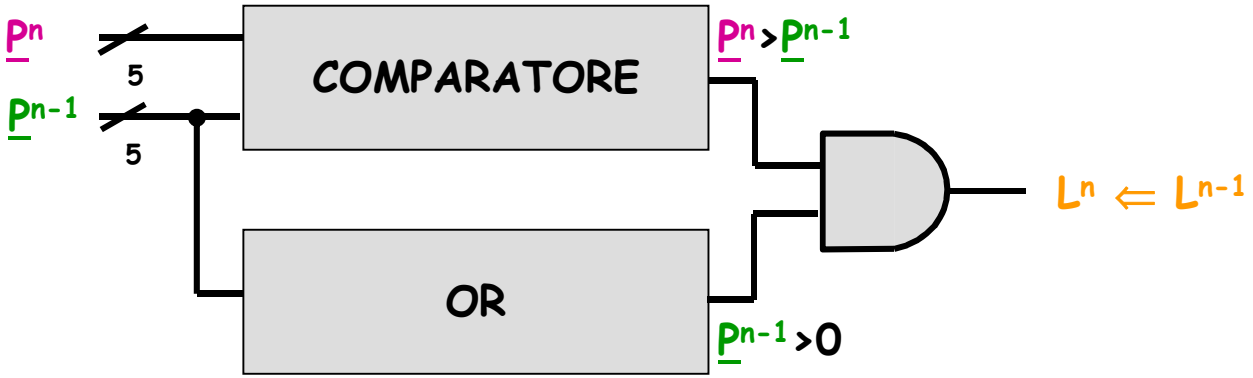
$\underline{P}^{n-1} \equiv (p_4\ p_3\ p_2\ p_1\ p_0)^{n-1}$

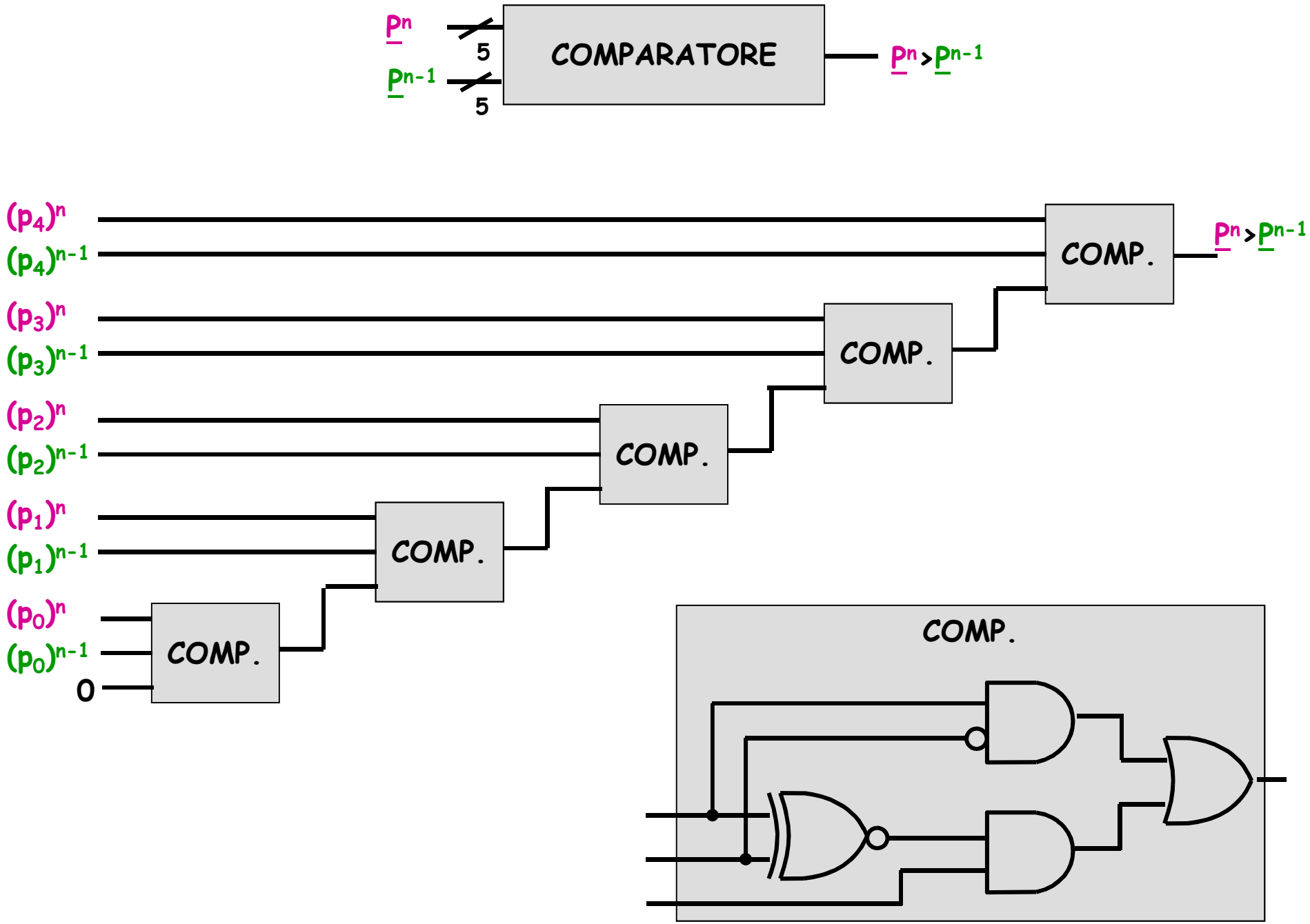


$L^n \Leftarrow L^{n-1}$ (coppia di lettere maiuscole in ordine alfabetico)

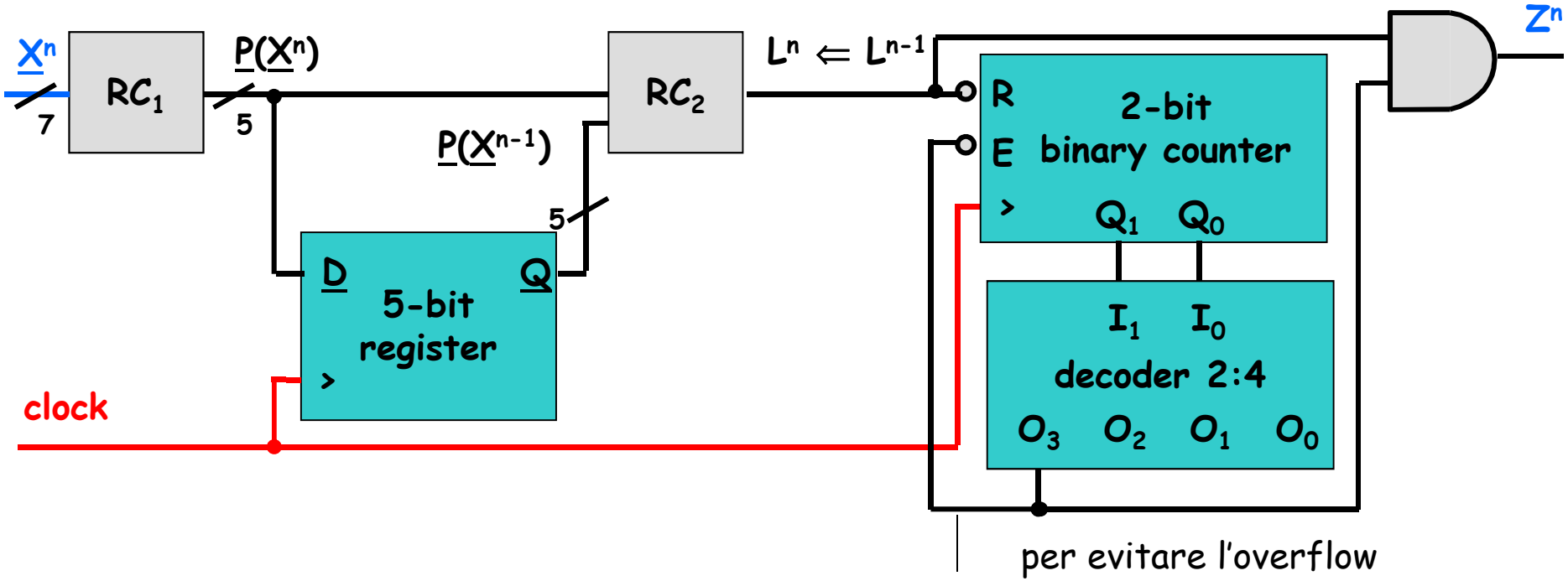
se ($\underline{P}^n > 0$) & ($\underline{P}^{n-1} > 0$) & ($\underline{P}^n > \underline{P}^{n-1}$), ovvero se ($\underline{P}^{n-1} > 0$) & ($\underline{P}^n > \underline{P}^{n-1}$)

$$L^n \Leftarrow L^{n-1} = \{(p_4+p_3+p_2+p_1+p_0)^{n-1}\}$$
$$\{(p_4)^n\,(p_4')^{n-1} + ((p_4)^n \oplus (p_4)^{n-1})'$$
$$[(p_3)^n\,(p_3')^{n-1} + ((p_3)^n \oplus (p_3)^{n-1})'$$
$$[(p_2)^n\,(p_2')^{n-1} + ((p_2)^n \oplus (p_2)^{n-1})'$$
$$[(p_1)^n\,(p_1')^{n-1} + ((p_1)^n \oplus (p_1)^{n-1})'$$
$$[(p_0)^n\,(p_0')^{n-1}]]]]\}$$




Pn
Pn-1
5
5
COMPARATORE
Pn>Pn-1
(p4)n
(p4)n-1
(p3)n
(p3)n-1
(p2)n
(p2)n-1
(p1)n
(p1)n-1
(p0)n
(p0)n-1
0
COMP.
COMP.
COMP.
COMP.
COMP.
Pn>Pn-1
COMP.

Ai fini della generazione del segnale di uscita non è necessario memorizzare l'esito dei singoli confronti e combinarli poi attraverso un operatore AND. Basta contarli !!!

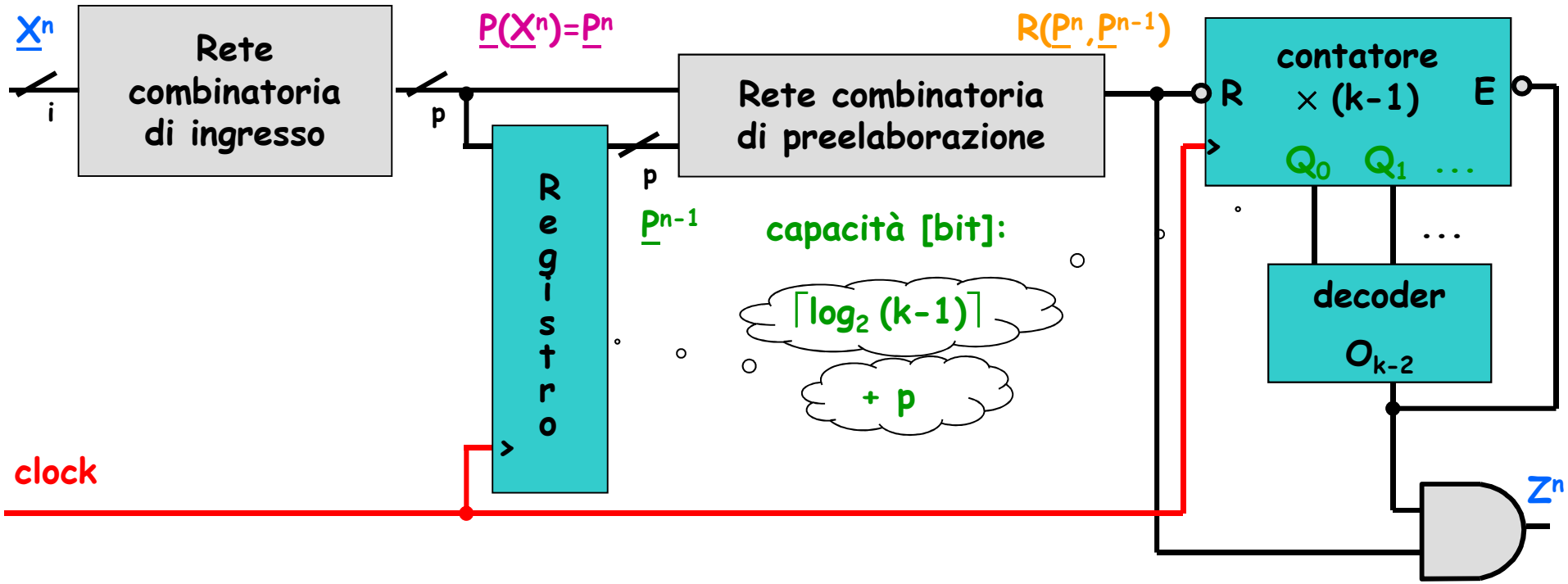


La capacità di memoria ora richiesta è pari all'87.5 %.

La soluzione è immediatamente configurabile per qualunque valore di k (basta modificare la base del contatore (k-1) ed eventualmente il decoder). Rispetto alla soluzione precedente, la capacità di memoria richiesta si riduce da $5 + (k-2)$ bit a $5 + \lceil \log_2 (k-1) \rceil$ bit.

Quando il segnale di uscita Z di una rete sequenziale sincrona dipende in ogni intervallo di clock dalle (o da qualche proprietà delle) ultime k configurazioni dei segnali di ingresso $\underline{X} = \{X_1, X_2, \ldots, X_i\}$ secondo una relazione che coinvolge iterativamente ed identicamente ciascuna delle k-1 coppie di (proprietà di) configurazioni consecutive, è possibile eseguire il progetto in maniera rapida, strutturata e flessibile applicando il seguente

4° modello di riferimento per macchine con ampiezza di memoria finita



Esercizio 5

Una rete sequenziale sincrona è caratterizzata da 2 segnali di ingresso (START, X) e da 2 segnali di uscita (END, Z), tutti sincroni. Attraverso l'ingresso X la rete riceve serialmente parole di 32 bit. Ogni parola, costituita da 8 cifre decimali rappresentate secondo il codice BCD, identifica una data in termini di anno ($A \equiv \{A_m\, A_c\, A_d\, A_u\}$), mese ($M \equiv \{M_d\, M_u\}$) e giorno ($G \equiv \{G_d\, G_u\}$). Le cifre identificative di ciascuna data sono ricevute secondo l'ordine G_d, G_u, M_d, M_u, A_m, A_c, A_d, A_u. I bit di ogni cifra sono ricevuti a partire da quello più significativo. L'intervallo di ricezione del 1° bit di ciascuna parola è identificato dal valore logico 1 del segnale START. La rete ha il compito di attivare il segnale di uscita Z ogni qual volta l'anno corrispondente ad una data è bisestile. Il valore di Z è da intendersi significativo soltanto in corrispondenza dell'intervallo di ricezione dell'ultimo bit di ciascuna data, evidenziato dalla rete stessa attraverso l'attivazione del segnale END.

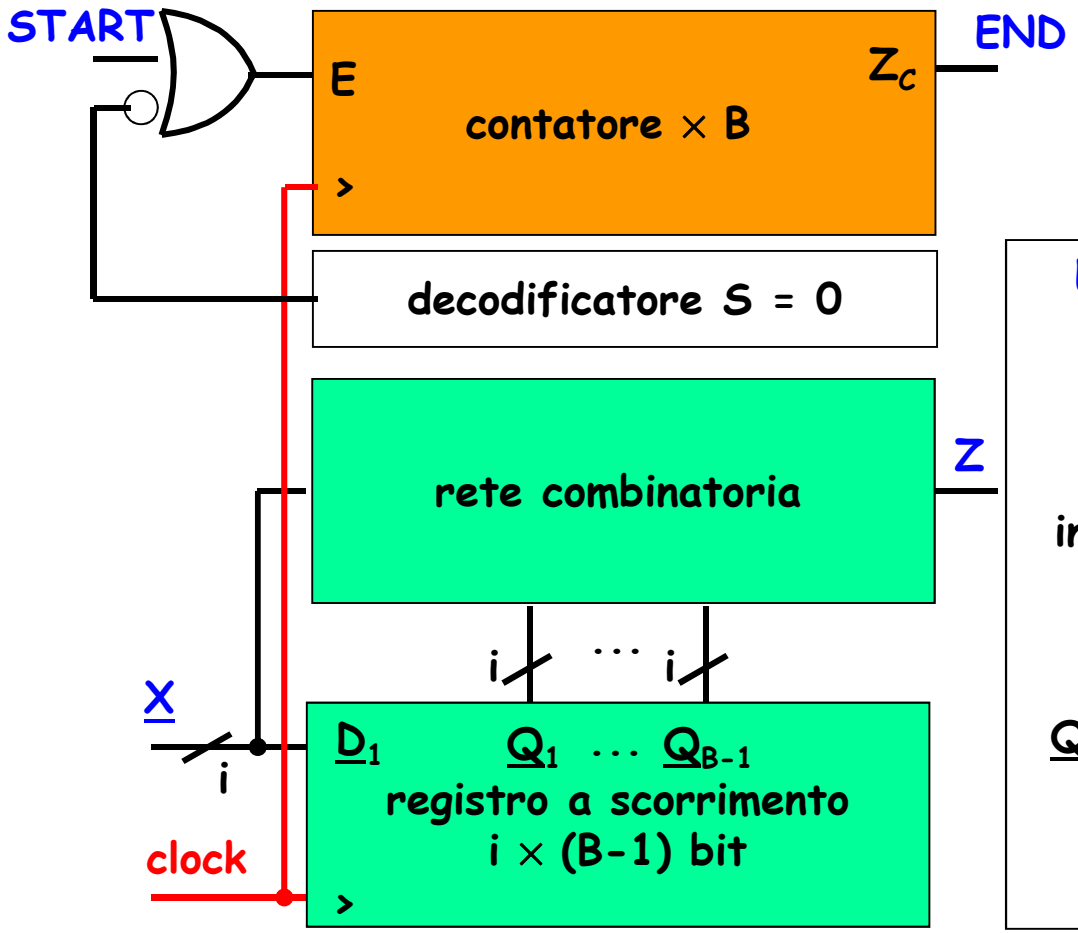
Un anno è bisestile se è divisibile per quattro,
a meno che sia il primo anno di un secolo non multiplo di quattro:

$$(A_{\text{mod } 4} = 0) \text{ and not } ((A_{\text{mod } 100} = 0) \text{ and } ((A/100)_{\text{mod } 4} \neq 0))$$

ovvero se:

$$((A_d A_u)_{\text{mod } 4} = 0) \text{ and } ((A_d \neq 0) \text{ or } (A_u \neq 0) \text{ or } ((A_m A_c)_{\text{mod } 4} = 0))$$

Il problema può essere risolto applicando il modello introdotto nel precedente esempio: una rete di sincronizzazione, basata su un contatore con base di conteggio B coincidente con il numero di simboli per parola, provvede ad identificare l'intervallo di ricezione di ciascun simbolo di ogni parola; una rete di elaborazione, strutturata come macchina con ampiezza di memoria finita, svolge il compito di gestire i singoli bit di ciascuna parola in accordo alle specifiche.



END^n vale 1 solo quando il simbolo X^n in ingresso è, per costruzione, l'ultimo di una parola. Corrispondentemente il registro a scorrimento rende disponibili in uscita ordinatamente i precedenti simboli di una parola (o tutti quelli significativi ai fini del calcolo di Z^n):

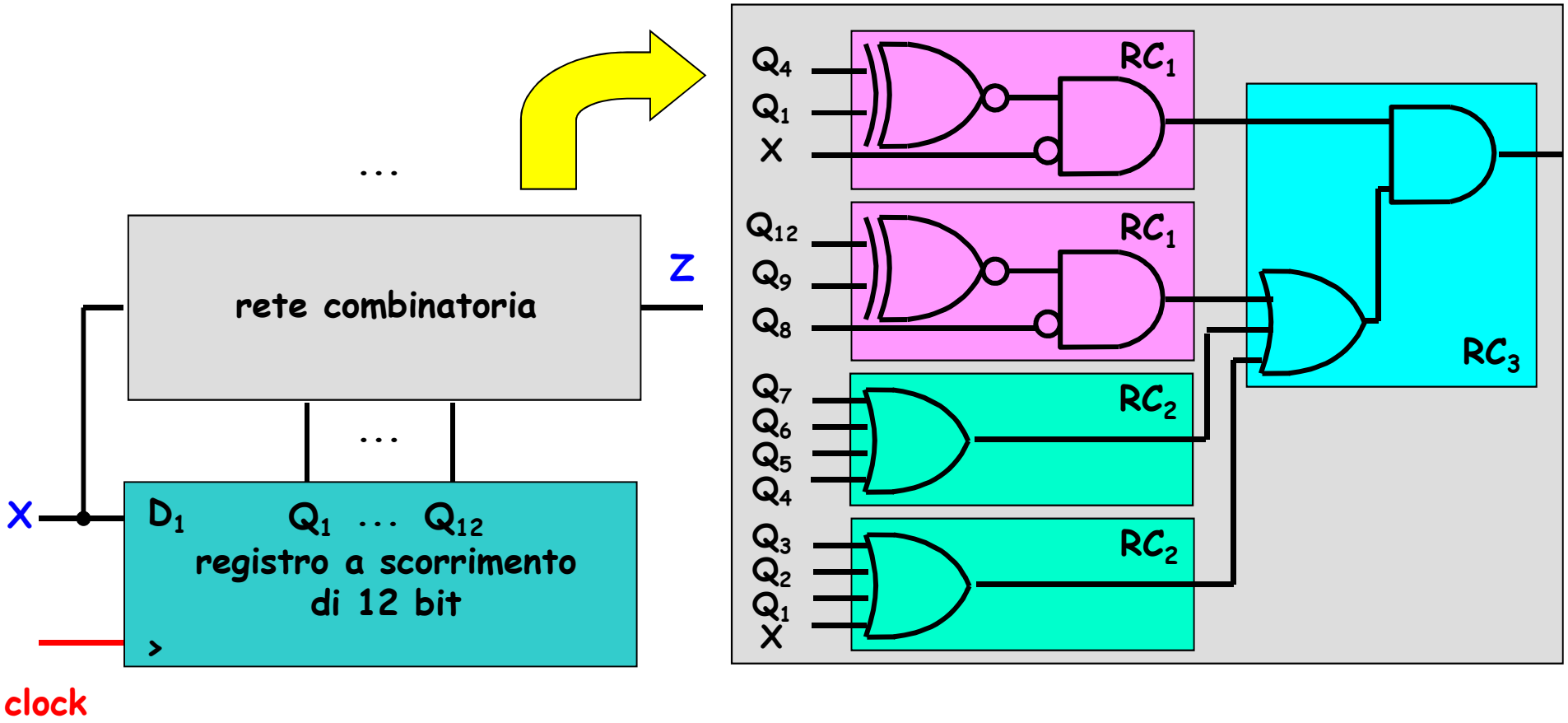
$Q_1^n = X^{n-1}$, $Q_2^n = X^{n-2}$, …, $Q_{B-1}^n = X^{n-B+1}$.

Il problema da risolvere è ora combinatorio !!!

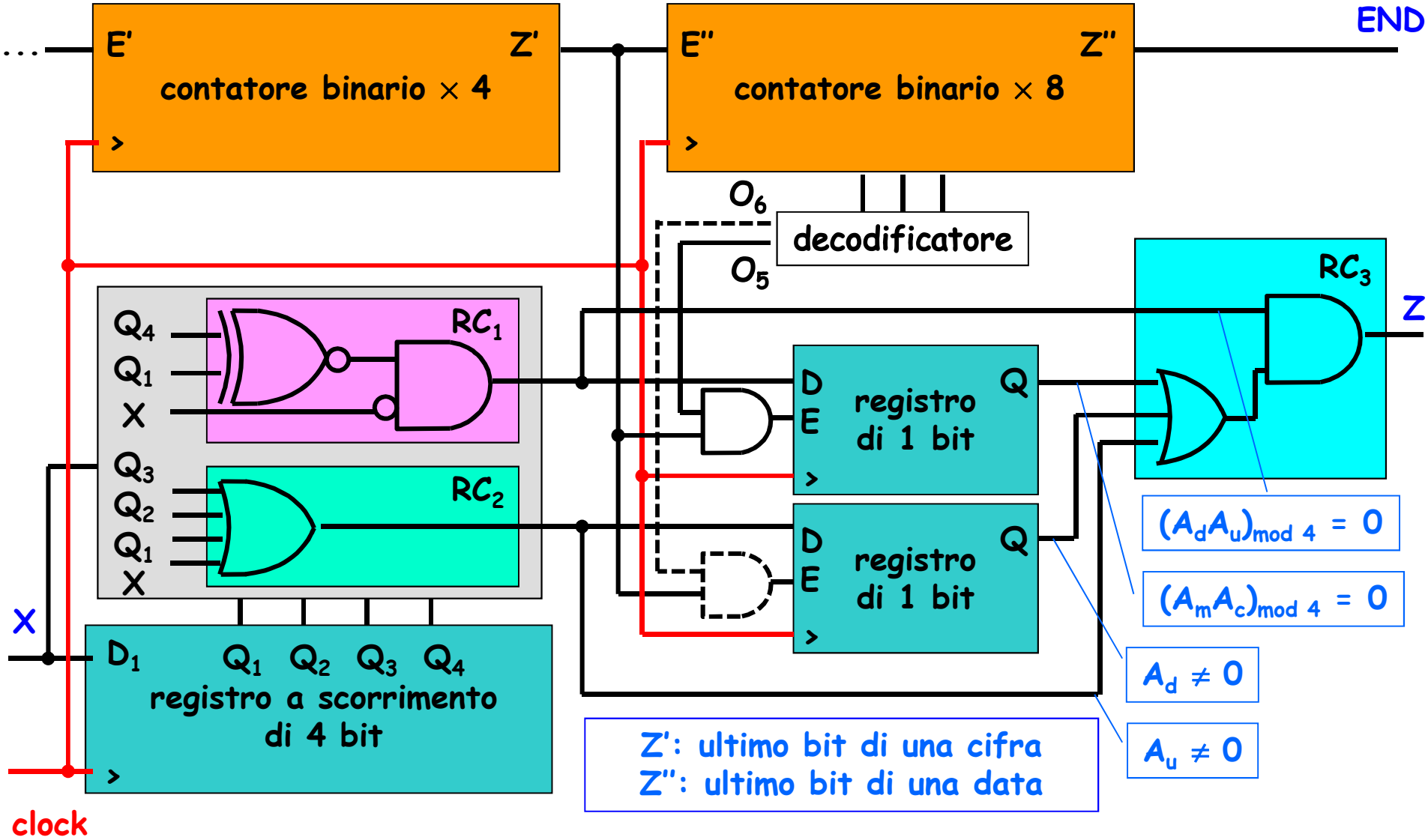
Nel caso specifico basta selezionare un contatore × 32 ed un registro a scorrimento di soli 12 bit, poiché né i quattro bit di G_d, G_u, M_d, M_u, né i tre bit più significativi di A_m intervengono nella relazione di ingresso-uscita della rete combinatoria:

$$Z_{RC} = A_{u0}' (A_{u1} \oplus A_{d0})' (A_{u0} + A_{u1} + A_{u2} + A_{u3} + A_{d0} + A_{d1} + A_{d2} + A_{d3} + A_{c0}' (A_{c1} \oplus A_{m0})')$$

(espressione non minima)



Facendo riferimento ai modelli già discussi a proposito delle macchine con ampiezza di memoria finita, è possibile identificare una soluzione più efficiente, che evita di replicare inutilmente le due reti combinatorie RC_1 e RC_2, e nel contempo permette di ridurre la capacità di memoria richiesta (50%).



Esercizio 6

Un sistema sequenziale sincrono è caratterizzato da sette segnali di ingresso (START, OP, M, T_1, T_0, X_i, Y_i) e da due segnali di uscita (END, Z), tutti sincroni.

Il sistema ha il compito di elaborare coppie di dati (X, Y) di n bit ($X \equiv \{X_{n-1}, \ldots, X_1, X_0\}$, $Y \equiv \{Y_{n-1}, \ldots, Y_1, Y_0\}$), applicati in ingresso serialmente tramite, rispettivamente, i segnali X_i e Y_i.

Il segnale M specifica, per ogni coppia di dati, la modalità di trasferimento in ingresso:

- M = 0 ⇒ "little endian" (least significant bit (X_0, Y_0) first),
- M = 1 ⇒ "big endian" (most significant bit (X_{n-1}, Y_{n-1}) first).

I segnali T_1 e T_0 identificano il tipo di dati presentati in ingresso:

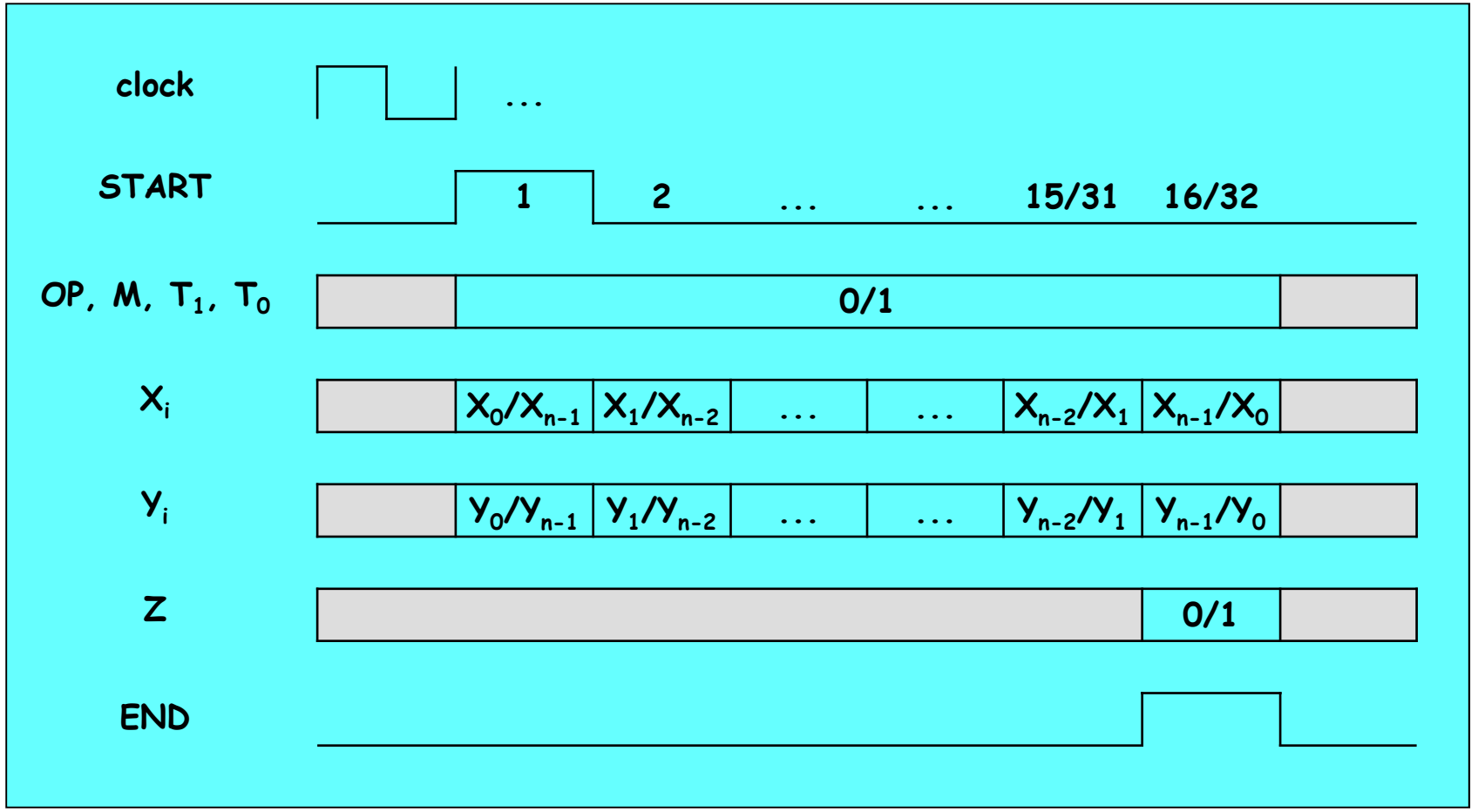
- $T_1 T_0$ = 00 ⇒ "unsigned short integer" (n = 16),
- $T_1 T_0$ = 01 ⇒ "signed short integer" (n = 16),
- $T_1 T_0$ = 10 ⇒ "unsigned integer" (n = 32),
- $T_1 T_0$ = 11 ⇒ "signed integer" (n = 32).

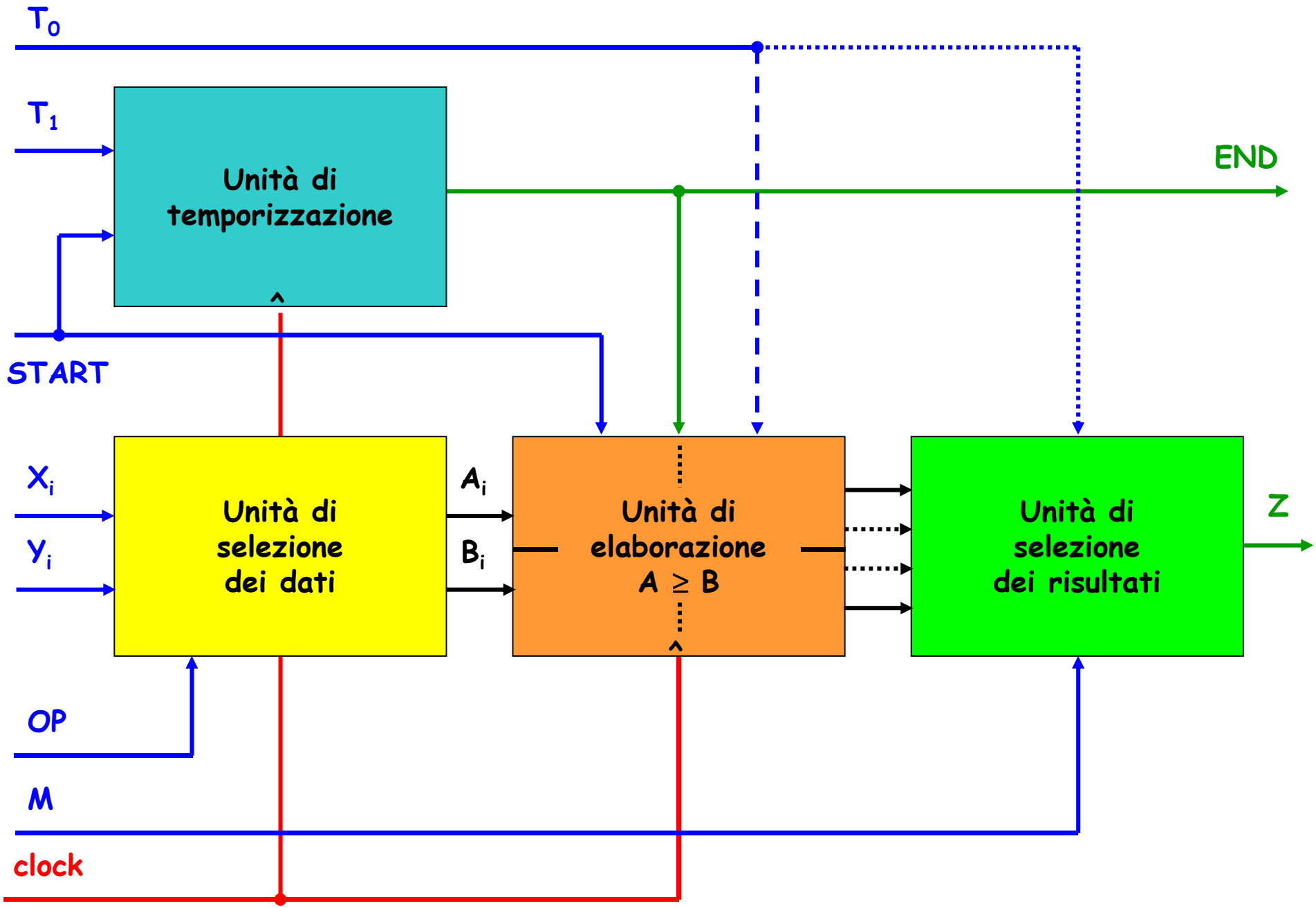
Il segnale OP identifica il tipo di elaborazione da svolgere:

- OP = 0 ⇒ $X \geq Y$,
- OP = 1 ⇒ $X \leq Y$.

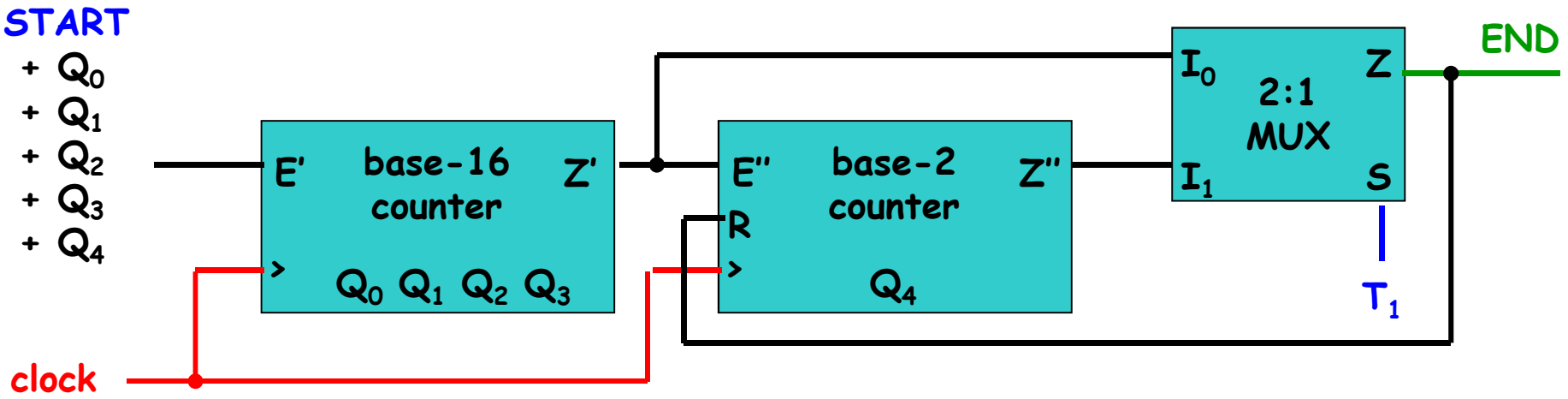
Il segnale START, attivo a livello logico 1 e di durata unitaria, identifica l'intervallo di ricezione del primo bit di ciascuna coppia di dati. Il risultato di ogni elaborazione deve essere evidenziato dal sistema tramite il segnale di uscita Z in corrispondenza dell'intervallo di ricezione dell'ultimo bit dei dati corrispondenti. In particolare Z deve valere 1 se il confronto richiesto ha dato esito positivo, 0 in caso contrario. Il sistema ha anche il compito di attivare il segnale END durante l'intervallo di generazione di ciascun risultato.

Nell'ipotesi che i segnali OP, M, T_1, T_0 non possano cambiare di valore durante il processo di elaborazione, si esegua la sintesi del sistema in accordo al modello indicato in figura, utilizzando i componenti ritenuti più idonei allo scopo e motivando esplicitamente tutte le scelte progettuali operate.

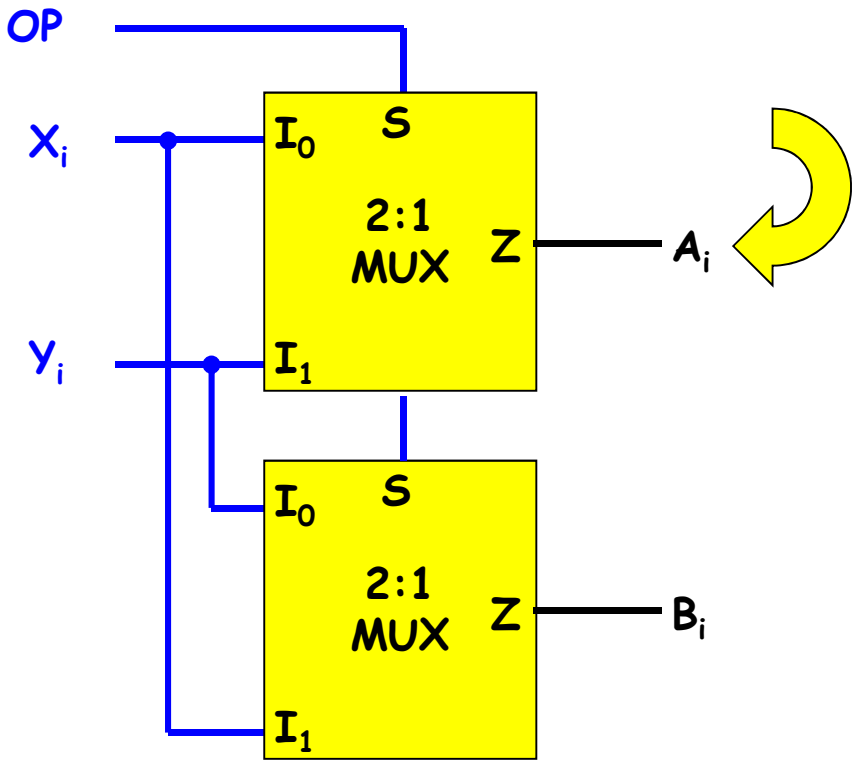


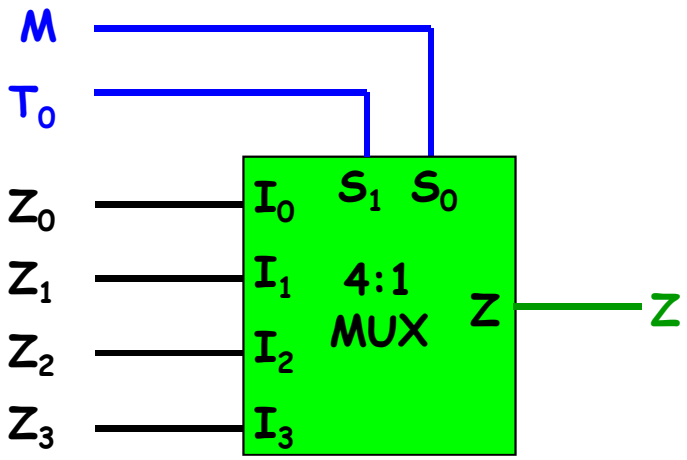

T0
T1
Unità di temporizzazione
END
START
Xi
Yi
Unità di selezione dei dati
Ai
Bi
Unità di elaborazione A ≥ B
Unità di selezione dei risultati
Z
OP
M
clock

L'unità di temporizzazione



Le unità di selezione dei dati e dei risultati





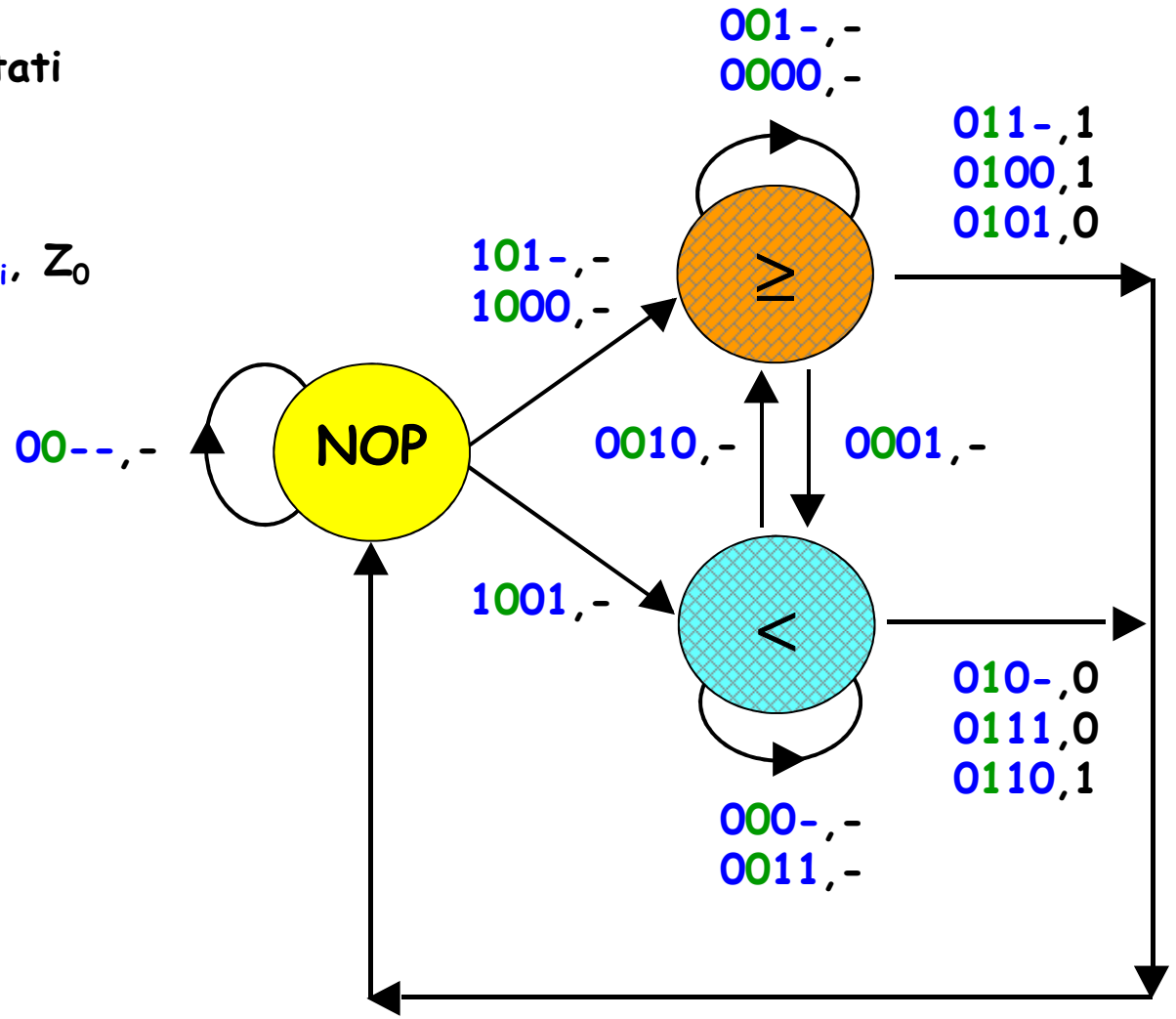
Progetto dell'unità di elaborazione

1° caso: little endian (M=0), unsigned integer (T_0=0)

Il grafo degli stati

Legenda:

START END A_i B_i, Z_0



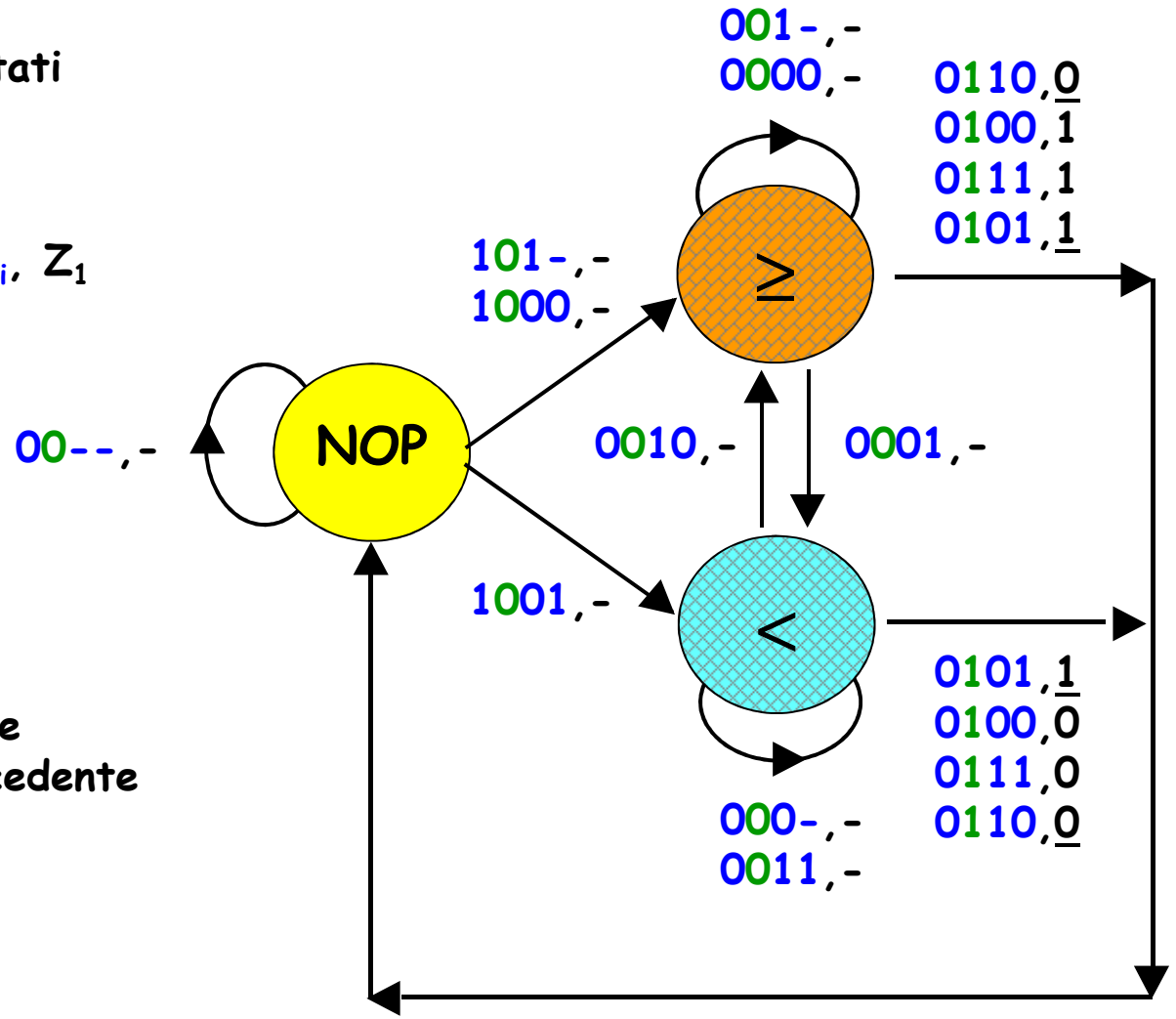
Progetto dell'unità di elaborazione

2° caso: little endian (M=0), signed integer (T_0=1)

Il grafo degli stati

Legenda:

START END A_i B_i, Z_1



0/1 : differenze
rispetto al caso precedente

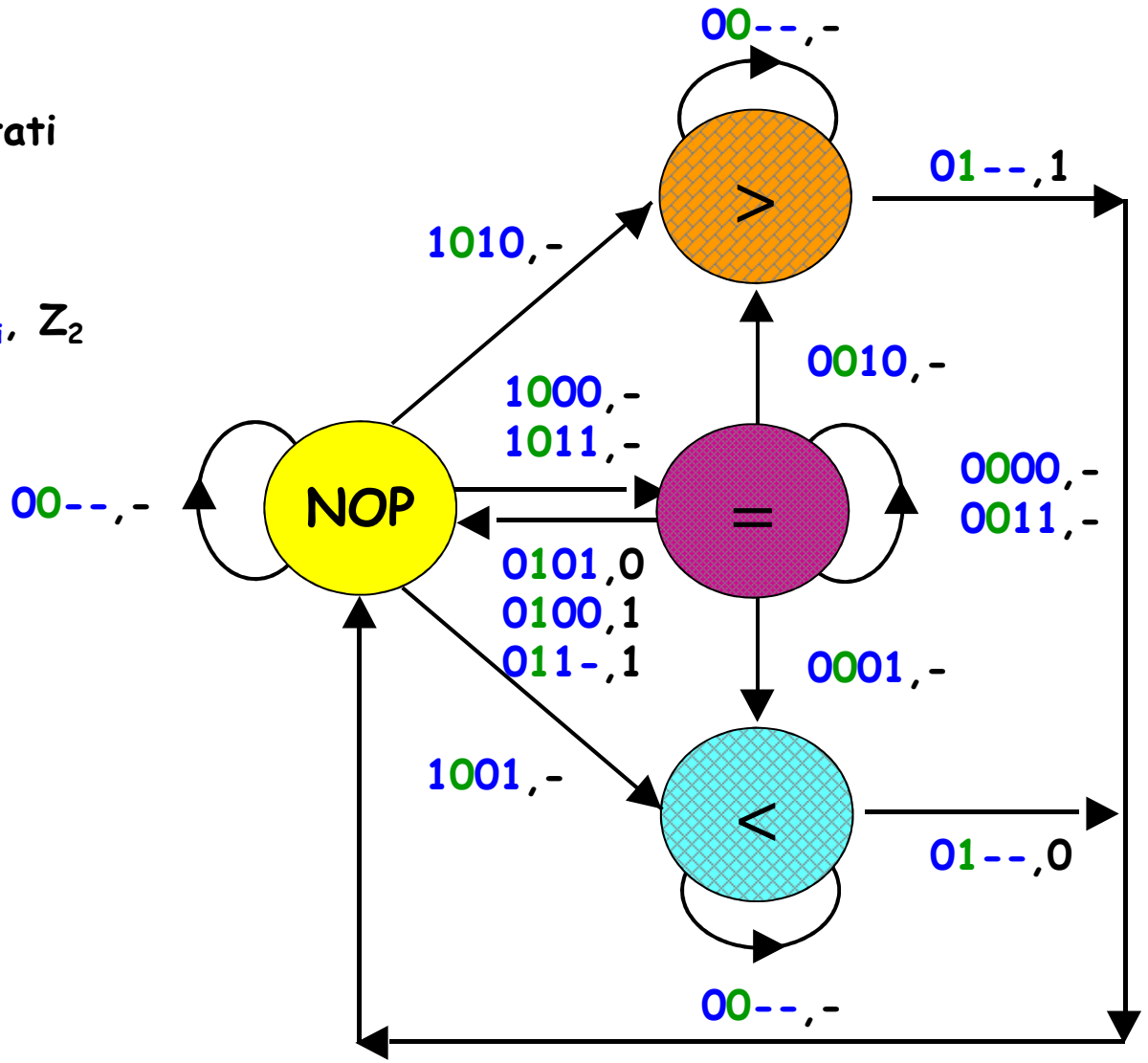
Progetto dell'unità di elaborazione

3° caso: big endian (M=1), unsigned integer (T_0=0)

Il grafo degli stati

Legenda:

START END A_i B_i, Z_2



Progetto dell'unità di elaborazione

4° caso: big endian (M=1), signed integer (T_0=1)

Il grafo degli stati

Legenda:

START END A_i B_i, Z_3

<u>0</u>/<u>1</u> : differenze
rispetto al caso precedente