Bruno Fadini *Nicola Mazzocca*

Reti logiche: complementi ed esercizi

Liguori Editore

Pubblicato da Liguori Editore
Via Posillipo 394, 80123 Napoli

Prima edizione italiana Ottobre 1995

9 8 7 6 5 4 3 2 1 0

2002 2001 2000 1999 1998 1997 1996 1995

Le cifre sulla destra indicano il numero e l'anno dell'ultima ristampa effettuata

Printed in Italy, Officine Grafiche Liguori, Napoli

ISBN 88-207-2568-1

Indice

Cap. IV: Reti sequenziali asincrone

Cap. V: Reti sequenziali sincrone

Cap. VI: Reti composte

Prefazione

Questo testo è una raccolta di esercizi svolti per il progetto di reti logiche, tratti dalle esercitazioni tenute nei corsi di Calcolatori elettronici e di Reti logiche presso i Corsi di Laurea e di Diploma in Ingegneria dell'Informazione delle Università di Napoli. Il testo fa riferimento alla teoria e ai metodi di progettazione sviluppati nei volumi "Teoria e progetto delle reti logiche" di B. Fadini e A. Esposito e "Macchine per l'elaborazione delle informazioni" di B. Fadini e U. De Carlini, pubblicati dallo stesso editore. La teoria è brevemente richiamata all'inizio di ogni capitolo e all'interno dei paragrafi destinati alla presentazione degli esercizi.

Il testo privilegia l'aspetto metodologico, nel senso che fa in ogni caso riferimento ai modeli teorici per il progetto delle reti, anche laddove poteva essere invocata una soluzione intuitiva o ad hoc. E' stato perciò necessario sviluppare alcuni complementi alla citata teoria, che sono sviluppati talora nei paragrafi iniziali, talora negli stessi esercizi.

Nella scelta degli esercizi, si sono preferiti quelli che danno luogo a circuiti realizzati come prodotti commerciali della famiglia TTL oppure a reti che comunque si realizzano adoperando gli stessi prodotti.

Gli esercizi sono stati tutti simulati ed al testo è accluso un dischetto, contenente tutti i circuiti presentati, che può essere adoperato per un approfondimento dello studio dei circuiti stessi e soprattutto della loro tempificazione. Il programma adoperato per la simulazione dei circuiti logici è Logic Works ™, un pacchetto didattico per la simulazione di circuiti logici prodotto dalla "Capilano Computing". Il pacchetto è installato presso i laboratori per le esercitazioni ed è distribuito con il testo "Logic Works- Interactive Circuit Design Software", edito dalla "The Benjamin/Cummings Publishing Inc.".

Per ogni esercizio sviluppato, è segnalato il tipo di circuito che ne deriva e l'obiettivo che si pone l'esercizio stesso; in qualche caso, infatti, l'obiettivo primario è una esercitazione di progettazione, in altri un approfondimento

della teoria, in altri ancora la conoscenza di circuiti fondamentali e così via. Inoltre, per alcuni esercizi è posto in evidenza il riferimento ai testi che trattano l'argomento teorico corrispondente oppure lo stesso circuito di cui all'esercizio: con l'acronimo RL è allora indicato il testo "Teoria e progetto delle reti logiche", con MEI quello di "Macchine per l'elaborazione delle informazioni". Infine, laddove la rete sviluppata corrisponda ad una commerciale, nei riferimenti è indicata la sigla commerciale del prodotto.

Il testo tratta in capitoli separati gli argomenti "Reti combinatorie", "Alee", "Flip-flop", "Reti sequenziali fondamentali o asincrone", "Reti sincrone impulsive", "Reti composte e progetto di sistemi"; peraltro, esso non rispetta una rigida sequenzialità didattica, nel senso che ritiene noto il contenuto fondamentale di tutta la disciplina fin dal primo capitolo. Ciò consente una serie di osservazioni e di complementi altrimenti non fattibili: basti pensare al caso dei flip-flop, che sono i componenti fondamentali delle reti logiche, ma sono essi stessi reti logiche ed adoperano come loro componenti altri flip-flop. E' allora necessario già conoscere la teoria e la tecnica di progetto delle reti logiche sequenziali quando si studiano i flip-flop, che pure precedono il capitolo del progetto delle reti sequenziali, in quanto queste ultime adoperano i primi.

Capitolo primo
Reti combinatorie

1. Richiami teorici

Una rete combinatoria è una macchina nella quale l'uscita è funzione soltanto della combinazione di valori dei segnali in ingresso, in antitesi con la rete sequenziale, nella quale l'uscita è anche funzione dello "stato interno". Una rete combinatoria avente n ingressi booleani $x_1 \ldots x_n$ ed m uscite booleane $y_1 \ldots y_m$ è descritta da m funzioni booleane:

$$y_i = f_i(x_1, \ldots, x_n)$$

Poiché ogni funzione booleana può essere espressa mediante le funzioni elementari and, or, not, le funzioni di cui sopra si possono tutte realizzare in una forma elementare di primo tipo (and-or):

$$y_i = \gamma_1 + \ldots + \gamma_k \qquad (1)$$

ove ciascuna delle γ_i è una clausola, cioè un prodotto di letterali oppure un unico letterale, ad esempio:

$$y_i = a \cdot \bar{b} + \bar{a} \cdot b \cdot c + d$$

Dualmente, le funzioni si possono realizzare in forma or-and:

$$y_i = \sigma_1 \cdot \ldots \cdot \sigma_t \qquad (2)$$

con le σ_i clausole-somma, cioè somme di letterali oppure unico letterale.

Ne scaturiscono le forme a 2 livelli and-or (or-and) delle reti, costituite da un primo livello di porte and (or) e da un secondo livello di una sola porta or (and).

Le forme circuitali a 2 livelli and-or (or-and) sono poco usate, in quanto, essendo costituite da soli circuiti passivi, presentano problemi elettronici di carico. Le porte nand e nor sono viceversa attive elettronicamente ed inoltre costituiscono da sole insiemi funzionalmente completi (f. c.): ogni funzione booleana si può realizzare esclusivamente con porte nand (nor).

Si può in particolare dimostrare che una forma elementare and-or si trasforma banalmente in una forma nand a 2 livelli: basta sostituire tutte le porte con porte nand mantenendo le priorità e negare le clausole costituite da un solo letterale, ad esempio (con riferimento all'esempio precedente):

$$y_i = (a \uparrow \bar{b}) \uparrow (\bar{a} \uparrow b \uparrow c) \uparrow \bar{d}$$

Quindi:

le forme elementari and-or (1) si possono realizzare circuitalmente con forme a 2 livelli nand;

e dualmente

le forme elementari or-and (2) si possono realizzare circuitalmente con forme a 2 livelli nor.

In qualche caso, si realizza la funzione negata della (1), negandola poi in un livello finale (dualmente per la 2):

$$y_i' = \overline{y_i}; \qquad y_i' = \gamma_1' + \ldots + \gamma_q'; \qquad y_i = \overline{\gamma_1' + \ldots + \gamma_q'} = \gamma_1' \downarrow \ldots \downarrow \gamma_q'$$

Il progetto logico combinatorio richiede in genere la stesura della tabella di verità e la ricerca della forma minima, che avviene con le mappe di Karnaugh per funzioni fino a 4-5 variabili, con il metodo tabellare di Mc Cluskey e il metodo delle righe e colonne dominanti per funzioni di più variabili. Per funzioni di molte variabili si adoperano anche metodi di ricerca euristici di quasi-minimo. Esistono programmi di calcolo per la ricerca delle forme minime o quasi-minime.

Con reti combinatorie si realizzano alcune macchine elementari fondamentali (decodificatori e codificatori, multiplexer e demultiplexer, controllori e generatori di parità, ecc.), che fungono da componenti di reti più complesse.

Questo capitolo è in generale dedicato allo studio della struttura delle reti elementari di cui sopra, che costituiscono altresì i primi circuiti commerciali. Per essi si sviluppa il progetto, peraltro in genere banale. Altri esempi di progetto di reti combinatorie, viste come componenti di reti

sequenziali o in generale di reti composte, sono inclusi nei capitoli che seguono.

2. Decodificatore completo - Demultiplexer

Tipo di circuito: rete combinatoria
Riferimento: MEI, VII-2; circuito commerciale 74138
Obiettivo: conoscenza delle reti fondamentali

2.1 Decodificatore

Testo

Si ricorda che un decodificatore binario "completo" è una macchina con k ingressi e $m=2^k$ uscite, ciascuna delle quali è attiva soltanto in corrispondenza di uno dei 2^k valori di ingresso. Il decoder viene anche detto 1/m.

Progettare un decoder 1/8 0-attivo.

Progetto

Dette $y_0 \ldots y_7$ le m=8 uscite, E l'abilitazione, x_i i k ingressi e P_i i mintermini di questi ultimi, un decoder 1-attivo, con abilitazione 1-attiva ha per equazioni:

$$y_i = E \cdot P_i \qquad (i=0..7)$$

Se l'abilitazione e le uscite sono 0-attive, si ha:

$$y_i = \overline{\bar{E} \cdot P_i} \qquad (i=0..7)$$

e quindi la and si trasforma in nand, la E nel suo negato.

Descrizione del circuito (fig. 2.1)

Il circuito è realizzato con 8 nand, una per ciascuna uscita, a quattro ingressi: 3 per realizzare il mintermine e uno per l'abilitazione. Gli ingressi non sono inviati direttamente alle nand per problemi di carico (e quindi di fan-in), in quanto essi dovrebbero alimentare 4 nand e un invertitore; sono invece applicati subito ad un unico invertitore, che costituisce così il carico

per il circuito a monte, fungendo anche da amplificatore (di potenza) dell'ingresso.

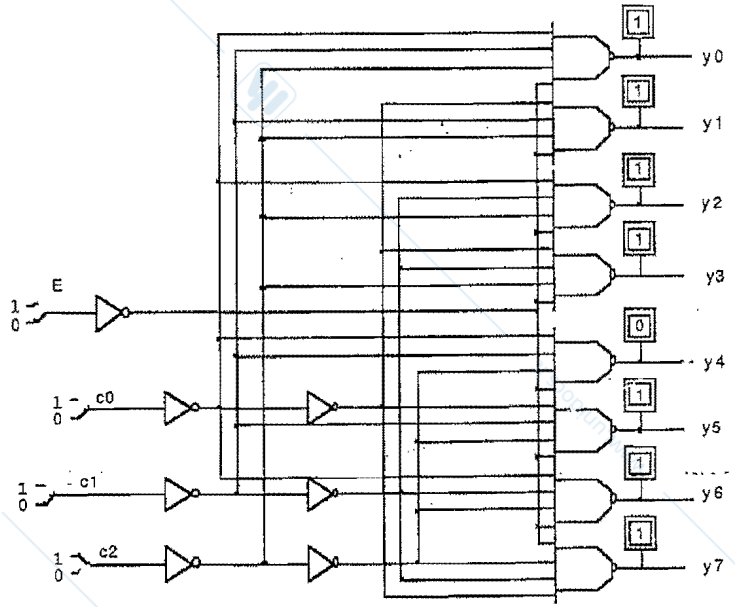


Fig. 2.1: Decodificatore 1/8

2.2 Demultiplexer

Si ricorda che un demultiplexer binario indirizzabile ad n uscite è una macchina che smista l'unico ingresso binario E su quella delle uscite y_i individuata dall'indirizzo specificato dai $k=\log_2 n$ bit-indirizzo:

$$y_i = E \cdot P_i$$

ove i P_i sono i mintermini dei k bit-indirizzo.

	decoder	demux
E	abilitazione	bit-dato
$c_0 \ldots c_{k-1}$	codice	indirizzo
$y_0 \ldots y_{n-1}$	decodifica	bit-dati

Fig 2.2: Decoder e demultplexer

Il demultiplexer coincide dunque con il decodificatore sulla base della corrispondenza di cui alla figura 2.2.

3. Decodificatore decimale

Tipo di circuito: rete combinatoria
Riferimento: MEI, VII-3, circuito commerciale 74141
Obiettivo: conoscenza delle reti fondamentali

Testo

Si ricorda che un decodificatore è "incompleto" se, avendo k ingressi e m uscite, è $m < 2^k$. Un decodificatore incompleto o è realizzato con uno completo, usandone solo alcune uscite, oppure è progettato ad hoc. In quest'ultimo caso, i punti di non specificazione corrispondenti ai codici non previsti in ingresso consentono di semplificare il circuito.

Progettare un decodificatore del codice BCD (8421) 0-attivo.

Progetto

Per il decoder 1-attivo, dette $f_0..f_9$ le m=10 uscite, E l'abilitazione, $c_0...c_3$ i 4 ingressi e poste le 10 funzioni in un'unica mappa di Karnaugh come in fig. 3.1, si ottengono le seguenti equazioni:

c_1c_0 \ c_3c_2	00	01	11	10
00	f_0	f_4	--	f_8
01	f_1	f_5	--	f_9
11	f_3	f_7	--	--
10	f_2	f_6	--	--

Fig. 3.1: Mappa delle funzioni del decoder decimale

$$y_0 = E \cdot \overline{c_3} \cdot \overline{c_2} \cdot \overline{c_1} \cdot \overline{c_0} \qquad y_2 = E \cdot \overline{c_2} \cdot c_1 \cdot \overline{c_0} \qquad y_5 = E \cdot c_2 \cdot \overline{c_1} \cdot c_0 \qquad y_8 = E \cdot c_3 \cdot \overline{c_0}$$
$$y_1 = E \cdot \overline{c_3} \cdot \overline{c_2} \cdot \overline{c_1} \cdot c_0 \qquad y_3 = E \cdot \overline{c_2} \cdot c_1 \cdot c_0 \qquad y_6 = E \cdot c_2 \cdot c_1 \cdot \overline{c_0} \qquad y_9 = E \cdot c_3 \cdot c_0$$
$$y_4 = E \cdot c_2 \cdot \overline{c_1} \cdot \overline{c_0} \qquad y_7 = E \cdot c_2 \cdot c_1 \cdot c_0$$

Per il decoder 0-attivo l'ingresso E si tramuta in $\overline{E}$ e le and in nand. Su questa base è realizzato il circuito commerciale 74141.

Descrizione del circuito (fig. 3.2)

Il circuito è realizzabile con 2 nand (y_1, y_0) a 5 ingressi (4 per le P_i e 1 per E), 5 (da y_2 a y_6) a 4 ingressi (3+1) e 2 (y_8, y_9) a 3 (2+1). Non disponendo di nand a 5 ingressi, si è realizzato il prodotto $\overline{c_3} \cdot \overline{c_2} \cdot \overline{c_1}$ con una and e questa è stata posta in ingresso alle due nand per y_1 e y_0. (si potrebbe anche usare una nand a otto ingressi, fissandone tre ad "1").

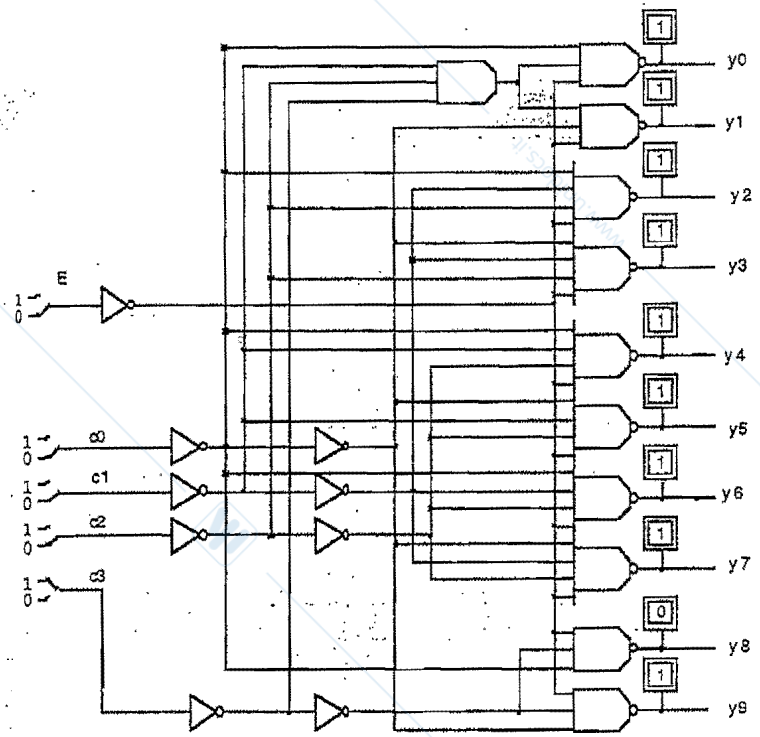


Fig. 3.2: Decodificatore BCD

Similmente a quanto visto al § 2, gli ingressi non sono inviati direttamente alle nand per problemi di carico, in quanto essi dovrebbero alimentare più di una porta; sono invece applicati subito ad un unico invertitore, che costituisce così il carico per il circuito a monte, fungendo anche da amplificatore (di potenza) dell'ingresso.

4. Codificatore a priorità

Tipo di circuito: rete combinatoria
Riferimento: MEI, VII-4, circuito commerciale 9318
Obiettivo: conoscenza delle reti fondamentali

Testo

Si ricorda che un codificatore binario "completo" è una macchina con $m=2^k$ ingressi e k uscite. Gli ingressi hanno il vincolo che al più 1 è attivo e, allorché uno di essi è attivo, l'uscita ne è il codice binario corrispondente. Se la condizione di vincolo non è rispettata, si stabilisce una priorità fra gli ingressi binari e si realizza un "codificatore a priorità", per il quale viene considerato attivo l'ingresso prioritario.

Progettare un codificatore a priorità 0-attivo, con abilitazione 0-attiva ed 8 ingressi binari.

Progetto

Per il codificatore completo 1-attivo, detti x_0 ... x_7 gli m=8 ingressi, E l'abilitazione, $c_0 \ldots c_2$ i 3 bit-codice in uscita e supposto che il codificatore non abilitato dia in uscita il codice di tutti 0, si ha:

$$c_2 = E \cdot (x_4 + x_5 + x_6 + x_7)$$
$$c_1 = E \cdot (x_2 + x_3 + x_6 + x_7)$$
$$c_0 = E \cdot (x_1 + x_3 + x_5 + x_7)$$

Per trasformare il codificatore in uno a priorità, occorre premettere una "rete di priorità" che "filtri", fra i vari ingressi, quello prioritario. Detti $x_7 \ldots x_0$ gli ingressi primari (che non rispettano il vincolo) ed $x'_7 \ldots x'_0$ quelli "filtrati" (attivo uno solo), le equazioni della rete di priorità sono in generale:

$$x'_i = x_i \cdot \overline{x_{i+1}} \cdot \ldots \cdot \overline{x_7}$$

e, spostando sulla rete a priorità l'abilitazione, si otterrebbe in definitiva:

$$x'_i = E \cdot x_i \cdot \overline{x_{i+1}} \cdot \ldots \cdot \overline{x_7} \qquad (1)$$

$$c_2 = x_4' + x_5' + x_6' + x_7'$$
$$c_1 = x_2' + x_3' + x_6' + x_7' \qquad (2)$$
$$c_0 = x_1' + x_3' + x_5' + x_7'$$

Si consideri ora che, in assenza di rete di priorità, le equazioni del codificatore sono tali che, ove più variabili primarie siano attive, il codice che ne risulterebbe sarebbe quello corrispondente alla or dei codici dei singoli ingressi: se ad esempio fosse $x_6=x_5=1$, essendo il primo codice 110, il secondo 101 e la loro unione 111, ne risulterebbe il codice di x_7. Sono allora automaticamente verificate le seguenti priorità:

- x_7 (codice 111) è prioritario su tutti;
- x_6 (codice 110) è prioritario su x_4, x_2;
- x_5 (codice 101) è prioritario su x_4, x_1;
- x_3 (codice 011) è prioritario su x_2, x_1;
- tutti sono prioritari su x_0 (ma x_0 non compare in nessuna equazione).

L'equazione generale (1) si specializza allora come segue:

$$x_7' = E \cdot x_7 \qquad x_4' = E \cdot x_4 \qquad x_1' = E \cdot x_1 \cdot \overline{x_2} \cdot \overline{x_4} \cdot \overline{x_6}$$
$$x_6' = E \cdot x_6 \qquad x_3' = E \cdot x_3 \cdot \overline{x_4} \cdot \overline{x_5} \cdot \overline{x_6} \qquad (1')$$
$$x_5' = E \cdot x_5 \cdot \overline{x_6} \qquad x_2' = E \cdot x_2 \cdot \overline{x_4} \cdot \overline{x_5}$$

Un utile segnale ausiliario può indicare che l'uscita del codificatore è significativa, cioè che il codificatore è abilitato e vi è un dato in ingresso:

$$p = E \cdot \sum_{i=0}^{7} x_i$$

mentre un secondo segnale ausiliario può indicare il caso in cui l'uscita 000 non è dovuta al fatto che si abbia $x_0=1$, ma al caso in cui sono nulli tutti gli ingressi:

$$n = E \cdot \overline{\sum_{i=0}^{7} x_i} = \overline{\sum_{i=0}^{7} x_i'} = \prod_{i=0}^{7} \overline{x_i'} \qquad (3)$$

e si può anche porre:

$$p = E \cdot \overline{n} \qquad (4)$$

Per trasformare la rete in 0-attiva
- tutti gli ingressi si trasformano nei loro negati;
- le (2) si trasformano in nor;
- la (3) si trasforma in nand;
- la (4) diventa una nand fra E e la nand della (3).

Descrizione del circuito (fig. 4.1)

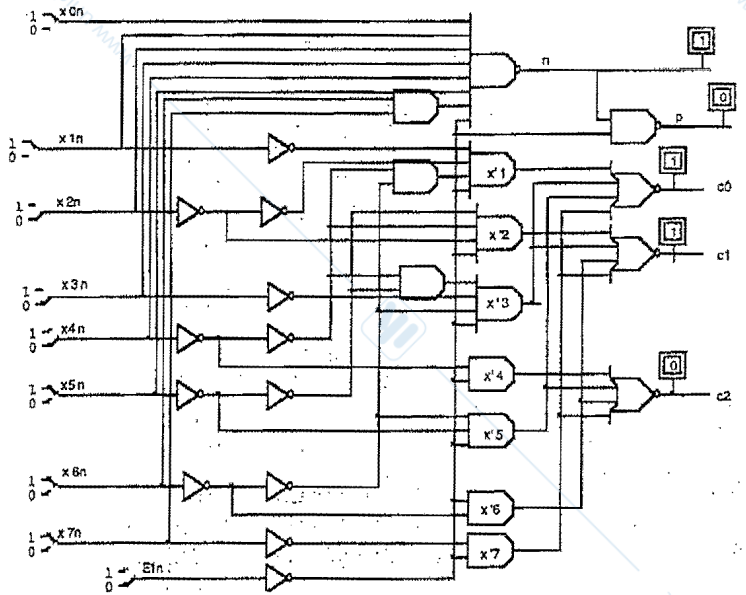


Fig. 4.1: Codificatore a priorità

Analogamente a quanto visto al § 3, si ha:
- gli ingressi non sono inviati direttamente alle nand per problemi di carico (e quindi di fan-in), in quanto essi dovrebbero alimentare più di una porta; sono invece applicati subito ad un unico invertitore, che costituisce così il carico per il circuito a monte, fungendo anche da amplificatore (di potenza) dell'ingresso.

- poiché il simulatore non dispone di nand a 9 né a 6 ingressi, si sono realizzati con alcune and i prodotti parziali, che poi sono posti in ingresso alle nand per n, x'_1 e x'_3 (per le nand a 6 si potrebbero anche adoperare nand a otto ingressi, fissandone ad "1" due).

5. Trascodificatore per visualizzatore a 7 segmenti

Tipo di circuito: rete combinatoria
Riferimento: circuito commerciale 7448
Obiettivo: progettazione

Visualizzatore a 7 segmenti

Un visualizzatore a 7 segmenti è un dispositivo dotato di 7 lampadine disposte come in figura 5.1; accendendo una combinazione di queste così come indicato nella stessa figura, si ottiene la visualizzazione di una delle 10 cifre decimali. Dal punto di vista logico, presenta un ingresso di 4 bit, corrispondenti al codice 8421, ed un codice a 7 bit associato a ciascuna cifra decimale, che indica la corrispondente combinazione di lampadine da accendere.

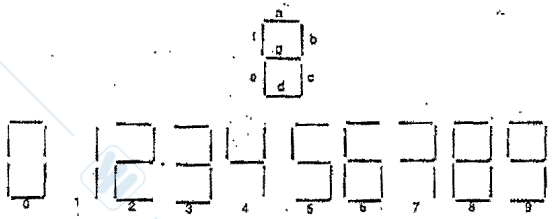


Fig. 5.1: Visualizzatore a 7 segmenti

Testo

Si ricorda che un trascodificatore è una macchina che, dati due distinti codici per codificare il medesimo insieme di oggetti, trasforma l'uno nell'altro. Progettare un trascodificatore dal codice 8421 a quello di un visualizzatore a 7 segmenti.

Si noti che questa macchina viene anche detta (impropriamente) decodificatore, in quanto rende evidente (sul visualizzatore) il codice in ingresso.

Progetto

La tabella di verità del trascodificatore è banale: è sufficiente allineare sulla stessa riga di una tabella i due codici (cfr. fig. 5.2); a ciascuna lampada è associata una distinta funzione dei quattro bit-codice.

c_3	c_2	c_1	c_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	1
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

Fig. 5.2: Tabella di verità

Ne risultano le sette mappe di Karnaugh di figura 5.3.

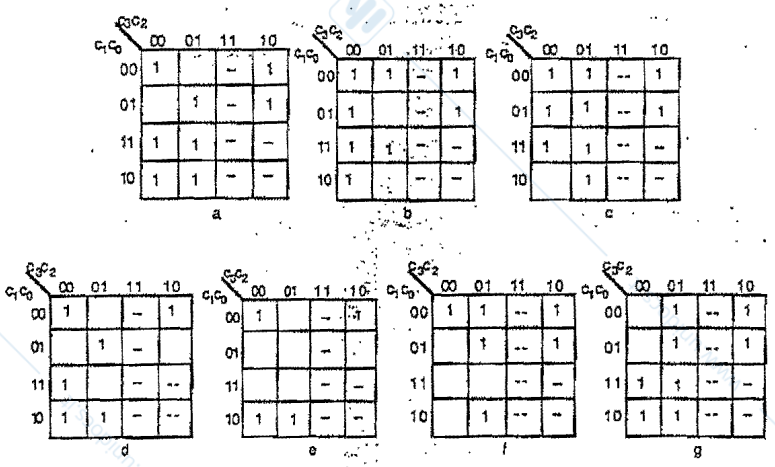


Fig. 5.3: Progetto delle 7 funzioni

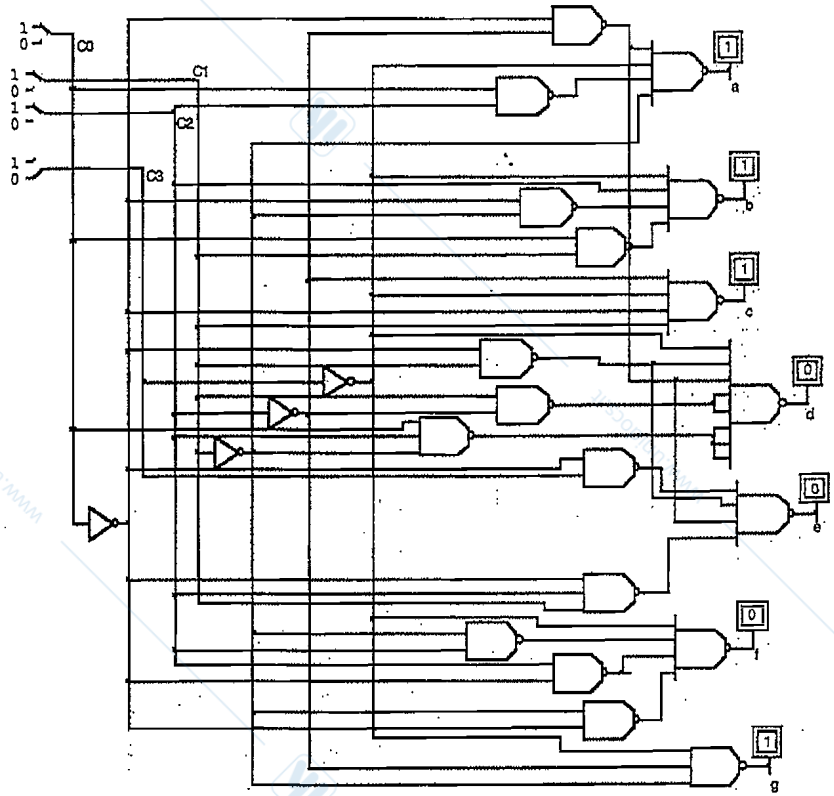


Fig. 5.4: Codificatore a priorità

Dalle mappe scaturiscono semplicemente le funzioni:

$$a = c_3 + c_2 \cdot c_0 + \overline{c_2} \cdot \overline{c_0} + c_1$$
$$b = c_3 + \overline{c_2} + \overline{c_1} \cdot \overline{c_0} + c_1 \cdot c_0$$
$$c = c_3 + c_2 + \overline{c_1} + c_0$$
$$d = c_3 + \overline{c_2} \cdot c_1 + c_2 \cdot \overline{c_1} \cdot c_0 + \overline{c_2} \cdot \overline{c_0} + c_1 \cdot \overline{c_0}$$
$$e = \overline{c_2} \cdot \overline{c_0} + c_1 \cdot \overline{c_0}$$
$$f = c_3 + c_2 \cdot \overline{c_1} + c_2 \cdot \overline{c_0} + \overline{c_1} \cdot \overline{c_0}$$
$$g = c_3 + c_2 + c_1$$

Descrizione del circuito (fig. 5.4)

Il circuito è stato realizzato con porte NAND, utilizzando una porta ad 8 ingressi per una funzione di 5 variabili. Si noti l'uso di una medesima porta di primo livello, corrispondente alla clausola $\overline{c_0} \cdot \overline{c_2}$, per le tre funzioni a, d, e. In generale in un progetto combinatorio si possono sviluppare semplificazioni di tal genere; alcuni strumenti di ausilio alla progettazione sviluppano in proposito metodi di quasi-ottimizzazione.

6. Multiplexer

Tipo di circuito: rete combinatoria
Riferimento: circuito commerciale 74153
Obiettivo: conoscenza delle reti fondamentali

Testo

Si ricorda che un multiplexer binario indirizzabile è una macchina che inoltra sull'unica uscita y quello fra gli n bit-dati in ingresso il cui indirizzo è individuato dai $k=\log_2 n$ bit-indirizzo. Progettare un multiplexer a 4 ingressi.

Progetto

Detta E l'abilitazione, y l'uscita, x_i i 4 ingressi, c_i i due bit-indirizzo e P_i i relativi mintermini, l'equazione del multiplexer è:

$$y = E \cdot \sum_{i=0}^{3} x_i \cdot P_i$$

in quanto, se abilitato (E=1), su y viene inoltrato x_0 se i bit-indirizzo esprimono l'indirizzo 000 (associato a P_0), x_1 per l'indirizzo 001, e così via.

Descrizione del circuito (fig. 6.1)

La rete è banalmente riprodotta in figura. Si noti soltanto che:
- l'abilitazione è 0-attiva;
- i segnali c_1, c_0 non sono applicati direttamente alle and, ma tramite un invertitore (cfr. § 2, 3, 4).

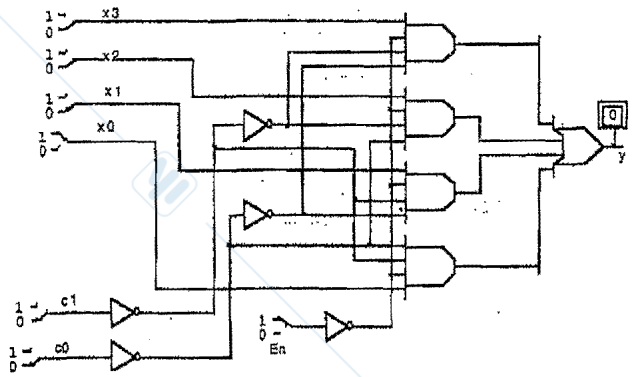


Fig. 6.1: Multiplexer a 4 ingressi

7. Rete di parità

Tipo di circuito: rete combinatoria
Riferimento: circuito commerciale 74180
Obiettivo: conoscenza delle reti fondamentali

Testo

Si ricorda che una rete di parità fornisce in uscita un segnale binario y che indica se il numero di bit 1 in ingresso è pari o dispari. Si dicono di parità sia le reti per le quali y=1 indica la parità (reti di parità in senso stretto), sia quelle per le quali y=1 indica la disparità (reti di disparità).

Progettare una rete di disparità su 9 bit. Studiare l'uso di detta rete nei casi di generazione e di controllo del bit di parità.

Progetto

Essendo la funzione di disparità associativa, la rete può essere costruita adoperando come componenti funzioni di disparità su due variabili e computando la disparità della disparità ..... della disparità a gruppi di 2, come nella espressione che segue e nel circuito di figura 7.1:

$$d(x_0,\ldots,x_8) = d\Big(d\Big\{d\big[d(x_0,x_1),\ d(x_2,x_3)\big],\ d\big[d(x_4,x_5),\ d(x_6,x_7)\big]\Big\},\ x_8\Big)$$

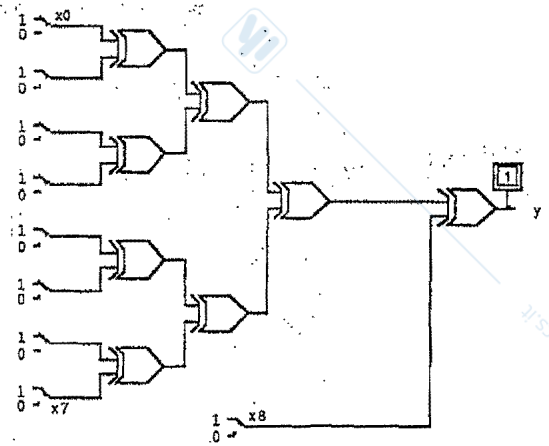


Fig. 7.1: Rete di parità: schema logico

Analisi del circuito

La rete può essere adoperata sia in sede di generazione del bit di parità sia in sede di controllo.
In sede di generazione:
- $x_0 \ldots x_7$ sono gli 8 bit di un byte, x_8 indica se il bit di parità deve essere tale che i 9 bit abbiano parità pari (x_8=0) oppure dispari (x_8=1): y è il bit di parità.

In sede di controllo :
- per un controllo su 9 bit (1 byte + bit di parità), y è la disparità sui 9 bit: se il bit di parità è tale che sui 9 bit la parità è dispari, y=1 significa che il controllo ha avuto successo, y=0 segnala l'errore; se invece il bit di parità fornisce parità pari sui 9 bit, y=1 è il segnale di errore;
- per un controllo su 8 bit, $x_0 \ldots x_7$ sono gli 8 bit, x_8 indica se la parità sugli 8 deve essere pari (x_8=0) oppure dispari (x_8=1); y=1 è allora il segnalatore di errore.

Descrizione del circuito (fig. 7.2)

La rete di figura 7.1 è ridisegnata in figura 7.2, con l'aggiunta di alcune porte NOT con funzioni di amplificatori di potenza (i NOT sono in realtà incorporati nel circuito XOR, che diventa dunque uno di equivalenza). Su questa base è realizzato il circuito commerciale 74180.

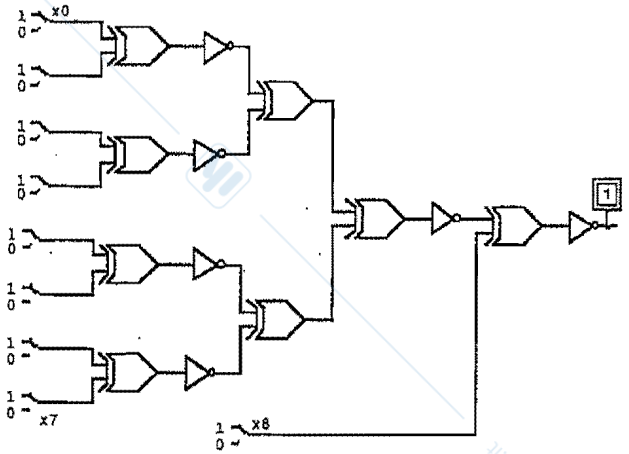


Fig. 7.2: Rete di parità con invertitori

8. Full adder

Tipo di circuito: rete combinatoria
Riferimento: MEI, VI-11; circuito commerciale 74183
Obiettivo: conoscenza delle reti fondamentali

Testo

Si ricorda che si intende per full adder una macchina per l'addizione di due cifre binarie che tenga conto del riporto entrante e fornisca somma e riporto uscente. Progettare un full adder binario.

Progetto

Detti a, b i due addendi, c il riporto entrante, S il bit-somma e R il bit-riporto in uscita, le forme elementari minime delle funzioni S e R sono:

$$S = \bar{a}\cdot\bar{b}\cdot c + \bar{a}\cdot b\cdot\bar{c} + a\cdot\bar{b}\cdot\bar{c} + a\cdot b\cdot c$$
$$R = a\cdot b + b\cdot c + a\cdot c$$

Con tali funzioni a due livelli si ottiene un circuito veloce; viceversa, la soluzione con due half-adder è a più livelli e quindi più lenta.

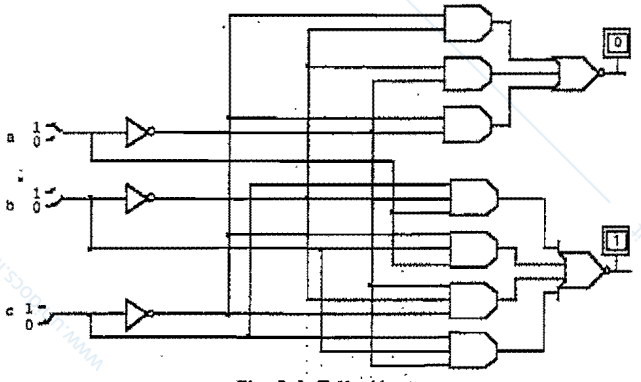


Fig. 8.1: Full adder

Descrizione del circuito (fig. 8.1)

Per motivi di carico elettronico, il circuito o si realizza con tutte porte nand oppure con una nor al secondo livello, lasciando le and al primo. Poiché la nor inverte, essa opera su S_c e R_c, funzioni complemento di S e R:

$$S = \overline{S_c} = \overline{\bar{a}\cdot b\cdot c + a\cdot\bar{b}\cdot c + a\cdot b\cdot\bar{c} + \bar{a}\cdot\bar{b}\cdot\bar{c}}$$
$$R = \overline{R_c} = \overline{\bar{a}\cdot\bar{c} + \bar{a}\cdot\bar{b} + \bar{b}\cdot\bar{c}}$$

Su questa base è costruito il prodotto commerciale 74183.

9. Decodificatore composto

Tipo di circuito: rete combinatoria
Riferimento: MEI, III-3; circuiti commerciali 74138, 74139
Obiettivo: uso di circuiti commerciali

Testo

Progettare un decodificatore 1/32 adoperando circuiti commerciali.

Progetto

Non esistono decodificatori con 32 linee binarie di uscita. Si realizza allora il decodificatore con una struttura ad albero.

Descrizione del circuito (fig. 9.1)

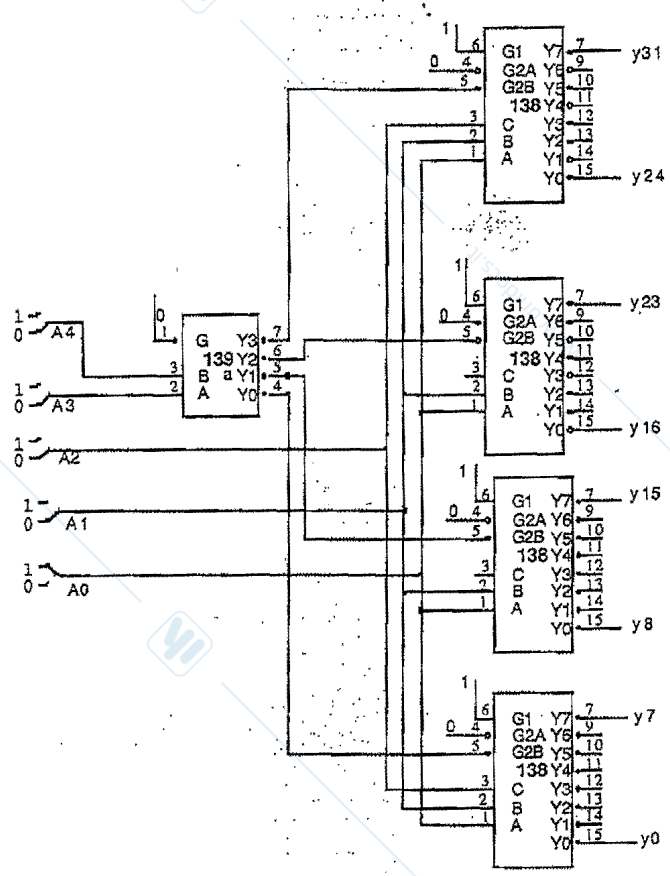


Fig. 9.1: Decodificatore 1/32

Si adopera un decoder 1/4, il 74139, che opera sui bit-indirizzo A_4, A_3. Si selezionano così 4 gruppi di codici: quello che inizia con 00, cioè i codici 00xxx e poi i codici 01xxx, 10xxx, 11xxx. La seconda parte del codice è

decodificata da 4 decodificatori 1/8 (74138), ciascuno dei quali è abilitato da una delle uscite del decodificatore 1/4 ed opera sui bit A_2, A_1 e A_0.

Il decodificatore 74139 è abilitato da G, che è posto identicamente a 0. Il decodificatore è in realtà un doppio decodificatore 1/4, del quale si adopera soltanto una delle due sezioni; in figura è appunto disegnata la sola sezione utilizzata.

Il decodificatore 74138 è abilitato dalla and $E = G1 \cdot \overline{G2A} \cdot \overline{G2B}$; essendo i decoder impiegati 0-attivi, si è adoperato il segnale $\overline{G2B}$ come abilitazione, ponendo identicamente $G1 = 1$, $\overline{G2A} = 0$.

10. Multiplexer composto

Tipo di circuito: rete combinatoria
Riferimento: MEI, VII-9; circuito commerciale 74153
Obiettivo: uso di circuiti commerciali

Testo

Progettare un multiplexer a 16 ingressi adoperando circuiti commerciali.

Progetto

Non disponendo di un multiplexer a 16 ingressi, lo si realizza mettendo assieme multiplexer più semplici: in particolare, si adoperano come componenti multiplexer a 4 ingressi.

Descrizione del circuito (fig. 10.1)

Si adopera il circuito 74153 che racchiude 2 multiplexer a 4 ingressi, entrambi selezionati dai medesimi bit-indirizzo. Pertanto al primo livello vi sono 2 chip (4 multiplexer), che operano sui bit-indirizzo A_1, A_0; questi bit, quindi, selezionano simultaneamente 4 linee di ingresso, una per ciascun multiplexer.

Al secondo livello è posto un unico multiplexer 1/4 (il chip in effetti ne contiene 2, ma se ne adopera uno solo), che, operando sui bit A_3, A_2, seleziona una delle linee di uscita dei multiplexer di primo livello.

Tutti i multiplexer sono abilitati, avendo posto identicamente GA=0 (GA è una abilitazione 0-attiva).

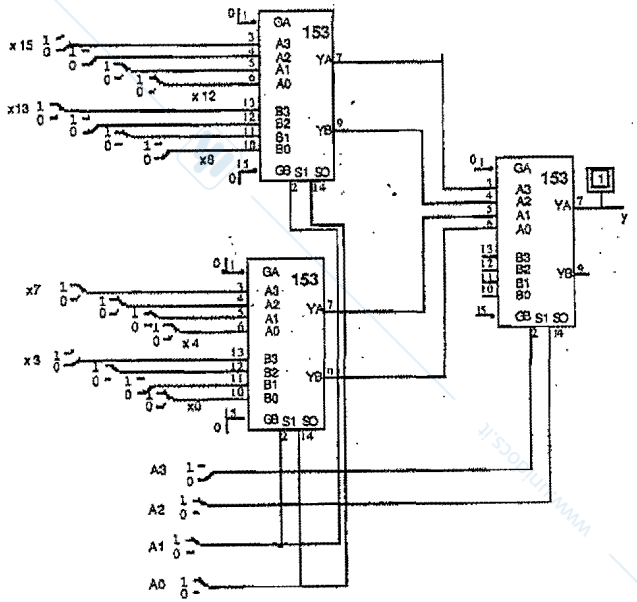


Fig. 10.1: Multiplexer a 16 ingressi

11. Generatore di funzioni booleane

Tipo di circuito: rete combinatoria
Riferimento: MEI, VII-10; circuito commerciale 74151 e 74153
Obiettivo: progettazione combinatoria

Testo

Progettare un addizionatore completo adoperando come componenti multiplexer a 8 bit-dati oppure a 4 bit-dati (logica folded).

11.1 Progetto con multiplexer

La forma normale di una funzione booleana:

$$y = E \cdot \sum_{i=0}^{3} x_i \cdot P_i$$

coincide con l'equazione di un multiplexer (cfr.§ 6). Questo si può dunque adoperare come generatore di funzioni purché si adottino le seguenti corrispondenze:

	multiplexer	generatore
$\overline{E}$	abilitazione	abilitazione
$c_0 \ldots c_{k-1}$	indirizzo	variabili
$x_0 \ldots x_{n-1}$	bit-dati	bit-specificazione

ove i bit-specificazione della funzione da generare coincidono con la colonna di 0 e 1 della tabella di verità o, il che è lo stesso, sono i bit del numero caratteristico.

Nel caso specifico le equazioni del full adder sono:

$$S = \overline{a} \cdot \overline{b} \cdot c + \overline{a} \cdot b \cdot \overline{c} + a \cdot \overline{b} \cdot \overline{c} + a \cdot b \cdot c$$
$$R = a \cdot b + b \cdot c + a \cdot c$$

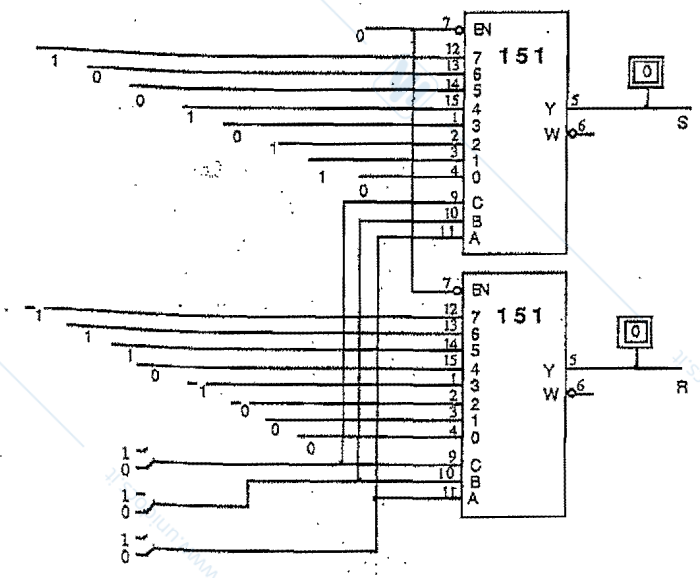


Fig. 11.1: Full adder realizzato con multiplexer

Si hanno, quindi, le seguenti espressioni e numeri caratteristici:

$$S = P_1 + P_2 + P_4 + P_7$$
$$R = P_3 + P_5 + P_6 + P_7$$

per S: $\alpha_0 = 0;\ \alpha_1 = 1;\ \alpha_2 = 1;\ \alpha_3 = 0;\ \alpha_4 = 1;\ \alpha_5 = 0;\ \alpha_6 = 0;\ \alpha_7 = 1;$
per R: $\alpha_0 = 0;\ \alpha_1 = 0;\ \alpha_2 = 0;\ \alpha_3 = 1;\ \alpha_4 = 0;\ \alpha_5 = 1;\ \alpha_6 = 1;\ \alpha_7 = 1;$

Questi valori degli α_i vanno posti sui bit-dati del multiplexer.

Descrizione del circuito (fig. 11.1)

Si sono adoperati due circuiti 74151, alimentati come descritto dalle formule di cui sopra.

11.2 Progetto in logica folded

Si ricorda che un multiplexer indirizzabile con k bit-indirizzo è anche un generatore di funzioni di k+1 variabili, con la cosiddetta "logica folded". Esso genera le funzioni ponendo in ingresso k variabili indipendenti sui bit-indirizzo e sui bit-dati una informazione che per ciascuna di esse è 0, 1 oppure un letterale della (k+1)-esima variabile.

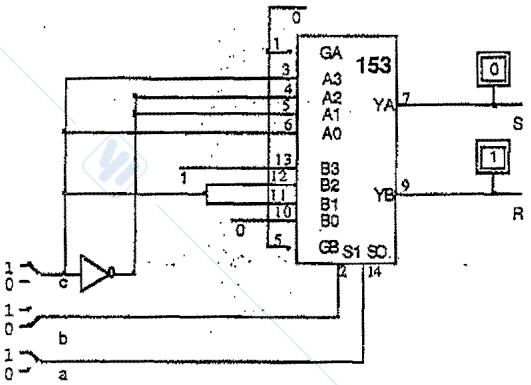


Fig. 11.2: Full adder realizzato in logica folded

Nel caso specifico, si adoperano, per le stesse funzioni di cui sopra, multiplexer di 4 bit-dati; dalle forme normali di cui sopra risulta:

per S: $\alpha_0 = c;\ \alpha_1 = \overline{c};\ \alpha_2 = \overline{c};\ \alpha_3 = c;$
per R: $\alpha_0 = 0;\ \alpha_1 = c;\ \alpha_2 = c;\ \alpha_3 = 1.$

Si noti che l'espressione di R mette in evidenza le condizioni di riporto preciso ($\alpha_0 = 0$), generato ($\alpha_3 = 1$) e propagato ($\alpha_1 = \alpha_2 = c$).

Descrizione del circuito (fig. 11.2)

Si è adoperato nella realizzazione il componente 74153, che contiene i due multiplexer necessari.

12. Anticipatore di riporto

Tipo di circuito: rete combinatoria
Riferimento: MEI, VII-13; circuito commerciale 9342
Obiettivo: approfondimento teorico

Testo

Si ricorda che in un addizionatore binario parallelo, per migliorare i tempi di ritardo, si possono usare delle apposite reti di "carry lookahead", che calcolano il riporto entrante nello stadio i-esimo mediante una rete di pochi livelli, ponendo il riporto stesso in funzione direttamente delle cifre di peso meno significativo i-1, i-2, etc. e non dei loro riporti.

Progettare un anticipatore di riporti (carry lookahead) che anticipi il riporto in 4 stadi di un addizionatore binario; la rete operi sui bit di "riporto propagato" P e "riporto generato" G calcolati nei 4 addizionatori interessati.

Progetto

Indicato con 0 il pedice del bit meno significativo, con A, B i bit-addendi, con r, R i riporti entranti ed uscenti, l'anticipatore deve generare i riporti r_1, r_2, r_3 in funzione delle condizioni di riporto generato $G = A \cdot B$ e propagato $P = A \oplus B$ degli addizionatori di peso 0, 1, 2 (fig. 12.1):

$$\begin{aligned} r_1 &= R_0 = G_0 + P_0 r_0 \\ r_2 &= R_1 = G_1 + P_1 r_1 = G_1 + P_1 G_0 + P_1 P_0 r_0 \\ r_3 &= R_2 = G_2 + P_2 r_2 = G_2 + P_2 G_1 + P_2 P_1 r_1 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 r_0 \end{aligned} \qquad (1)$$

In tal modo, i riporti negli stadi 1, 2, 3 sono computati tutti nello stesso tempo massimo relativo all'attraversamento di 2 livelli.

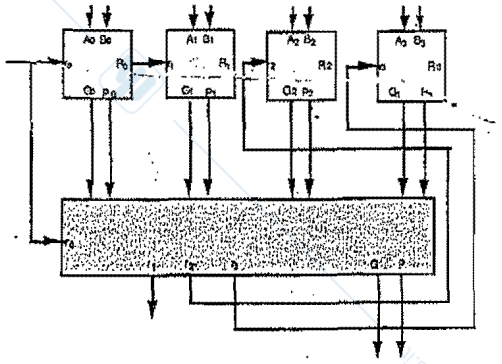


Fig. 12.1: Collegamento fra addizionatori e anticipatore di riporti

La rete può anche utilmente computare le variabili P e G che indicano le condizioni complessive di propagazione o generazione dei riporti nei 4 stadi:

$$P = P_0 + P_1 + P_2 + P_3$$
$$G = G_3 + G_2 \cdot P_3 + G_1 \cdot P_3 \cdot P_2 + G_0 \cdot P_3 \cdot P_2 \cdot P_1 \qquad (2)$$

Si può così adoperare l'anticipatore per un ulteriore livello di anticipazione dei riporti fra gruppi di 4 addizionatori di bit (come in una aritmetica in base 2^4=16).

Descrizione del circuito (fig. 12.2)

La rete è realizzata con due livelli and-nor per motivi di carico. Sono dunque realizzate le funzioni a 2 livelli negate di quelle delle (1) e (2):

$$\overline{R_0} = \overline{G_0} \cdot \left(\overline{P_0} + \overline{r_0}\right) = \overline{G_0} \cdot \overline{P_0} + \overline{G_0} \cdot \overline{r_0}$$

$$\overline{R_1} = \overline{G_1} \cdot \overline{P_1} + \overline{G_1} \cdot \overline{G_0} \cdot \overline{P_0} + \overline{G_1} \cdot \overline{G_0} \cdot \overline{r_0}$$

$$\overline{R_2} = \overline{G_2} \cdot \overline{P_2} + \overline{G_2} \cdot \overline{G_1} \cdot \overline{P_1} + \overline{G_2} \cdot \overline{G_1} \cdot \overline{G_0} \cdot \overline{P_0} + \overline{G_2} \cdot \overline{G_1} \cdot \overline{G_0} \cdot \overline{r_0} + \overline{G_2} \cdot \overline{G_1} \cdot \overline{G_0} \cdot \overline{r_0}$$

Le uscite P, G sono negate, coerentemente con gli ingressi G_i, P_i.

$$\overline{P} = \overline{P_0 \cdot P_1 \cdot P_2 \cdot P_3} = \overline{P_0} + \overline{P_1} + \overline{P_2} + \overline{P_3}$$

$$\overline{G} = \overline{G_3} \cdot \overline{G_2} \cdot \overline{G_1} \cdot \overline{G_0} + \overline{G_3} \cdot \overline{G_2} \cdot \overline{G_1} \cdot \overline{P_1} + \overline{G_3} \cdot \overline{G_2} \cdot \overline{P_2} + \overline{G_3} \cdot \overline{P_3}$$

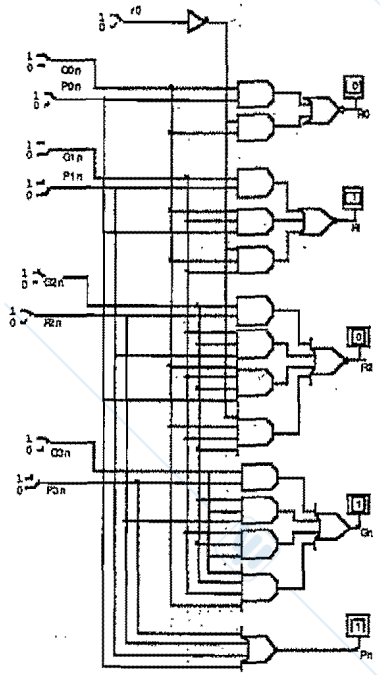


Fig. 12.2 Anticipatore di riporti

Capitolo secondo
Alee

1. Richiami teorici

Si dice che una rete possiede un'*alea* se può presentare un comportamento anomalo per effetto dei suoi ritardi interni. Un'alea è:
- *transitoria*, se le uscite della rete assumono valori diversi da quelli progettati soltanto nel transitorio conseguente alle variazioni degli ingressi. Tale alea è tipica delle reti combinatorie e viene anche detta *combinatoria*;
- *di regime*, se il valore imprevisto permane anche dopo il transitorio. Tale alea è tipica delle reti sequenziali e può derivare da un'alea transitoria della parte combinatoria oppure da fenomeni intrinseci nella rete sequenziale.

Le alee combinatorie sono:
- *Alea multipla*: deriva dalla simultanea variazione di due o più variabili di ingresso;
- *Alea per impulsi concomitanti*: deriva dalla simultanea presenza di due o più impulsi.
- *Alea statica*: si ha in un transitorio in cui, a fronte della variazione di un'unica variabile per la quale l'uscita dovrebbe permanere costante, si ha una variazione temporanea. Tale alea si può avere in reti a 2 livelli.
- *Alea dinamica:* si ha in un transitorio in cui, a fronte della variazione di un'unica variabile per la quale l'uscita dovrebbe variare, si ha una oscillazione temporanea del segnale in uscita. Tale alea si può avere in circuiti a più livelli che contengono alee statiche nei livelli intermedi.

Le alee di regime nelle reti sequenziali asincrone, oltre a quelle derivanti dalle suddette alee transitorie della parte combinatoria, sono classificabili come:
- *Alee essenziali*, intrinseche nelle proprietà della tabella di definizione della rete; se ne eliminano gli effetti aggiungendo appositi ritardi.

- Alee dovute a *corse critiche,* derivanti da una alea multipla indotta dalla codifica degli stati e dal fatto che la variazione di una variabile di stato prima o dopo dell'altra ("corsa") conduce a stati stabili differenti. Si eliminano evitando le corse, cioè codificando gli stati in modo che ogni transizione avvenga fra stati "adiacenti", cioé con la variazione di una sola variabile. E' questo il maggiore vincolo nella progettazione delle reti asincrone, in quanto tale codifica richiede spesso l'aggiunta di nuove variabili di stato (cfr. ad esempio § IV-4).
- Alee derivanti da insufficienti durate degli ingressi (cfr. ad esempio § III-4): si tratta di grossolani errori di progettazione.

Nelle reti sequenziali sincrone, alee di regime possono presentarsi con l'uso di flip-flop latch: un impulso di durata eccessiva (cfr. ad esempio § III-14) viene in pratica assunto dalla rete come sequenza successiva di più impulsi in ingresso.

In questo capitolo vengono esemplificate le alee propriamente dette, rinviando ai capitoli sulle reti sequenziali quelle derivanti dal dimensionamento dei ritardi. L'obiettivo prevalente è quello di approfondire le motivazioni e le conseguenze dei fenomeni di alea.

2. Alea multipla

Tipo di circuito: rete combinatoria
Riferimento: RL, V-5A
Obiettivo: approfondimento teoria

Testo

Si ricorda che in un circuito, allorché variano simultaneamente due ingressi, si ha una condizione cosiddetta di alea multipla: è possibile che in uscita si abbiano dei valori transitori non previsti da un progetto che tenga conto soltanto delle condizioni di regime.

Costruire un circuito esemplificativo di quanto sopra.

Descrizione del circuito (fig. 2.1)

Il circuito esemplificativo consta di un circuito equivalenza di una variabile con se stessa:

$$y = a \cdot a + \bar{a} \cdot \bar{a}$$

identicamente uguale ad 1.

Per provare il circuito è stato inviato su *a* un segnale periodico (clock). A causa del ritardo degli invertitori, il segnale *an* ($\bar{a}$) ritarda rispetto ad *a*. Allora, quando *a* scende, *an* resta ancora basso per un tempuscolo, durante il quale è $a+an = 0$: in uscita dal circuito si nota un picco basso sul segnale *y*, che dovrebbe rimanere sempre alto.

Per evitare l'alea occorre evitare le transizioni simultanee dei due segnali: è quanto si realizza con le tecniche a livelli (asincrone).

Per evitare l'effetto dell'alea si adoperano tecniche sincrone, secondo le quali un segnale a livelli (*y* nell'esempio) produce effetti soltanto in concomitanza di un apposito segnale di sincronizzazione, sfasato rispetto a quello che genera le variazioni del segnale a livelli.

Se ad esempio *y* varia in concomitanza dei fronti del segnale di clock c_1, esso viene preso in esame in corrispondenza dei fronti di un altro clock c_2, sfasato di 90° rispetto al primo.

Ritardi del circuito

- porte AND, OR (tranne quella su c_2): 1
- clock c_1 (=a)
  durata alto: 10
  durata basso: 10
- ritardo simulato da porta AND (clock c_2)
  5 (1/4 del periodo di c_1)

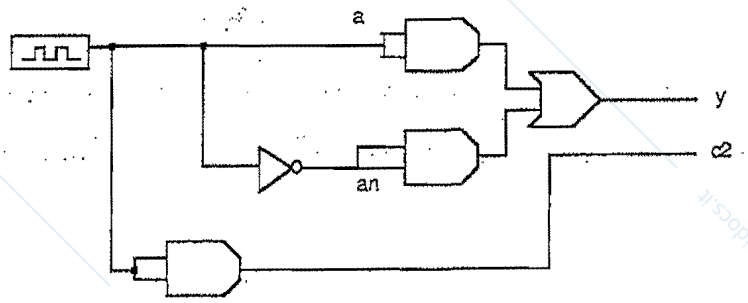


Fig. 2.1a Circuito dimostrativo di un'alea multipla

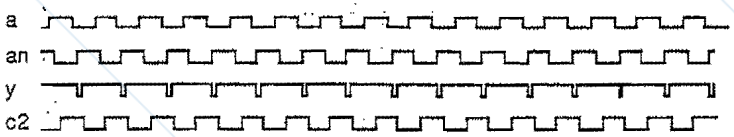


Fig. 2.1b: Alea multipla: tempificazione

3. Alea statica

Tipo di circuito: rete combinatoria
Riferimento: RL, V-5C
Obiettivo: approfondimento teoria

Testo

Si ricorda che in un circuito, allorché gli ingressi variano fra valori adiacenti (quando, cioè, varia una sola variabile), si può avere una condizione di alea statica o dinamica. Nel caso che l'uscita prima e dopo della variazione assuma il medesimo valore, l'alea è detta statica, altrimenti l'alea è dinamica.

Nei circuiti and-or (nand a 2 livelli) l'alea si ottiene se l'uscita del circuito è 1 prima e dopo la variazione (e dualmente per i circuiti nor). L'alea si ha se la transizione non è inclusa in una clausola adoperata per la sintesi della rete e si elimina aggiungendo al circuito la porta rappresentata da detta clausola.

Costruire un circuito esemplificativo di quanto sopra.

Descrizione del circuito (fig. 3.2)

E' esemplificata la funzione:

$$y = a\cdot b + \bar{a}\cdot c$$

rappresentata dalla mappa di Karnaugh di fig. 3.1:

c \ ab	00	01	11	10
0			1	
1	1	1	1	

Fig. 3.1: Mappa di Karnaugh del circuito

L'alea esiste nella transizione di *a* da 1 a 0 o viceversa e si elimina aggiungendo la porta *bc*:

$$y' = ab + \bar{a}c + bc$$

Il circuito consta di 2 parti, l'una per la costruzione di y, l'altra per y'. Due switch fissano $b=c=1$, un ingresso periodico posto su *a* ne simula la variazione. Per il ritardo imposto dall'invertitore, y (che dovrebbe essere sempre 1) raggiunge il valore 0 per un tempuscolo; y' è viceversa costantemente uguale ad 1. I ritardi del circuito sono stati posti a uguali a 2 unità di tempo per evidenziare i fenomeni di alea multipla.

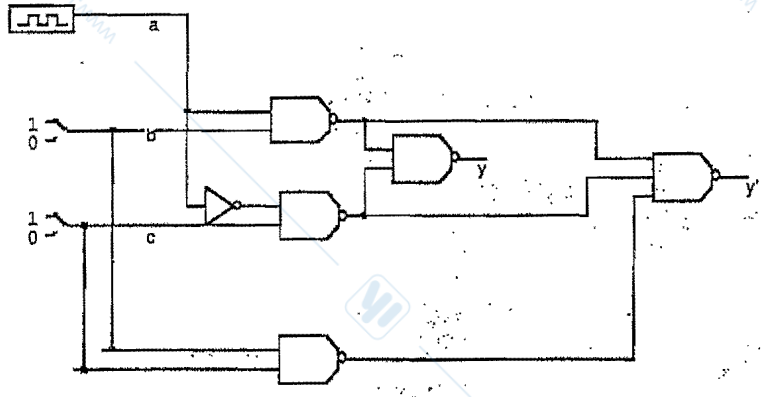


Fig. 3.2a: Alea statica

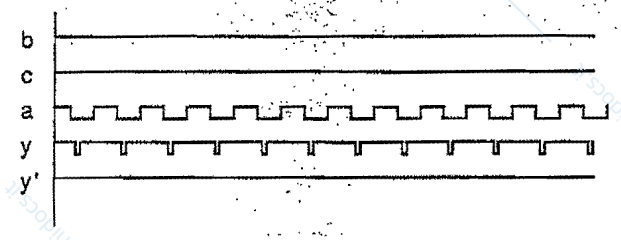


Fig. 3.2b: Alea statica: tempificazione

4. Alea dinamica

Tipo di circuito: rete combinatoria
Riferimento: RL, V-D
Obiettivo: approfondimento teoria

Testo

Si ricorda che in un circuito combinatorio possono aversi alee statiche o dinamiche: nel caso che l'uscita prima e dopo della variazione assuma valori opposti, l'alea è detta dinamica. L'alea dinamica consiste in una oscillazione dell'uscita durante la transizione del segnale d'ingresso: ad esempio, la variazione 1→0 avviene con l'oscillazione 1→0→1→0→1. L'alea si verifica solo per circuiti a più livelli e deriva da un'alea statica presente in una sottorete a 2 livelli; si elimina eliminando l'alea statica.

Costruire un circuito esemplificativo di quanto sopra.

Descrizione del circuito (fig. 4.1)

E' esemplificata la funzione a 3 livelli:

$$z = y \cdot t = (a \cdot b + \bar{a} \cdot c) \cdot (a + \bar{b})$$

Due switch fissano $b=c=1$, un ingresso periodico entra su a, simulandone così la variazione. Per i ritardi dell'invertitore $\bar{a}$ (alea statica) e per il maggiore ritardo ipotizzato della porta OR che costruisce t, z assume l'andamento tipico dell'alea dinamica.

Ritardi

I ritardi del circuito sono stati posti tutti uguali ad 1, tranne quello della porta or che genera il segnale t (6 u.t.) e delle porte not (2 u.t.). La scelta dei tempi di ritardo permette di meglio evidenziare i fenomeni di alea descritti.

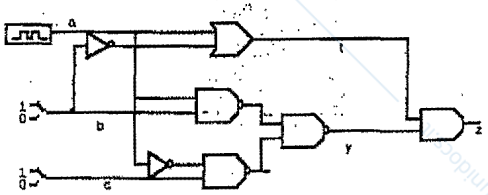


Fig. 4.1a: Alea dinamica

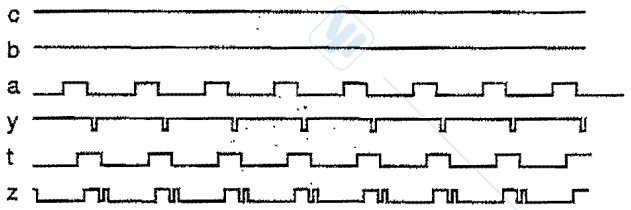


Fig. 4.1b: Alea dinamica: tempificazione

5. Impulsi concomitanti

Tipo di circuito: rete combinatoria
Riferimento: RL, V-5B
Obiettivo: approfondimento teoria

Testo

Due (o più) impulsi concomitanti che operano su un medesimo circuito sono da evitare in quanto provocano un'alea. E', infatti, impossibile in pratica mantenere il sincronismo fra di loro: uno dei due può slittare rispetto all'altro e causare effetti aleatori.

Mostrare l'effetto di due impulsi concomitanti che agiscano su una porta AND e su una OR.

Descrizione del circuito (fig. 5.1)

Sono esemplificate le funzioni:

$$x = c \cdot c; \qquad y = c + c$$

Entrambe dovrebbero essere identicamente uguali a c, come effettivamente avviene per x e y: c è collegato ad un segnale periodico e x, y lo seguono con il ritardo proprio delle porte.

Nella parte di sotto del circuito sono riproposte le stesse porte, che generano x', y'. Le funzioni sono uguali ad x e y ma si è simulato un ritardo di un impulso rispetto all'altro (attraverso i 2 inverter in cascata). Come si può notare, x' è sempre nullo, y' presenta 2 impulsi in concomitanza con c.

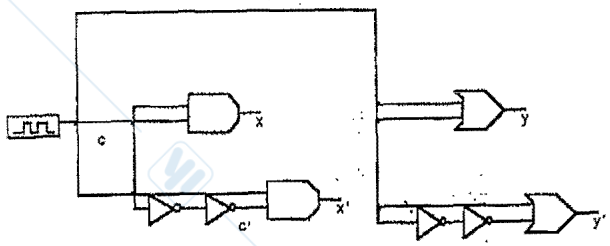


Fig. 5.1a: Impulsi concomitanti

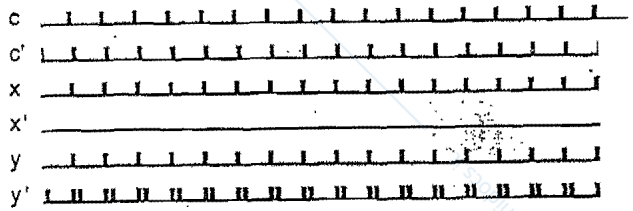


Fig. 5.1b: Impulsi concomitanti: tempificazione

6. Alea essenziale

Tipo di circuito: rete sequenziale asincrona
Riferimento: RL, VII-2
Obiettivo: approfondimento teoria

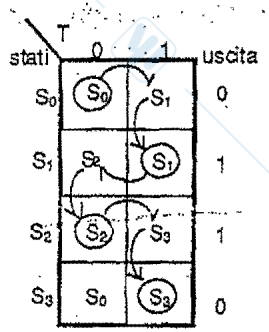
Testo

Una rete asincrona possiede un'alea essenziale se, a partire da un medesimo stato, con una singola variazione dell'ingresso I→I' la rete dovrebbe terminare in uno stato diverso da quello raggiunto con tre variazioni I→I'→I→I'. Costruire una rete esemplificativa.

Scelte di progetto

Si assume come esempio un contatore, considerando l'ingresso quello di conteggio (è bene analizzare questo circuito dopo aver studiato i contatori e il flip-flop T). E' questo un caso classico di alea essenziale, in quanto, ovviamente, la macchina deve contare le variazioni dell'ingresso e quindi 3 variazioni producono un effetto diverso da quella di una sola.

In particolare, si analizza il funzionamento di un contatore asincrono modulo-2 che conta i fronti di salita dell'unico ingresso T (esso è anche un flip-flop T edge-triggered), descritto dalla tabella di stato che segue.



Descrizione del circuito (fig. 6.1)

Il circuito coincide con quello del flip-flop T asincrono di cui al § III.16, ma senza i ritardi sulle linee di reazione che eliminano l'effetto dell'alea essenziale.

Note sulla tempificazione

Si osservi che il contatore, piuttosto che contare: 3-2-0-1....3-2-0-1, conta invece 3-2-0..2-0...2-0...: l'alea trasforma la transizione 0→1 in quella 0→2, corrispondente alle tre variazioni dell'input 0→1→0→1. Osservando con attenzione il display, si scorge che avviene in effetti una transizione a valanga 0→1→3→2. Il fenomeno è ottenuto rendendo lenta la risposta dell'invertitore che genera $Tn = \overline{T}$. Poiché la transizione di Tn ritarda, si susseguono nel tempo gli stati di cui alla tabella.

stato prec.	T	Tn	y'_1	y'_2	stato seg.	tabella	note
0	0	1	0	0	0	riga 1, col 1	stato stabile di partenza
0	1	1	0	1	1	riga 1, col 2	Tn non varia ancora
1	1	1	1	1	3	riga 2, col 1	y'_1 dipende da T, y'_2 da Tn; non completa la transizione e torna indietro nella tabella
3	1	1	1	1	3	riga 3, col 1	come se stesse in 3 stabile
3	1	0	1	0	2	riga 3, col 2	assume la variazione di T come se fosse nello stato 3
2	1	0	1	0	2	riga 4, col 2	e finisce nello stato 2

Ciò avviene perché le linee di reazione sono rapide rispetto ai tempi di ritardo del circuito combinatorio. Se si apponesse su queste un opportuno ritardo, l'alea essenziale non produrrebbe effetti (si veda § III-16). Il ritardo sulle linee di reazione deve essere in generale maggiore di quello della rete combinatoria; nel caso specifico, è sufficiente che sia maggiore di quello della porta or.

Ritardi

Per rendere evidente il fenomeno, si sono posti tutti i ritardi pari a 5, tranne l'invertitore di Tn che ha un ritardo pari a 20.

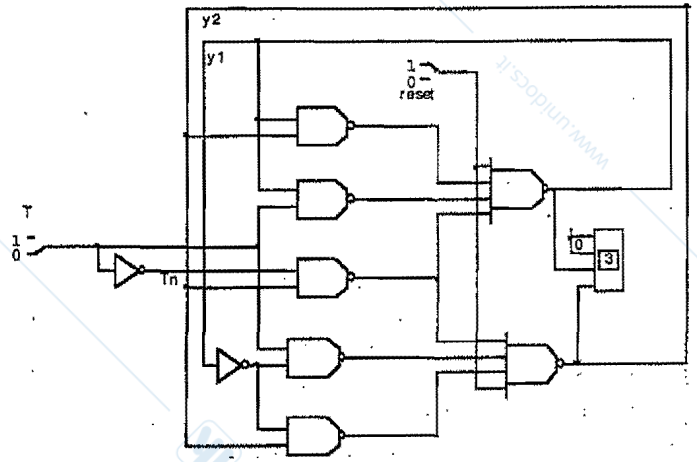


Fig. 6.1a: Alea essenziale

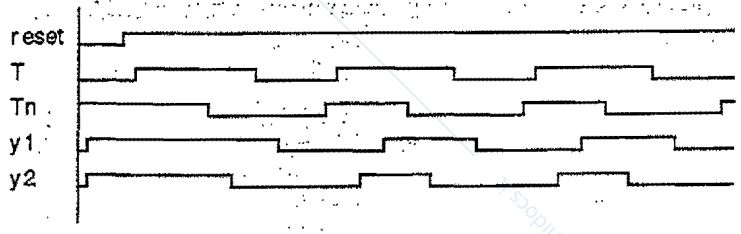


Fig. 6.1b: Alea essenziale: tempificazione

Capitolo terzo
Flip flop

1. Richiami teorici

I flip-flop, organi elementari di memoria, sono i componenti fondamentali delle reti logiche. Essi memorizzano un bit di informazione nei due valori convenzionali di 0 ed 1; un flip-flop si dice anche in *reset* se memorizza 0, in *set* se memorizza 1.

Un flip-flop "a memorizzazione dell'ingresso" possiede in generale almeno 3 distinti stati di ingresso:

- stato di *reset*, che pone il flip-flop in reset,
- stato di *set*, che pone il flip-flop in set,
- stato *neutro*, che lascia inalterato il bit memorizzato,

che vengono codificati attraverso due o più variabili binarie.

Dal punto di vista del comportamento esterno, i flip-flop si possono classificare in vario modo a seconda dei parametri presi in esame (in quanto segue si assumono tutti i segnali 1-attivi; vale la dualità per segnali 0-attivi).

a) Codifica degli ingressi

- *RS* o flip-flop fondamentale, con i 2 ingressi binari R per il reset e S per il set e con la condizione di vincolo R·S=0;
- *D*, con un "ingresso-dati" D che definisce il valore che il flip-flop deve assumere;
- *T*, con un ingresso "trigger" T, che, se attivo, provoca la commutazione del flip-flop;
- *JK*, che si comporta come un flip-flop T per J=K=1, come un RS altrimenti (con J corrispondente a S, K a R);
- *misti* (D+R, JK+RS, etc.).

b) Sincronizzazione degli ingressi

-*Sincroni (o abilitati)*, se il posizionamento del flip-flop avviene in sincronismo con un apposito segnale di "abilitazione" o " sincronizzazione" , detto anche impropriamente *clock;* sono necessariamente sincroni i flip-flop D e JK, nel senso che possono funzionare solo con tale tipo di posizionamento.
-*Asincroni* altrimenti: è tipicamente asincrono il flip-flop RS fondamentale.
Si noti che il flip-flop RS esiste anche in una versione sincronizzata; anche il flip-flop T esiste nelle 2 versioni sincrona (il T definisce se commutare, l'abilitazione quando) e asincrona (il flip-flop commuta sul fronte dell'unico segnale T).

c) Tempificazione.
I flip-flop sincroni possono essere:
- *latch*, se seguono i valori dei segnali di posizionamento durante tutto il tempo in cui l'abilitazione è attiva;
- *edge triggered*, se sono sensibili ai segnali di posizionamento solo sul fronte della salita (o discesa) del segnale di sincronismo;
- *master-slave*, se si posizionano sul secondo fronte dell'impulso di sincronizzazione al valore definito dagli ingressi in corrispondenza del primo fronte.

I flip-flop sono reti sequenziali e in quanto tali essi possono essere:
- *reti fondamentali o asincrone*; lo sono il flip-flop RS fondamentale, i flip-flop edge-triggered e il flip-flop T non sincronizzato (è un particolare edge-triggered, in quanto commuta sul fronte di T);
- *reti sincrone impulsive*: lo sono i flip-flop sincronizzati latch.

Un caso particolare è costituito dal flip-flop master-slave che esiste in due versioni diverse:
- con la condizione di vincolo che gli ingressi non varino durante la presenza dell'impulso di sincronizzazione: è realizzato allora mediante una rete sincrona (vedi esempi nel seguito del capitolo);
- senza la suddetta condizione ed è allora una rete asincrona.

La sopra elencata varietà di ingressi, tempificazione e tecniche di realizzazione genera una miriade di flip-flop diversi. Nel presente capitolo viene sviluppata una ampia e sistematica ma non esaustiva casistica di soluzioni possibili. Esse sono sviluppate con due obiettivi: l'uno è un approfondimento ed esemplificazione della teoria dei flip-flop, l'altro la progettazione di reti sequenziali (il flip-flop viene allora visto come caso tipico di rete da progettare).

2. Il flip-flop RS

RS fondamentale

Il flip-flop RS fondamentale è definito dalla tabella asincrona di figura 2.1: in essa è mostrata la transizione fra lo stato S_0 ed S_1 quando l'ingresso passa dallo stato neutro R=S=0 a quello di set. Si notino i punti di indifferenza in corrispondenza dell'ingresso non specificato R=S=1.

stati \ RS	00	01	11	10	uscita
S_0	S_0	S_1	–	S_0	0
S_1	S_1	S_1	–	S_0	1

Fig.2.1: Flip-flop RS: tabella di definizione

Detto F' il valore seguente dell'uscita ed F quello precedente, il comportamento del flip-flop si può anche esprimere attraverso la sua *equazione di stato:*

$$F = S + F\cdot\overline{R}$$

che si legge; "l'uscita è alta se vi è il set oppure se era alta precedentemente e non vi è il reset".

L'equazione di stato del flip-flop potrebbe anche essere quella del circuito per la sua realizzazione: a tale risultato si perverrebbe se si sintetizzasse la tabella con la scelta di un'unica variabile di stato, appunto F. Tuttavia, il flip-flop che ne verrebbe fuori, detto *flip-flop RS dinamico,* non presenta alcun vantaggio rispetto alla realizzazione che segue, di più complessa concezione ma di più agevole ed economica costruzione.

Il flip-flop a NOR (NAND) incrociati (fig. 2.2)

La realizzazione classica del flip-flop RS fondamentale è quello a NOR incrociati se con set e reset 1-attivi, a NAND incrociati se 0-attivi, mostrato nella figura 2.2. I segnali di posizionamento sono realizzati nella simulazione attraverso 2 monostabili, in modo da mantenere sempre verificata la condizione di indifferenza:

S·R = 0 per quello a NOR, S+R =1 per quello a NAND.

Allorché si attiva il switch posto a monte del monostabile, si genera sul set (o sul reset) un impulso dalle caratteristiche temporali definite (nel caso specifico, dopo un ritardo di 1 unità di tempo, nasce un impulso lungo 10 u.t.). Si è supposto che ciascuna porta logica abbia un ritardo di 4 u.t.

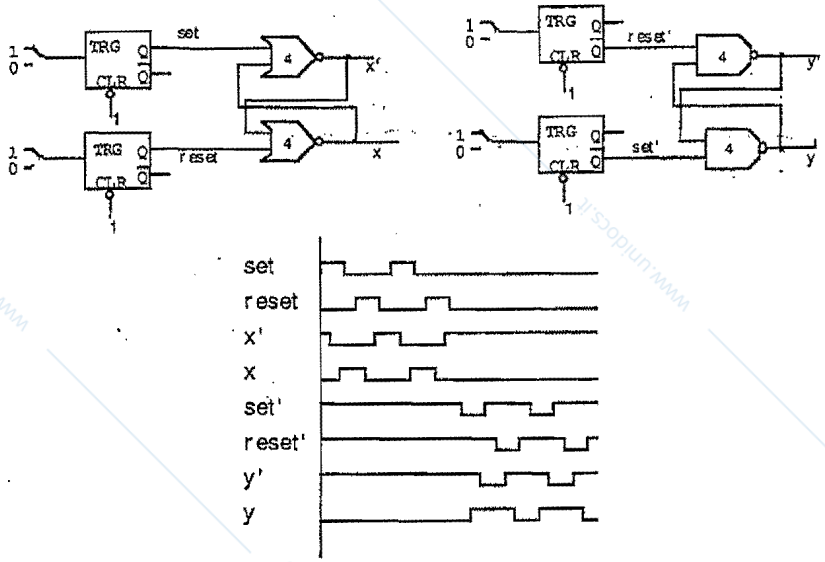


Fig. 2.2: Flip flop RS fondamentale: circuito e simulazione

Analisi del circuito

L'analisi è sviluppata sul flip-flop a NOR; essa si estende a quello a NAND per dualità. Partendo dal flip-flop nello stato di reset ed alzandosi il set, si abbassa dopo 4 unità di tempo il lato falso del flip-flop (uscita x'), che pertanto si porta con entrambe le uscite al valore 0. Permanendo ora il set (se non permanesse si avrebbe un comportamento anomalo, esaminato al § 4), la porta NOR superiore possiede entrambe gli ingressi 0 e quindi, con il ritardo di altre 4 u.t., l'uscita x si porta ad 1 e vi permane anche se viene meno il segnale di set.

Dall'analisi di cui sopra e dall'osservazione del circuito deriva la tabella di stato, basata su un circuito con 2 variabili di stato (x, x') e 4 stati di fig. 2.3a.

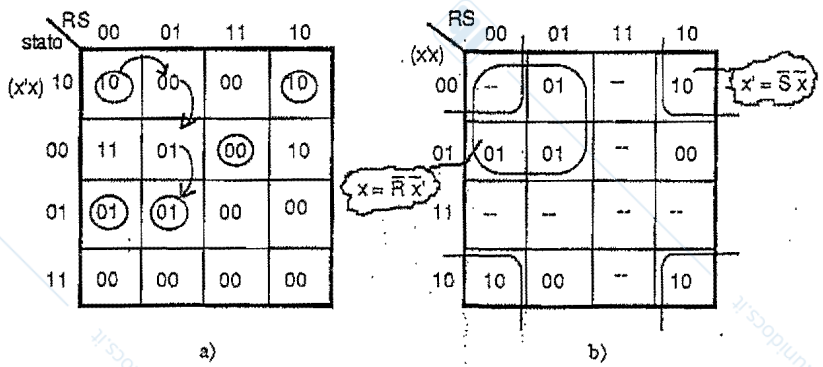


Fig. 2.3: Flip flop RS a NOR: a) tabella di analisi; b) tabella di sintesi

Nella tabella è posta in evidenza la transizione di cui sopra: su di essa la macchina si sposta dapprima in orizzontale e poi in verticale, compiendo un ciclo di 2 transizioni fra gli stati:

10→00→01

Sintesi del circuito

Il circuito di fig. 2.2 deriva da un progetto nel quale:

a) Si effettua la scelta di codificare i 2 stati della tabella iniziale (fig. 2.1) con 2 variabili di stato (invece di 1 sola): lo stato set con 01 e quello reset con 10. Si noti che in generale la minimizzazione delle variabili di stato non conduce necessariamente alla minimizzazione del circuito: nel caso specifico la minimizzazione delle variabili (1 sola) condurrebbe al flip-flop dinamico, di costo maggiore e con problemi di carico elettronico.

b) Al fine di evitare la corsa nella transizione 01→10 e viceversa, si impone la transizione attraverso lo stato (instabile) 00: 01→00→10.

c) Lo stato 11 che ne risulta è non specificato, così come il punto (00, 00). La tabella per la sintesi è mostrata in figura 2.3b).

d) Dal progetto combinatorio scaturisce il circuito a 2 NOR, direttamente ricavato dalle tabelle di sintesi. Si noti che i punti di non specificazione risultano a posteriori fissati ai valori di cui alla tabella di analisi (fig. 2.3a).

Flip-flop RS sincronizzato (fig. 2.4)

Il flip-flop RS sincronizzato latch deriva banalmente da quello fondamentale; basta porre:

$$set = S \cdot a \qquad reset = R \cdot a$$

In tal modo R ed S sono segnali a livelli che definiscono il nuovo stato del flip-flop, ma soltanto in corrispondenza di a il set e reset diventano attivi.

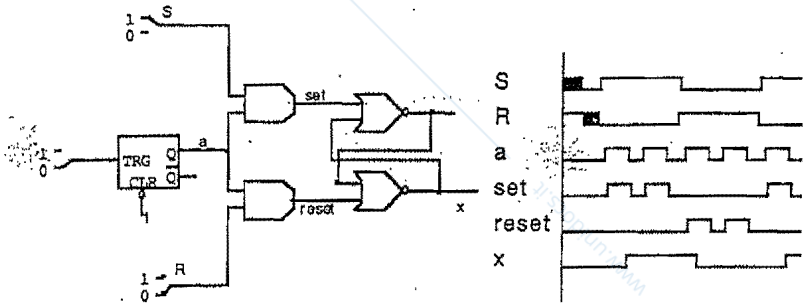


Fig. 2.4: RS sincronizzato

Il flip-flop diventa allora una macchina sincrona impulsiva a sincronizzazione esterna sull'impulso a. La relativa tabella è ancora quella di fig. 2.1, ma intesa come tabella sincrona: ogni transizione avviene soltanto in corrispondenza dell'impulso a. Nei paragrafi che seguono viene approfondita la tempificazione del flip-flop fondamentale e si illustrano i flip-flop edge-triggered e master-slave.

3. RS fondamentale: punti di indifferenza

Tipo di circuito: rete sequenziale asincrona
Obiettivo: analisi del flip-flop; comportamento di una rete se si applicano agli ingressi valori considerati come punti di indifferenza in fase di progetto

Testo

Nel flip-flop a NOR è non definito l'ingresso R=S=1; dualmente, per quello a NAND non è definito R=S=0. Cosa succede se si applica agli ingressi il valore non definito?

Analisi del problema

stato (x' x) \ RS	00	01	11	10
10	(10)	00	00	(10)
00	11	01	(00)	10
01	(01)	(01)	00	00
11	00	00	00	00

Fig. 3.1: Oscillazione del flip-flop RS per applicazione di R=S=1

Dalla tabella del flip-flop a NOR (fig. 3.1) risulta che per R=S=1 si ottiene lo stato 00 indipendentemente da quello di partenza e lo stato resta stabile finché è R=S=1. Da qui, se si passa agli ingressi di set (01) o di reset (10) si raggiungono i corrispondenti stati stabili. Se però R e S variassero simultaneamente, portandosi a R=S=0, il flip flop raggiungerebbe la posizione non stabile in cui oscilla fra gli stati 00 e 11.

Descrizione del circuito (fig. 3.2)

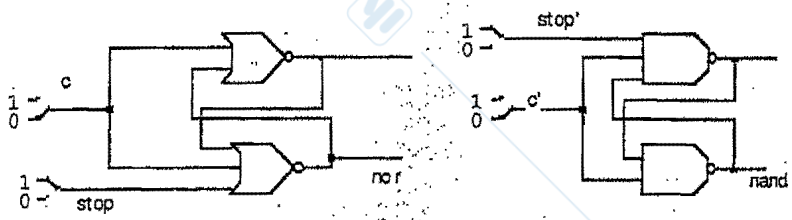


Fig. 3.2a: Instabilità del flip-flop RS sollecitato con ingresso don't care

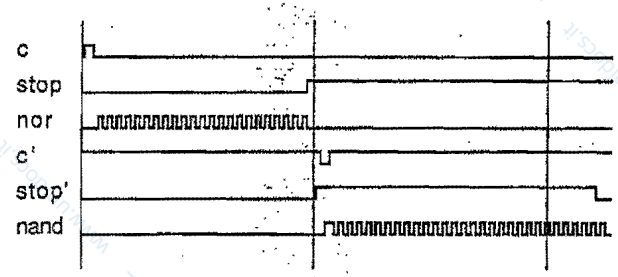


Fig. 3.2b: Instabilità del flip-flop RS sollecitato con ingresso don't care: tempificazione

Per simulare la variazione 11→00 si è posto agli ingressi set e reset del flip-flop un unico segnale c, che attraverso un switch varia da 1 a 0. Per arrestare l'oscillazione che si produce, è adoperato un apposito segnale (stop) che resetta il flip-flop. In figura è mostrata la seguente sequenza:

- c=0, stop=0: il flip flop a NOR è nello stato di reset;
- c=1 stop=0: il flip flop va nello stato 00;
- c=0 stop=0: il flip flop oscilla fra 00 e 11;
- stop=1: il flip flop si resetta fermando l'oscillazione.

Segue poi nel diagramma temporale una sequenza per il flip-flop a NAND:

- c'=1, stop'=0: il flip flop a NAND è nello stato di reset;
- c'=1, stop'=1: il flip flop mantiene lo stato di reset;
- c'=0 stop'=1: il flip flop va nello stato 11;
- c'=1 stop'=1: il flip flop oscilla fra 11 e 00;
- stop'=0: il flip flop si resetta fermando l'oscillazione.

4. RS fondamentale: timing

Tipo di circuito: rete sequenziale asincrona
Obiettivo: tempificazione reti asincrone

Testo

Si ricorda che la durata di un segnale di ingresso in una rete asincrona deve essere maggiore del ritardo complessivo della rete combinatoria:

$$d > R$$

In generale, anzi, se la transizione fra gli stati imposta dalla variazione degli ingressi avviene con un ciclo di lunghezza k, deve aversi:

$$d > kR$$

Analizzare tale relazione per il flip-flop RS fondamentale.

Analisi del problema

Dall'analisi compiuta al § 2 (fig. 2.3) si evince che la rete compie un ciclo di 2 transizioni nel passaggio dallo stato "reset" a quello "set" o viceversa (01→00→10); la durata del segnale di ingresso deve dunque essere:

$$d > 2R$$

Per segnali di durata inferiore, il flip-flop entra in oscillazione, come si può evincere dall'analisi della tabella delle transizioni di fig. 4.1.

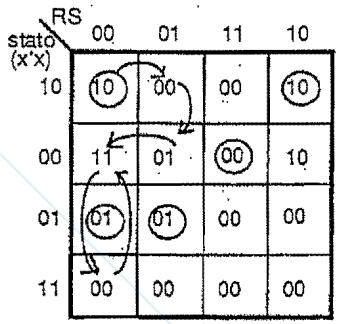


Fig. 4.1: Instabilità del flip-flop RS per insufficiente durata dell'impulso

Descrizione del circuito (fig. 4.2)

In figura è mostrato un circuito esemplificativo che consta di 2 flip-flop RS di tipo latch sincronizzati da un segnale di abilitazione ("clock"). Ciascuno dei due, al tempo dettato da un monostabile, dovrebbe essere settato o resettato a seconda della posizione del switch *dato* (il flip-flop può anche essere visto come uno di tipo D, cfr.§ 8). Peraltro, dei due monostabili, l'uno genera un impulso che rispecchia la condizione di durata *(clock ok)*, l'altro no *(clock corto)* e quindi il primo flip-flop opera correttamente, l'altro no.

Ritardi del circuito

- porte AND: 1 u.t. (inessenziale per quanto riguarda questa prova);
- porte NOR (ritardo R): 2 u.t.;
- ritardo monostabili: 15 u.t.;
- ampiezza impulso monostabili:
  - *clock ok"*: d=5 (>2R);
  - *clock corto*: d=3 (< 2R);
  - il caso d=4 sarebbe critico: il simulatore lo prende per buono.

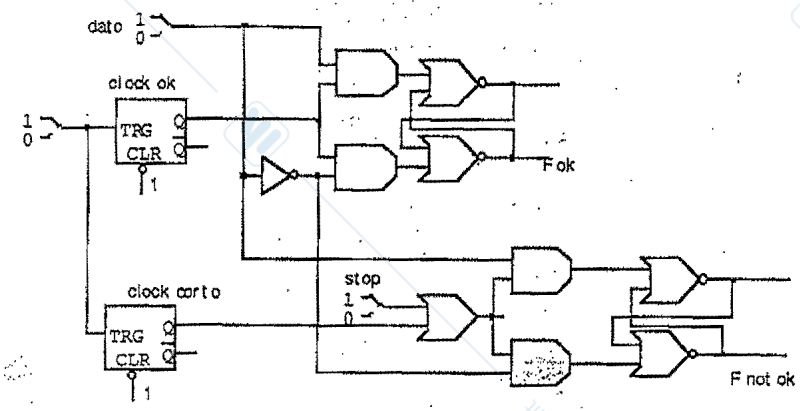


Fig. 4.2a: Flip-flop RS latch

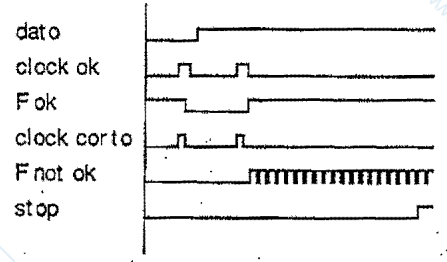


Fig. 4.2b: Tempificazione del flip-flop RS latch

5. RS edge triggered (master-slave)

Tipo di circuito: rete sequenziale asincrona
Riferimento: cfr. § 10
Obiettivo: studio di flip-flop

5.1 RS sul fronte di salita

Testo

Progettare un flip-flop RS edge-triggered sul fronte di salita.

Progetto

Si tratta ovviamente di un flip-flop abilitato che commuta (se del caso) sul fronte di salita del segnale di abilitazione *a*.

Non si sviluppa qui un progetto autonomo, ma si fa riferimento al progetto del flip-flop D edge-triggered realizzato con 2 RS fondamentali posti in serie (cfr. § 10). Il progetto del flip-flop RS, infatti, equivale banalmente a quello del flip flop D nel quale si ponga S=D, R= $\overline{D}$. Gli ingressi per i due flip-flop componenti sono:

$$S_p = \bar{a} \cdot S \qquad R_p = \bar{a} \cdot R$$
$$S_u = a \cdot p \qquad R_u = a \cdot \bar{p}$$

In ogni caso, si riporta la tabella del flip-flop RS edge-triggered, che è alla base del suo progetto sviluppato per vie autonome, sintetizzando la quale si perviene al medesimo risultato di cui sopra. Si noti che le colonne tratteggiate coincidono ordinatamente con quelle del flip-flop D, fatti salvi alcuni punti di indifferenza.

		a=0				a=1			
	SR	00	01	11	10	00	10	11	01
		000	001	011	010	100	110	111	101
stati	00	00	00	–	01	00	00	–	00
	01	01	00	–	01	01	11	–	01
	11	11	10	–	11	11	11	–	11
	10	10	10	–	11	10	10	–	00

Fig. 5.1 : Mappa del flip-flop RS edge triggered

Descrizione del circuito (fig. 5.2)

Si veda il § 10.

Tempificazione

Si noti che se per *a=1* si avesse in ingresso *R=S=1*, il flip-flop entrerebbe in oscillazione, così come quello fondamentale (cfr. § 3).

La tempificazione del flip-flop nei riguardi di a, tipica dei flip-flop edge, è mostrata in figura 5.2b: per $a=1$ nulla si muove all'interno del circuito, per $a=0$ il primo flip-flop (p) segue i valori di R, S e con la variazione $0\rightarrow1$ del fronte di a, p è ricopiato in u.

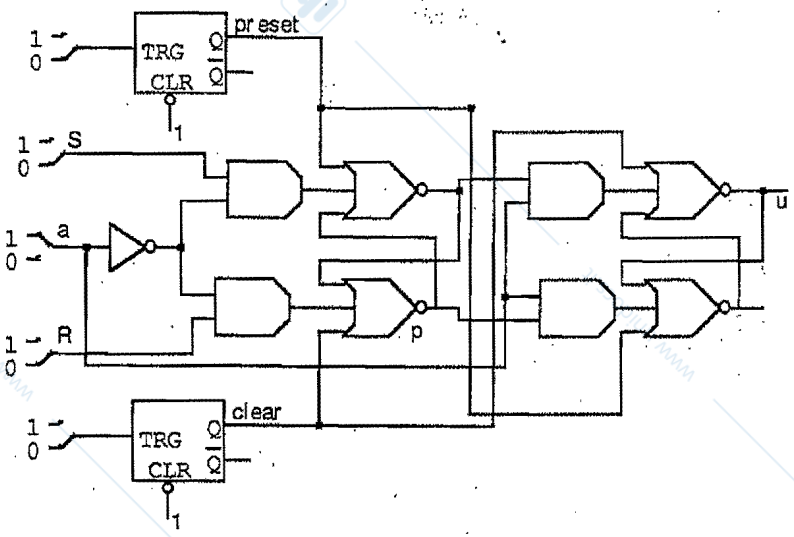


Fig. 5.2a: Flip-flop RS edge triggered sul fronte di salita

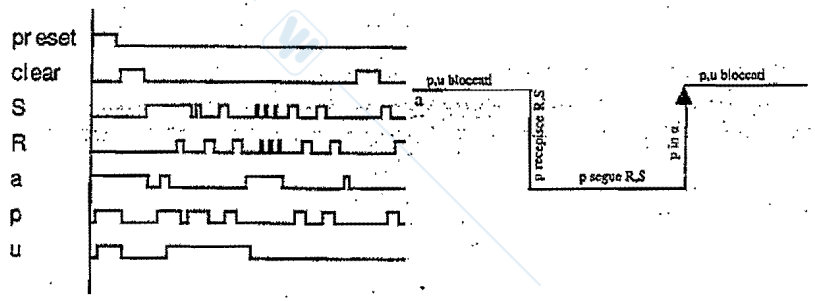


Fig. 5.2b: Flip-flop RS edge triggered sul fronte di salita: tempificazione

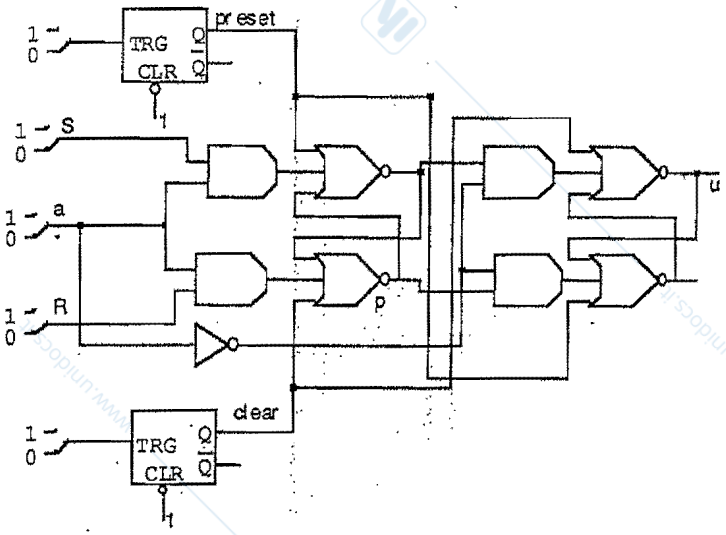


Fig. 5.3a: RS edge triggered su fronte di discesa

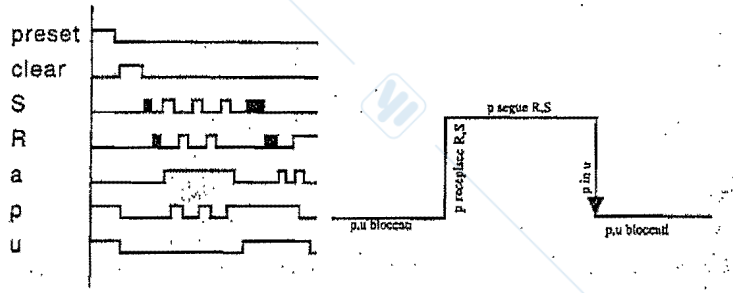


Fig. 5.3b RS edge triggered su fronte di discesa

5.2 RS sul fronte di discesa

Testo

Progettare un flip-flop RS edge-triggered sul fronte di discesa.

Progetto e tempificazione

Il circuito è del tutto analogo a quello già visto, fatta salva la diversa disposizione dell'invertitore su a. La tempificazione, duale di quella del flip-flop a fronte di salita, è mostrata in figura 5.3b.

5.3 RS master-slave

Per il flip-flop a fronte di discesa (e dualmente per quello a fronte di salita), se R e S non variano per $a=1$, il flip-flop edge triggered coincide con il flip-flop master-slave che acquisisce il dato sul fronte $0\rightarrow1$ (di salita, primo fronte di un impulso) e lo restituisce all'uscita su quello $1\rightarrow0$ (di discesa, secondo fronte). Si tratta, ovviamente, del flip-flop master-slave definito sincrono al § 1. Per ulteriori approfondimenti si veda il § 9 (flip-flop D sincrono con RS edge).

6. Il flip-flop D

Il flip-flop D è una macchina che memorizza il dato D in sincronismo con un segnale di abilitazione a; si tratta quindi di un flip-flop con le due variabili di ingresso D e a. In figura 6.1a) è mostrata la tabella fondamentale asincrona del flip-flop, con evidenziata la transizione $S_0\rightarrow S_1$. Dalla sintesi di questa tabella, condotta come quella del flip-flop RS, si ottiene il medesimo fllip-flop a NOR (NAND) incrociati, con:

$R = a\cdot\overline{D}$ $\quad$ $S = a\cdot D$

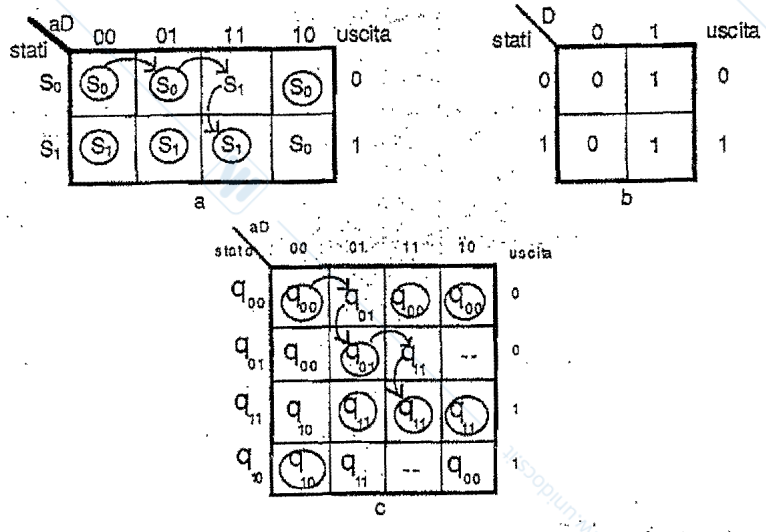


Fig. 6.1: Flip-flop D. a: tabella fondamentale; b: tabella sincrona; c: tabella edge-triggered

Alle stesse conclusioni si perviene più semplicemente considerando che un flip-flop RS, con: $R=a\cdot\overline{D}$, $S=a\cdot D$ si comporta esattamente come un flip-flop D.

In figura 6.1b è riportata la tabella sincrona del flip-flop: stavolta la macchina è considerata a sincronizzazione esterna, con impulso di sincronismo a non riportato in tabella . Realizzando la macchina con l'uso di un flip-flop RS sincronizzato sulle linee di reazione, si ottiene dalla tabella di stato $R=\overline{D}$, $S=D$ ed essendo nel flip-flop RS sincronizzato il set e il reset in and con a, si ottiene ancora $R=a\cdot\overline{D}$, $S=a\cdot D$.

In fig. 6.1c è riportata la tabella che definisce il comportamento del flip-flop edge triggered sul fronte di salita di a: su di essa è evidenziata la transizione fra gli stati $q_{00}\rightarrow q_{01}\rightarrow q_{11}$ in conseguenza del fatto che, a partire da $a=D=0$, si alzi prima D e poi a. Si tratta di una macchina asincrona o fondamentale (nel senso della teoria delle reti sequenziali), sincronizzata dal fronte del segnale a (nel senso della terminologia usata per esprimere il comportamento dei flip-flop).

Nei paragrafi che seguono è approfondita la tempificazione e la realizzazione concreta del flip-flop.

7. D latch dinamico

Tipo di circuito: rete sequenziale asincrona
Riferimento: prodotto commerciale 7475
Obiettivo: studio di flip-flop; alee statiche

Testo

Progettare un flip-flop D latch.

Progetto

Si tratta di una rete asincrona banale. Dalla sintesi della tabella di figura 6.1a, avendo assunto F come variabile di stato, risulta:

$$F = \overline{a}\cdot F + a\cdot D + D\cdot F$$

ove il termine D·F è aggiunto per evitare l'alea statica. Un flip-flop siffatto si dice "dinamico", in analogia con il flip-flop RS dinamico.

Descrizione del circuito (fig. 7.1)

-uno switch simula D;
-un monostabile simula il segnale di sincronismo;
- F è calcolato con una NOR invertita (per problemi di carico cfr § 1); immediatamente a valle della NOR è prelevata l'uscita Fn (Fnegato).

Il circuito presenta due versioni del flip-flop, l'una corretta (con il termine D·F), l'altra no. Si può notare come il flip-flop scorretto (F not ok) oscilla, per l'alea statica, quando il flip-flop è in set e gli perviene un nuovo segnale per il quale dovrebbe rimanere in set. Il switch *stop* arresta detta oscillazione.

Lo schema corrisponde a quello dei prodotti commerciali 7475.

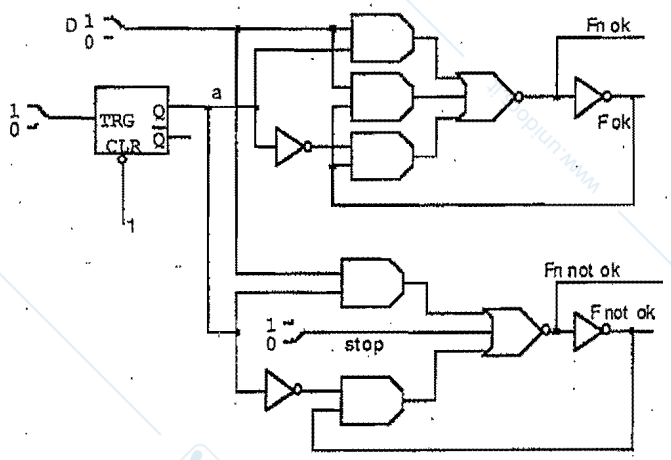


Fig. 7.1a: D latch dinamico

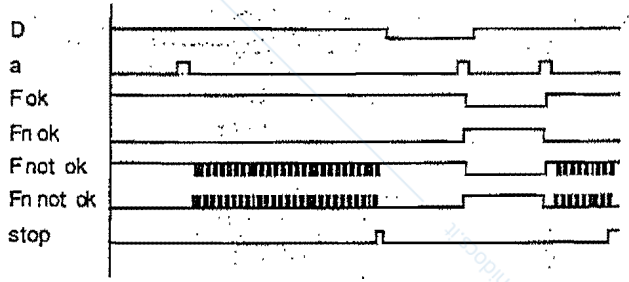


Fig. 7.1b: D latch dinamico: diagramma di tempificazione

8. D sincrono con RS latch

Tipo di circuito: rete sequenziale sincrona
Riferimento: MEI, VIII-3; circuito commerciale 7475
Obiettivo: studio di flip-flop

Testo

Studiare la tempificazione del flip-flop D sincrono con RS latch su linee di reazione.

Descrizione del circuito (fig. 8.1)

Il circuito è quello che deriva da quanto esposto al § 6: un flip-flop RS i cui ingressi sono pilotati attraverso una rete combinatoria dai segnali *D* (dato in ingresso) e *a* (abilitazione), con:

$$R = a \cdot \overline{D} \qquad S = a \cdot D$$

Il circuito mostrato, corrispondente a quello del flip-flop commerciale 7475, presenta sulle linee di reazione due porte AND con pura funzione di ritardo, che rendono uguali i tempi di risposta nelle variazioni 0→1 e 1→0.

Essendo la macchina sincrona e quindi impulsiva, *D* non può variare durante il tempo in cui è *a=1*. Per tale motivo il circuito è stato simulato con un monostabile sull'ingresso *a* in modo da evitare che si possa far variare *D* con *a=1*. Si noti che il circuito risultante è comunque quello di un flip-flop D latch e quindi seguirebbe il dato per *a=1*. Ciò potrebbe essere verificato ponendo un switch invece che un monostabile sull'ingresso *a*.

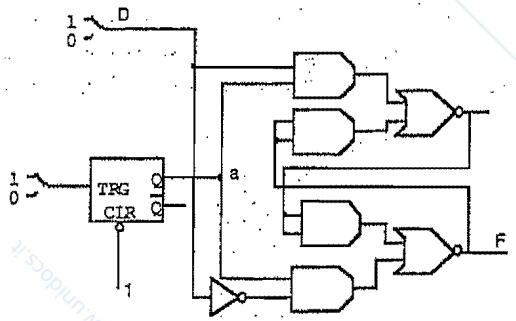


Fig. 8.1a: D sincrono con RS latch (D latch)

Tempificazione

Per evitare fenomeni di oscillazione del flip-flop RS, la durata dell'impulso *a* deve essere maggiore di 2R, ove R è il ritardo combinatorio del flip-flop RS (cfr. § 4). Essendo il ritardo di ciascuna porta unitario si ha R=2 u.t., e d> 4 u.t.. L'impulso del monostabile è dunque di durata 5 u.t..

Fig. 8.1b: D sincrono con RS latch (D latch): tempificazione

9. D sincrono con RS edge (master-slave)

Tipo di circuito: rete sequenziale sincrona
Riferimento: MEI, VIII-4
Obiettivo: studio di flip-flop

Testo

Progettare un flip-flop D sincrono con RS edge su linee di reazione (l'ipotesi di D sincrono equivale all'ipotesi che D non vari durante la fase attiva dell'impulso).

Funzioni di posizionamento

Le funzioni di posizionamento sono quelle già viste per il flip-flop latch:

$$R = a \cdot \overline{D} \qquad S = a \cdot D$$

9.1 D sul fronte di salita

Descrizione del circuito (fig. 9.1)

E' costituito dal flip-flop RS edge sul fronte di salita con i segnali di posizionamento di cui sopra.

Per porre in evidenza il comportamento impulsivo della macchina, il circuito è stato simulato con un monostabile sull'ingresso *a* in modo che D non possa variare durante l'impulso.

Sono state simulati due distinti impulsi: 1-attivo (variazione 0→1→0) e 0-attivo (1→0→1). A tale scopo si è posto $a = Q \oplus ctrl$, ove *ctrl* è un segnale di controllo. In tal modo per $ctrl = 0$ si ha $a = Q$ (*a* è normalmente 0 e diventa 1 quando l'impulso è attivo), per $ctrl = 1$ si ha $a = \overline{Q}$ (normalmente 1 e diventa 0 quando l'impulso è attivo).

Tempificazione

Per evitare fenomeni di oscillazione del flip-flop RS, la durata *d* dell'impulso *a* deve essere maggiore di 2R, ove R è il ritardo combinatorio del flip-flop RS (cfr. § 4). Per meglio evidenziare la tempificazione è stato posto *d=10*. La tempificazione del flip-flop nei riguardi del segnale a è mostrata in figura 9.1b. Si noti che:
- per $a=0$, *p* segue D;
- sul fronte 0→1, *p* va in *u*, realizzando il nuovo valore del flip-flop;
- per $a=1$ permane la situazione del fronte (*D* non varia, ma anche se variasse nulla accadrebbe).

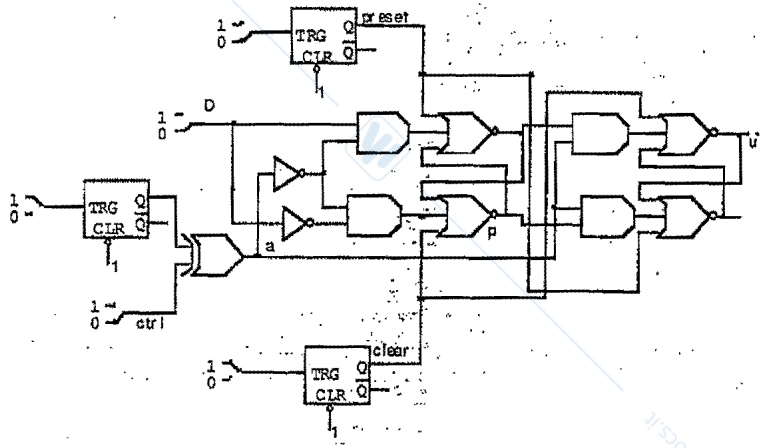


Fig. 9.1a: D sincrono impulsivo con RS edge sul fronte di salita

Quindi, per l'impulso 1-attivo si ha (fig. 9.1b):
- in assenza di impulso, *p* segue *D*;
- al primo fronte di *a* (0→1), il *p* così posizionato va sull'uscita *u*;
- per $a=1$ tutto resta invariato.

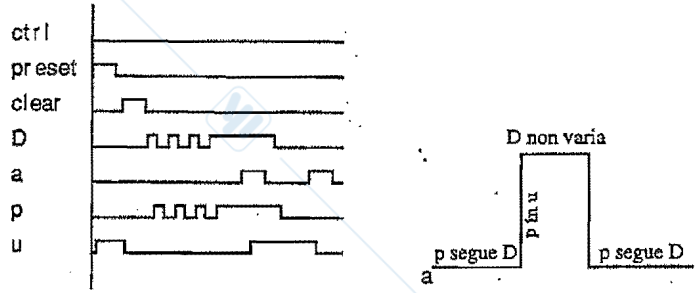


Fig. 9.1b: D edge sul fronte di salita ed impulso 1-attivo: tempificazione

Per l'impulso 0-attivo si ha invece (fig. 9.1c):
- in assenza di impulso (*a=1*), *p* ed *u* restano bloccati;
- sul primo fronte di *a* (1→0), *D* va in *p;*
- per *a=0*, *D* non varia e dunque nulla accade;
- sul secondo fronte di *a* (0→1), p è ricopiato sull'uscita *u*.

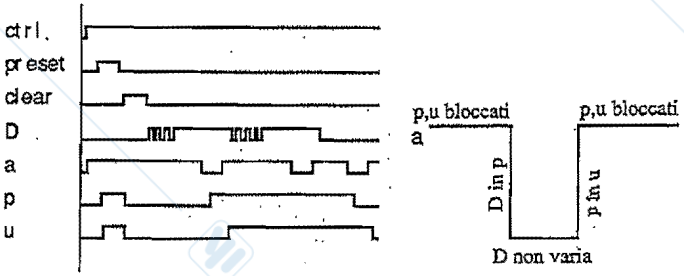


Fig. 9.1c: D edge sul fronte di salita ed impulso 0-attivo: tempificazione

Il comportamento del circuito è dunque quello di un flip-flop master-slave (di tipo sincrono, cfr. § 1) con impulso 0-attivo: acquisisce sul fronte 1→0 e restituisce in uscita sul fronte 0→1.

Confronto fra flip-flop master-slave e edge-triggered

Si noti che il circuito con impulso 0-attivo (master-slave) coincide con quello del flip-flop edge triggered sul fronte di salita (cfr. § 10); se D variasse per a=0, p lo seguirebbe e il valore trasferito su *u* sarebbe quello del fronte 0→1.

9.2 D sul fronte di discesa (fig. 9.2)

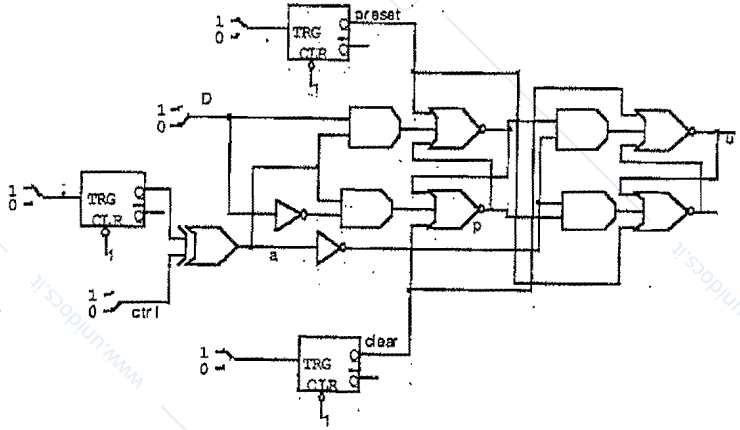


Fig. 9.2a: D sincrono impulsivo con RS edge su fronte di discesa

Del tutto analogo è il circuito con RS su fronte di discesa, fatta salva la inversione dei fronti. Per l'impulso 0-attivo (fig. 9.2b) si ha:
- in assenza di impulso, *p* segue *D;*
- al primo fronte di *a* (1→0), il *p* così posizionato va sull'uscita *u;*
- per *a=0* tutto resta invariato.

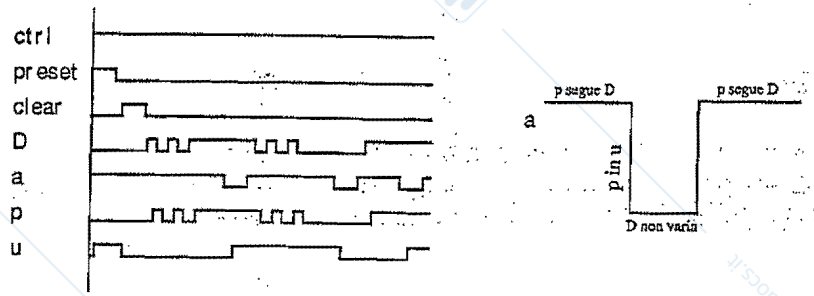


Fig. 9.2b: D edge sul fronte di discesa ed impulso 0-attivo: tempificazione

Per l'impulso 1-attivo (fig. 9.2c) si ha:
- in assenza di impulso (*a=0*), *p* ed *u* restano bloccati;
- sul primo fronte di *a* (0→1), *D* va in *p;*
- per *a=1 D* non varia e dunque nulla accade;
- sul secondo fronte di *a* (1→0), p è ricopiato sull'uscita *u*.

In conclusione, il flip-flop D sincrono con RS edge sul fronte di discesa sulle linee di reazione:
- è un master-slave su impulso 1-attivo;
- coincide con il D edge triggered sul fronte di discesa.

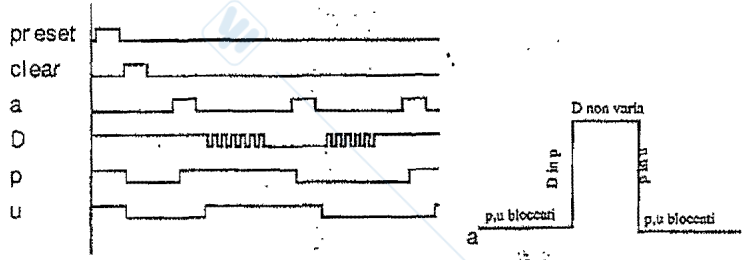


Fig. 9.2c: D sul fronte di discesa ed impulso 1-attivo: tempificazione

10. D edge con 2 RS

Tipo di circuito: rete sequenziale asincrona
Riferimento: MEI, VIII-5
Obiettivo: Progettazione asincrona; studio di flip-flop

Testo

Progettare un flip-flop D edge-triggered sul fronte di salita, che abbia altresì due segnali non abilitati ("asincroni") di preset e clear.

10.1 D sul fronte di salita

Scelta di progetto

Rete asincrona con flip-flop RS su linee di reazione.

Tabella di stato

Tralasciando per il momento i segnali di preset e clear, si assume come tabella di stato quella teorica di fig. 6.1c). Si codificano quindi gli stati con 2 variabili, u e p nell'ordine, come segue: q_{00}=(00), q_{01}=(01), q_{11}=(11) e q_{10}=(10). Ne risulta la tabella per il progetto di fig. 10.1.

stato (u,p) \ aD	00	01	11	10	uscita
q_{00}	00	01	00	00	0
q_{01}	00	01	11	--	0
q_{11}	10	11	11	11	1
q_{10}	10	11	--	00	1

Fig. 10.1: Tabella di stato del flip-flop D edge a fronte di salita

Funzioni di posizionamento

Avendo assegnati gli stati come dalla tabella, ne risulta che il flip-flop u rappresenta anche l'uscita del flip-flop D; occorre dunque progettarne i segnali di posizionamento S_u, R_u (set e reset di u), S_p, R_p (set e reset di p). Si sviluppa quindi il progetto dei segnali set e reset dei flip-flop u e p.

up \ aD	00	01	11	10
00	0	1	0	0
01	0	-	-	-
11	0	-	-	-
10	0	1	-	1

Sp

up \ aD	00	01	11	10
00	-	0	-	-
01	1	0	0	-
11	1	0	0	0
10	-	0	-	-

Rp

up \ aD	00	01	11	10
00	0	0	0	0
01	0	0	1	-
11	-	-	-	-
10	-	-	-	0

Su

up \ aD	00	01	11	10
00	-	-	-	-
01	-	-	0	-
11	0	0	0	0
10	0	0	-	1

Ru

Ne risulta:

$$S_p = \bar{a} \cdot D \qquad R_p = \bar{a} \cdot \bar{D}$$
$$S_u = a \cdot p \qquad R_u = a \cdot \bar{p}$$

I segnali di posizionamento iniziale *preset* (che pone il flip-flop ad 1) e *clear* (che pone il flip-flop a 0) si possono utilmente piazzare direttamente sui set e reset dei flip-flop RS.

Il flip-flop D che ne risulta è in effetti un flip-flop misto, raccogliendo in sé il D propriamente detto ed il flip-flop asincrono RS. Si ha allora:

$S_p = \bar{a} \cdot D + preset$ $\quad R_p = \bar{a} \cdot \bar{D} + clear$

$S_u = a \cdot p + preset$ $\quad R_u = a \cdot \bar{p} + clear$

Si noti che il circuito funziona come segue:
- per $a=0$ viene acquisito in p il valore di D; u (e quindi l'uscita) non varia;
- per $a=1$ il primo flip-flop (p) viene ricopiato nel secondo (u).

Descrizione del circuito (fig. 10.2)

Il circuito risulta composto da 2 flip-flop RS fondamentali, l'uno abilitato da $\bar{a}$, l'altro da a. I segnali di set e preset operano direttamente su entrambi i flip-flop; essi sono simulati con 2 monostabili al fine di evitare che in prova di simulazione si lascino il set o il preset attivi.

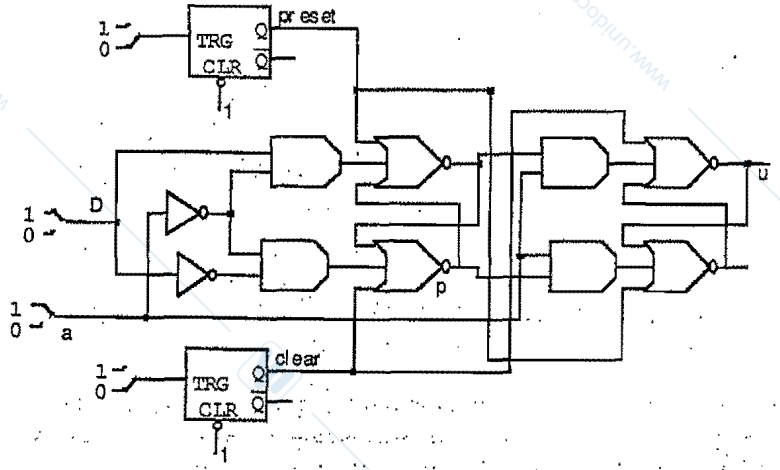


Fig. 10.2a: Flip-flop D edge triggered su fronte di salita

Tempificazione

La durata w di ciascuno dei due livelli del clock deve essere tale da mantenere la commutazione dei flip-flop RS; detto R il ritardo di ciascuna delle porte NOR, deve essere (cfr. § 4):

$w > 2R$

Nel caso specifico, essendo R=1, deve essere w>2 u.t.. La tempificazione presentata in figura mostra come:
- il flip-flop vene preliminarmente posto in set o in reset;
- essendo $a=0$, p segue D ma l'uscita u non varia;
- al fronte 0→1 di a il valore 1 di p viene trasferito in u;
- restando $a=1$, pur variando D, nulla varia né su p né su u;
- essendo di nuovo $a=0$, p segue D, ma l'uscita u non varia;
- al fronte 0→1 di a il valore 0 di p viene trasferito in u.

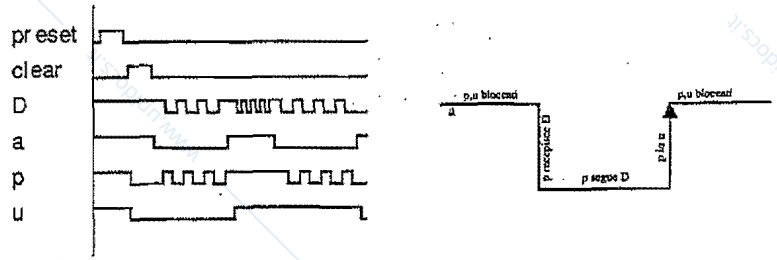


Fig.10.2b: Flip-flop D edge triggered su fronte di salita: tempificazione

Nota

Si noti che con lo stesso schema può essere realizzato un flip-flop RS abilitato edge triggered: è sufficiente sostituire D con S e $\bar{D}$ con R (cfr. § 5).

10.2 D sul fronte di discesa

Analogo è il circuito sul fronte di discesa, che si progetta egualmente e conduce a:

$S_p = a\,D + preset$ $\quad R_p = a\,\bar{D} + clear$

$S_u = \bar{a} \cdot p + preset$ $\quad R_u = \bar{a} \cdot \bar{p} + clear$

Fig. 10.3: Tempificazione del flip-flop D edge triggered su fronte di discesa e flip-flop master-slave

Esso risponde alla tempificazione di cui alla figura 10.3.

Confronto tra flip-flop master-slave ed edge triggered

Si faccia riferimento al flip-flop edge sul fronte di discesa la cui tempificazione è mostrata in figura 10.3a (le osservazioni valgono per dualità per il flip-flop sul fronte di salita).

Per quanto visto al § 9, questo circuito coincide con il flip-flop master-slave ad impulso 1-attivo, nel quale D non vari durante l'impulso (fig. 9.2a, riportata in fig. 10.3b). Coincidono pertanto in un unico circuito i flip-flop:
- D edge su fronte di discesa;
- D master-slave con impulso 1-attivo (e D che non varia durante l'impulso);
- D impulsivo sincrono con RS edge su fronte di discesa.

Si noti che tale coincidenza non è generale: ad esempio, non vale per il flip-flop JK (cfr. § 20).

11. D edge con 3 RS

Tipo di circuito: rete composta
Riferimento: MEI VIII - 5D, circuito commerciale 7474
Obiettivo: studio di flip-flop

Testo

Progettare un flip-flop D edge-triggered sul fronte di salita.

Impostazione del progetto

Il progetto viene sviluppato con una tecnica di progettazione atipica, finalizzata alla realizzazione del flip-flop con un circuito di basso costo, corrispondente a quello del flip-flop commerciale 7474. Nonostante la complessità della tecnica di progetto, il risultato sarà semplice: solo 6 porte nand (rispetto alle 10 del circuito di fig. 10.1) e con caratteristiche di simmetria, particolarmente adatte alla realizzazione dei circuiti integrati.

La prima scelta progettuale è quella di realizzare il circuito attraverso due reti in cascata: un flip-flop di tipo RS, U, nello stadio finale ed a monte una rete C per il suo pilotaggio, come in figura 11.1.

La rete C sarà a sua volta decomposta in 2 reti componenti fra di loro interconnesse, che risulteranno costitute ciascuna da soli flip-flop e senza alcuna porta nella parte combinatoria. Ne risulterà in totale una rete composta di soli 3 flip-flop.

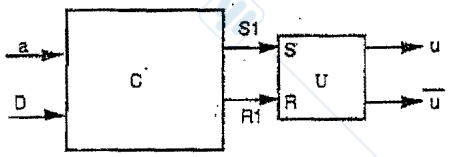


Fig. 11.1: Decomposizione in cascata della rete

Tabella di stato di C

La macchina C è simile a quella del flip-flop D nella sua interezza, ma possiede per uscite non già il valore dell'uscita *u* di tutta la macchina, bensì i segnali di posizionamento del flip-flop dello stadio finale. In figura 11.2a) ne è disegnata la tabella di stato, le cui uscite di Moore indicano gli stati per i quali l'uscita è il set o il reset del flip-flop U (primo e ultimo rigo) e quelli ove essa può essere l'ingresso neutro di U (R_1=S_1=1); questa scelta, consentendo una certa libertà di codifica per gli stati, ottimizzerà il progetto.

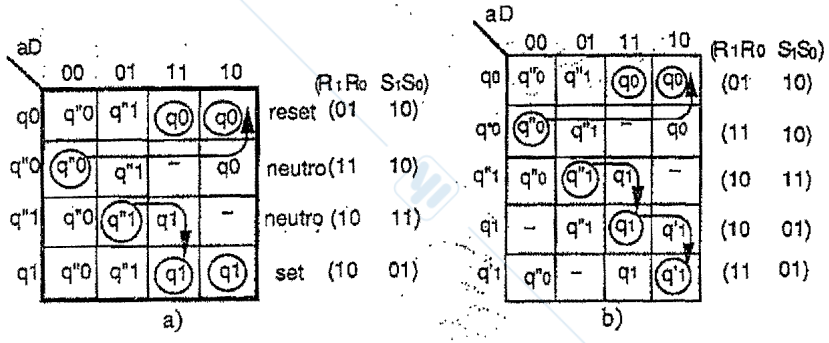


Fig. 11.2a Tabella di stato di C iniziale: b: tabella modificata

In figura 11.3a) è riportato il grafo equivalente alla tabella di figura 11.2a).

Progetto per decomposizione

Si fa riferimento alla tecnica di progetto generale illustrata in dettaglio al capitolo VI. Nel caso specifico, il progetto è finalizzato alla realizzazione della rete C come rete composta da due flip-flop:
- R_1 (R_0 ne è il lato falso), per il reset di U;
- S_1 (S_0 ne è il lato falso), per il set di U.

Dei due flip-flop si adoperano anche gli stati le cui uscite sono $R_1=R_0=1$ ($S_1=S_0=1$) come ulteriore stato neutro per il reset (set) di U. Si adotta quindi per gli stati la codifica segnata ai margini della figura 11.2a). Questa codifica suggerisce la disposizione dei nodi di C secondo il grafo di figura 11.3a) e le conseguenti partizioni:

$$S=(q_0q''_0)(q''_1)(q_1) \text{ e } R=(q_0)(q''_0)(q_1q''_1).$$

La partizione R sulle colonne è associata, al flip-flop R_1-R_0, in quanto ciascuna colonna ha il codice dei primi due bit (R_1, R_0) corrispondente ad uno degli stati 01, 11, 10 del flip-flop; analogamente la partizione S sulle righe è associata al flip-flop S_1S_0.

Nelle figure 11.3b), c) sono indicati rispettivamente i grafi delle partizioni S e C. Purtroppo, nessuna delle due partizioni è chiusa e quindi le reti componenti (i due flip-flop) sono funzione l'una dell'altra e viceversa.

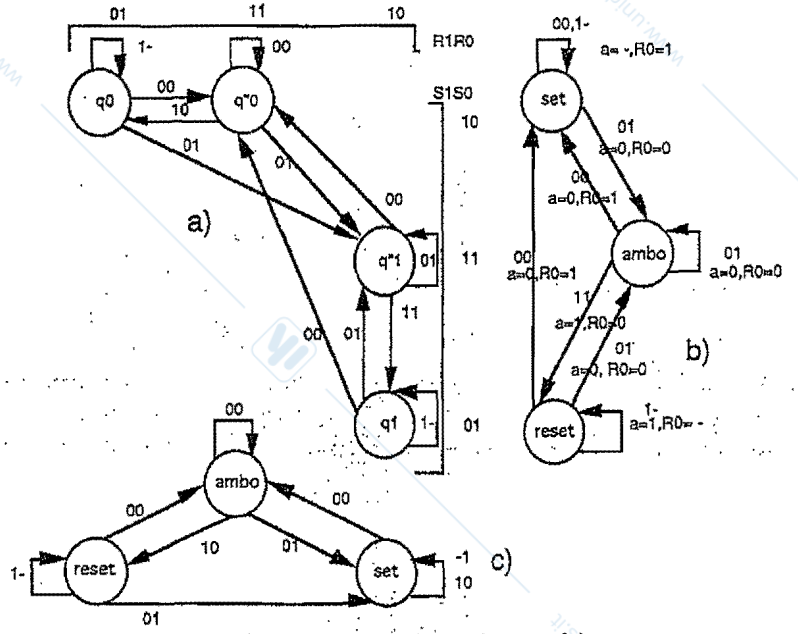


Fig. 11.3: Grafo della rete C e sua decomposizione:
a): grafo complessivo; *b)*: grafo della partizione S; *c)*: grafo della partizione R.

Si analizzano ora le caratteristiche dei due grafi componenti per trasformarli in quelli di flip-flop opportunamente posizionati. A tale scopo è utile far riferimento al grafo di un flip-flop RS a nand di fig. 11.4: nei nodi sono segnati nell'ordine l'uscita vera e quella falsa (11 è lo stato in cui entrambe le uscite sono vere), sugli archi i segnali di posizionamento, SR nell'ordine.

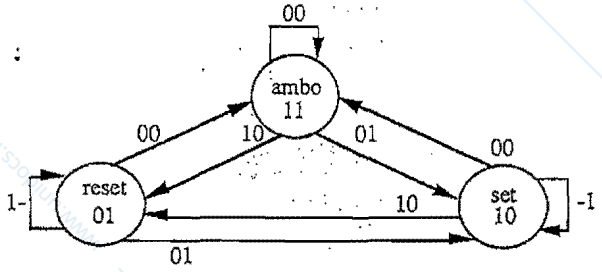


Fig. 11.4: Grafo del flip-flop RS a nand (0-attivo).

Progetto del flip-flop S (S_1 S_0)

Il grafo di fig. 11.3b) non coincide con quello del flip-flop se visto in funzione di a e D; peraltro, le sue transizioni si possono esprimere in funzione di a e R_0 come è appunto indicato in figura:

- nella prima riga si ha sempre che è $R_0=1$ indipendentemente da a;
- la transizione dalla prima alla seconda riga si ha per a=0 e $R_0=0$ (la transizione però avviene dopo che il flip-flop delle colonne sia già stato settato: diventa prima $R_0=0$ e poi commuta il flip-flop S);
- la transizione inversa si ha analogamente per a=0 e $R_0=1$;
- le due transizioni fra seconda e terza riga si hanno sulla terza colonna ($R_0=0$) e per a=1 (dalla seconda alla terza) ed a=0 (dalla terza alla seconda);
- la transizione dalla terza alla prima riga si ha sulla colonna ($R_0=1$) ed a=0.

Si ha pertanto:

$$set_S = a \qquad reset_S = R_0$$

Progetto del flip-flop R (R_1 R_0)

Il grafo di figura 11.3c) non coincide con quello di un flip-flop per la presenza del cappio 10 sullo stato *set* che nel flip-flop condurrebbe invece nello stato di *reset*. Una lieve modifica alla tabella e al grafo complessivo di C risolve il problema: si aggiunga lo stato q'_1, stabile sotto 10 ed equivalente a q_1, come illustrato in figura 11.2b, in modo che da q_1 si vada in q'_1, per

s·D=10 e qui si resti stabile, per salire a q''_0 per 00 (in tal modo, in questa seconda transizione vi sono meno corse); il grafo è presentato in figura 11.5.

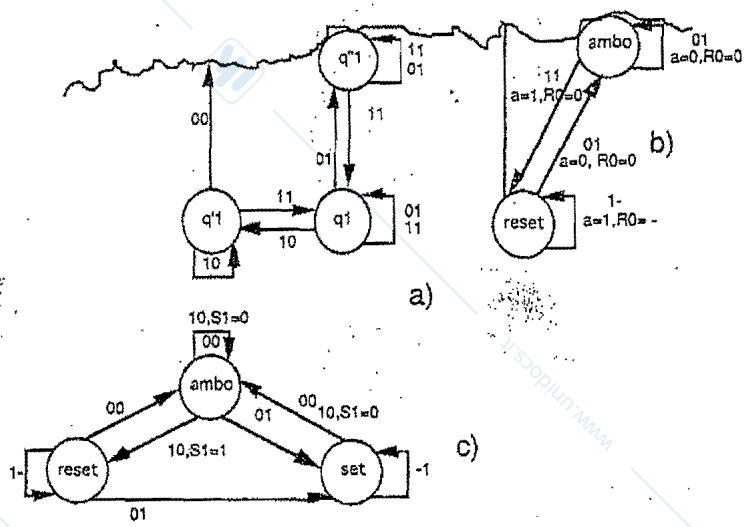


Fig. 11.5 Grafo della rete C modificato

Ne risulta, per il flip flop in esame, il grafo di fig. 11.5c), che coincide con quello del flip-flop con Set=a, Reset=D, ad eccezione degli archi ove è segnato S_1=0. La coincidenza diventa completa se peraltro si pone:

$$set_R = a \cdot S_1 \qquad reset_R = D$$

E' questa la soluzione adottata.

Descrizione del circuito (fig. 11.6)

Il circuito è composto dai 3 flip-flop di cui sopra; quello finale (U) fornisce l'uscita ed è posizionato dagli altri due, fra di loro interlacciati.

Il segnale di preset pone ad 1 il flip-flop finale U e quello S del suo set, mentre il clear pone U=0 e setta il flip-flop R per il reset .

La tempificazione è quella tipica dei flip-flop edge triggered.

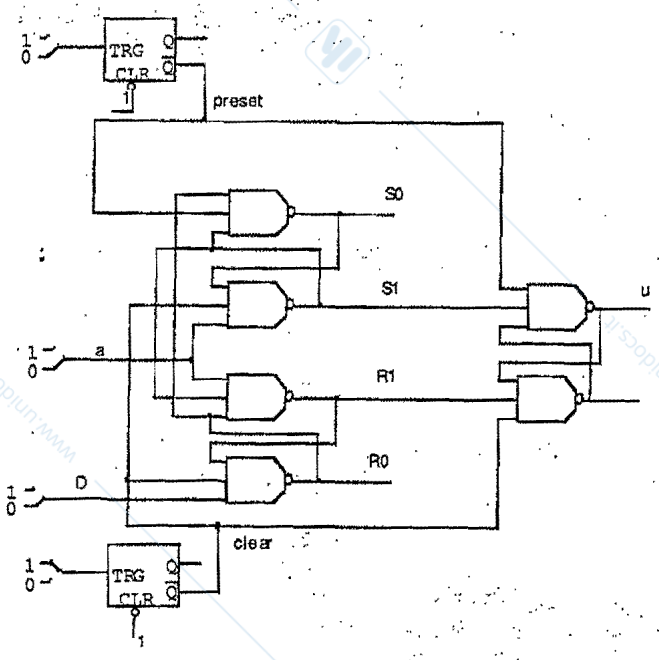


Fig. 11.6a: Flip-flop D con 3 RS componenti

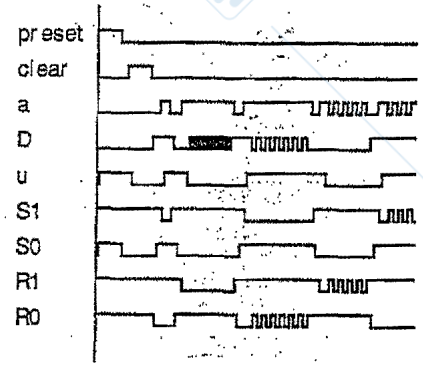


Fig. 11.6b: Flip-flop D con 3 RS componenti (tempificazione)

12. D master-slave asincrono

Tipo di circuito: rete sequenziale asincrona
Riferimento: RL, VII-6D
Obiettivo: studio di flip-flop

Testo

Sviluppare il progetto di un flip-flop D master-slave senza il vincolo che D non vari durante la fase attiva del segnale di sincronizzazione (master-slave " asincrono").

Impostazione del progetto

Il progetto si può sviluppare come progetto di macchina asincrona autonoma, con la metodologia classica delle macchine asincrone oppure assemblando due flip-flop D edge triggered sul fronte di salita e di discesa (§§ 9, 10, 11). In questo caso, ponendo in cascata un flip-flop D edge sul fronte di salita (*u-master*) ad uno sul fronte di discesa, (*u-slave*) si ottiene l'andamento temporale richiesto: il primo flip-flop memorizza all'interno del circuito complessivo (u-master) il segnale D sul fronte di salita di a e, restando alto a, essendo edge, mantiene tale valore anche se D varia. Il secondo flip-flop trasferisce il dato così memorizzato all'uscita del flip-flop (u-slave) sul fronte di discesa di a.

Descrizione del circuito (fig. 12.1)

Il flip-flop D master-slave asincrono è ottenuto ponendo in cascata i flip-flop sviluppati nel paragrafo 9: u-master sul fronte di salita (fig. 9.1a) e u-slave su quello di discesa (fig. 9.1b).

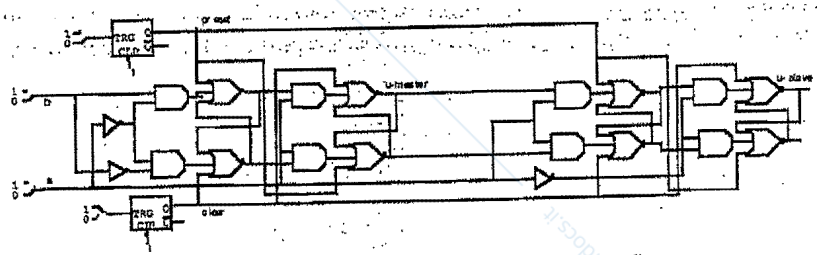


Fig. 12.1a: Flip-flop D master-slave "asincrono"

Dall'analisi della tempificazione si evince il comportamento da master-slave: u-master sale per D=1 sul fronte 0→1 di a, si mantiene alto con le variazioni di D e quindi u-slave lo ricopia sul fronte 1→0; poi analogamente u-slave si azzera e quindi sale di nuovo.

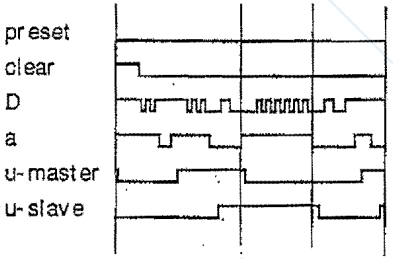


Fig. 12.1b: Flip-flop D master-slave "asincrono"

Si noti che i due flip-flop componenti sono a loro volta master-slave, ma " sincroni" , cioè sono tali solo se D non varia durante la fase attiva di a. Il flip-flop " asincrono" opera invece come master-slave anche se D varia ed è di complessità maggiore (il circuito sviluppato comprende 4 flip-flop RS).

13. Il flip-flop T

Il flip-flop *T* è il flip-flop fondamentale "a commutazione", in antitesi con quelli RS e D che sono "a memorizzazione dell'ingresso". Esso cambia il valore memorizzato in corrispondenza di un segnale di "trigger" T, che nei vari casi viene trattato come un impulso o un fronte. E' l'unico flip-flop che può avere in ingresso una sola variabile binaria (*T*). Il comportamento del flip-flop è illustrato dalle tabelle di figura 13.1

In figura 13.1a) è mostrata la tabella del flip-flop sincrono con il solo ingresso impulsivo T, che, come nel modello di macchina sincrona a sincronizzazione esterna, non è esplicitamente indicato. La tabella permette di evidenziare come, in corrispondenza dell'impulso T, il flip-flop commuti sempre.

In figura 13.1b) è riportata la tabella del flip-flop sincrono abilitato, i cui ingressi sono un segnale a livelli *T*, che definisce se il flip-flop deve commutare ed un impulso *a* (non esplicitato in tabella) che determina quando la commutazione deve avvenire.

In figura 13.1c) è infine riportata la tabella del flip-flop asincrono, che commuta sul fronte di salita dell'unico ingresso a livelli *T* (edge-triggered sul

fronte di salita). Del tutto analogo è quello che commuta sul fronte di discesa: la colonna delle uscite sarà allora 0011 piuttosto che 0110.

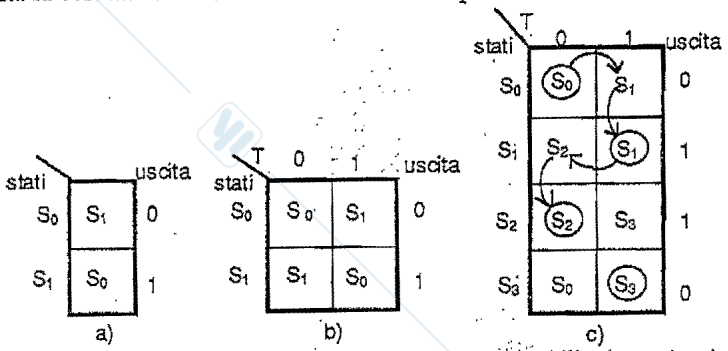


Fig. 13.1: Flip-flop T: *a)* sincrono fondamentale; *b)*: sincrono con abilitazione; *c)*: asincrono

Il flip-flop T è, per sua stessa definizione, un contatore modulo-2 (sincrono o asincrono a seconda dei casi). Esso è quindi il componente fondamentale dei contatori in modulo k>2. Si ricorda inoltre che, essendo il conteggio modulo-2 una forma di controllo di parità, il flip-flop T si usa per il controllo di parità sequenziale.

All'atto pratico, il flip-flop T è sostituito da quello JK (cfr. § 18), che diventa un flip-flop con il solo ingresso impulsivo per J=K=1 oppure uno abilitato per J=K=T.

14. T sincrono con RS latch: timing

Tipo di circuito: rete sequenziale sincrona
Riferimento: RL, VIII - 3A
Obiettivo: tempificazione del flip flop T fondamentale

Testo

Studiare la tempificazione del flip flop T sincrono fondamentale sviluppato utilizzando un flip-flop RS latch.

Nota di progetto

Lo schema sincrono fondamentale deriva dalla tabella di fig. 13.1a) e ha le seguenti equazioni:

$$R = T \cdot \overline{F} \qquad S = T \cdot F$$

ove R, S sono gli ingressi di un flip-flop RS fondamentale. Lo schema è quello di una rete sincrona a sincronizzazione esterna, ove T è l'impulso e l'unico ingresso. La corretta tempificazione è legata alla durata di T.

Descrizione del circuito (fig. 14.1)

Per studiare la tempificazione , sono simulati due flip-flop, l'uno (F ok) ben dimensionato, l'altro (F not ok) mal dimensionato. Si noti che:
- per entrambi i flip-flop il segnale T (T-ok e T-lungo) è realizzato con un monostabile;
- entrambi i flip flop sono resettati da un switch.

Ritardi

Tutti i ritardi sono posti pari a 4 u.t. (per meglio evidenziare la tempificazione) tranne:

- *monostabile* T-ok
  ritardo r = 30 u.t.
  ampiezza *w*=10 u.t.
- *monostabile* T-not-ok
  ritardo r=30 u.t.
  ampiezza w=20 u.t..

Tempificazione

Per il corretto funzionamento del flip flop RS componente (rete asincrona), l'ampiezza *w* dell'impulso deve essere tale da garantirne la commutazione: la durata dell' impulso deve essere maggiore del ritardo complessivo del flip-flop, per cui detto *R* il ritardo di ciascuna delle porte NOR, deve essere (cfr. § 4) w> 2R (nell'esempio w>8). Per il corretto funzionamento del flip-flop T (rete sincrona) l'ampiezza *w* dell'impulso deve essere tale da evitare più transizioni di stato. Detto *C* il ritardo della rete combinatoria (nel caso specifico *C* è il ritardo della porta AND), deve dunque essere w< 2R+C (nell'esempio *w<12*). Si deve dunque avere in sintesi:

$$2R < w < 4R$$

Nell'esempio è posto *w=20* per il flip flop "not ok": esso commuta due volte in quanto, partendo ad esempio da F=0 (ad un segnale di set segue quello di reset). La tolleranza per l'ampiezza di w è molto stretta, essendo

limitata al ritardo della rete combinatoria: occorre aggiungere ritardi ad hoc per rallentare la risposta della rete combinatoria se si vuole aumentare l'estremo superiore di w. Si noti inoltre che se fosse $w < 2R$ si avrebbe un segnale di posizionamento di durata troppo limitata per il flip flop RS, che raggiungerebbe una condizione di instabilità (cfr. § 4).

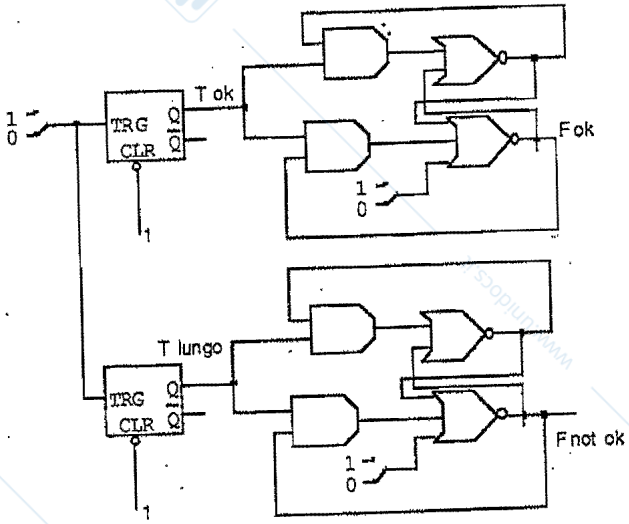


Fig. 14.1a: Flip-flop T sincrono con RS latch

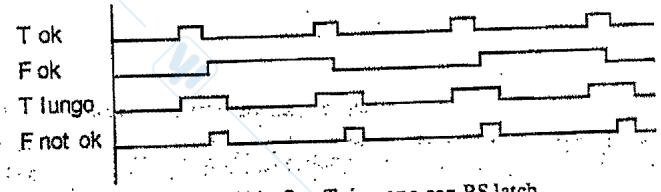


Fig. 14.1b: Flip-flop T sincrono con RS latch

Nota

Analogo è il caso del flip flop JK realizzato con:

$$R = K \cdot F \cdot a; \qquad S = J \cdot \overline{F} \cdot a$$

con *a* segnale di abilitazione.

15. T sincrono con RS edge

Tipo di circuito: rete sequenziale sincrona
Riferimento: MEI, VIII-6
Obiettivo: progettazione sincrona; studio di flip-flop

Testo

Progettare un flip-flop T sincrono con RS edge su linee di reazione. Considerare i due casi:
1) T unico segnale di ingresso, impulsivo;
2) T segnale a livello abilitante, *a* segnale impulsivo.

Tecnica di progetto

Progetto di rete sincrona a sincronizzazione esterna.

15.1 T semplice

Trattiamo dapprima il caso 1. La tabella del circuito è quella banale di fig.13.1a); detta *u* l'uscita del flip-flop, le funzioni di posizionamento sono:

$$S = \overline{u} \qquad R = u$$

mentre T svolge la funzione di segnale di sincronismo.

Descrizione del circuito (fig. 15.1)

Si assume il flip-flop RS edge sul fronte di discesa e si ottiene uno schema del tutto simile a quello corrispondente dei flip-flop D ed RS. Similmente a questi ultimi, sono aggiunti al circuito i segnali di *preset* e *clear*.

Tempificazione

L'uso del flip-flop edge migliora le caratteristiche di tempificazione del flip-flop sincrono: l'impulso T, che deve comunque essere maggiore di 2 u.t. per il funzionamento del flip-flop RS, ma non ha un estremo superiore come nel § 14 ed è posto lungo 10 u.t. La tempificazione nei riguardi di T è simile a quelle del flip-flop D, salvo che in *p* viene ricopiato $\overline{u}$ piuttosto che D.

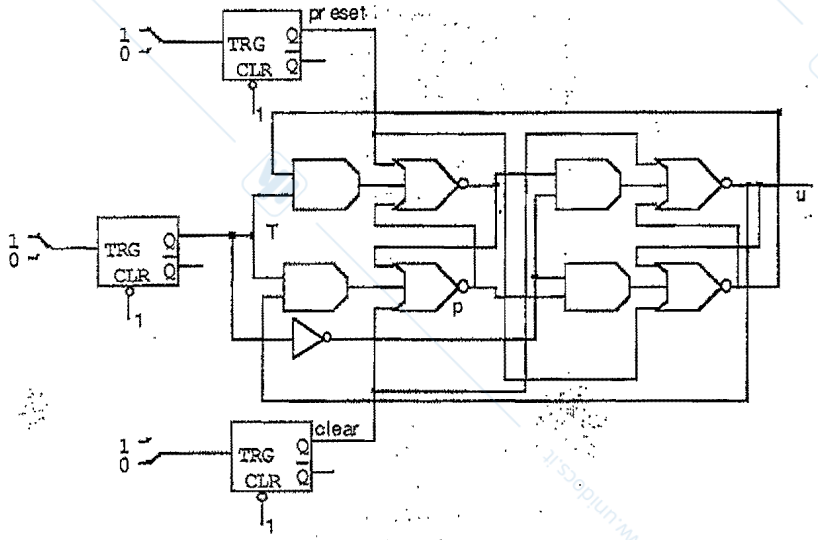


Fig. 15.1a: Flip-flop T sincrono con RS edge

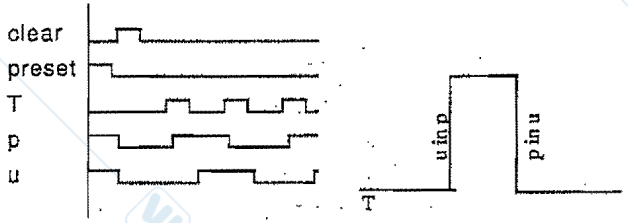


Fig. 15.1b: Flip-flop T sincrono con RS edge

Nota

Si noti che il circuito, analogamente al flip-flop D edge-triggered sul fronte di discesa (cfr. § 10), opera come un master-slave con impulso 1-attivo (acquisisce sul fronte di salita, commuta su quello di discesa) ed anche come un flip-flop edge triggered (comunque l'uscita varia esclusivamente sul fronte di discesa). Se si assumesse come componente un RS sul fronte di salita, il circuito risultante sarebbe un master-slave con imulso 0-attivo.

Il circuito coincide con quello del § 17 (T asincrono).

15.2 T abilitato

Il caso 2 (T segnale a livello, *a* impulso) risponde al modello di fig.13.1b e dà luogo sostanzialmente al medesimo circuito. Detto TA il segnale a livello, il ruolo che prima era di T viene ora assunto da *a*, che quindi opera come segnale di sincronismo del flip-flop RS; TA funge da abilitazione ed i segnali di posizionamento risultano:

$$S = \bar{u} \cdot TA \qquad R = u \cdot TA$$

Il segnale di sincronismo *a*, poi, si aggiunge alle due porte and che calcolano S e R in quanto, come nel caso 1, è questo lo schema del flip-flop RS componente.

16. T asincrono

Tipo di circuito: rete sequenziale asincrona
Riferimento: RL, IX-esempio 4
Obiettivo: progettazione asincrona; alea essenziale; contatori

Testo

Realizzare un flip-flop T sensibile ai fronti di salita di T o, il che è lo stesso, un contatore modulo 2 asincrono.

Tecnica di progetto

Progetto rete asincrona fondamentale e valutazione di soluzioni alternative per la realizzazione.

Progetto

La tabella di stato è quella del flip-flop T asincrono (cfr. fig. 13.1c). Si effettua dunque la seguente assegnazione degli stati, priva di corse:

$$S_0 = (00); \quad S_1 = (01); \quad S_2 = (11); \quad S_3 = (10)$$

e dette nell'ordine y_1, y_2 le variabili di stato e *u* l'uscita, si ottengono le tabelle di fig. 16.1 e le seguenti equazioni delle variabili di stato e uscita:

$$y'_1 = y_2\bar{T} + y_1T + y_1y_2 \qquad y'_2 = y_2\bar{T} + \bar{y}_1T + \bar{y}_1y_2 \qquad u = y_1$$

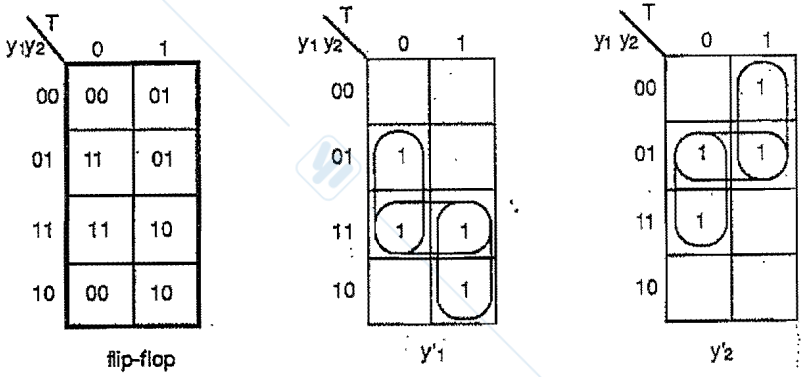


Fig.16.1: Mappe per il progetto

Si noti che l'alea statica è eliminata in entrambi le funzioni con l'aggiunta di una clausola.

Descrizione del circuito (fig. 16.2)

- T è simulato con un switch;
- lo stato della rete è evidenziato attraverso un display esadecimale;
- la rete è posta nello stato reset (Z=3) mediante il switch reset=0.

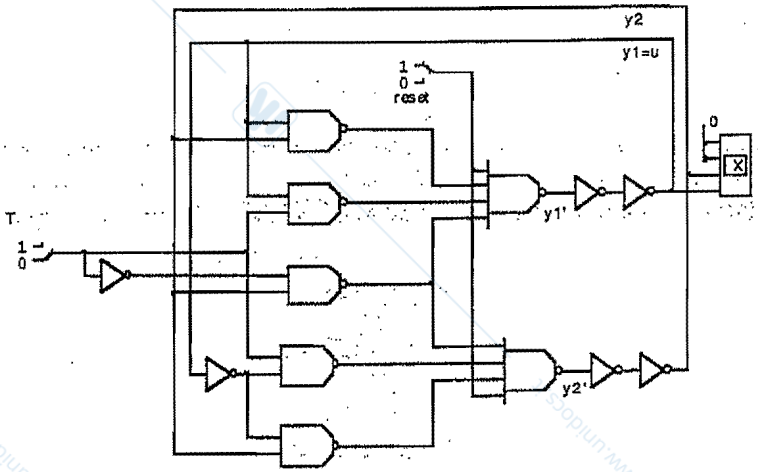


Fig. 16.2a: Flip-flop T asincrono (o edge-triggered o contatore modulo-2)

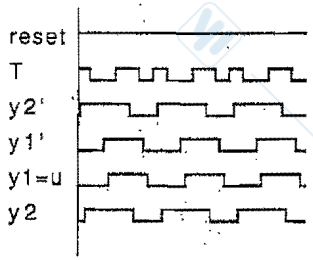


Fig. 16.2b: Flip-flop T asincrono : tempificazione

Tempificazione

Si ricorda che il flip-flop T è un contatore modulo-2 e che i contatori sono esempi classici di sistemi con alee essenziali. Per tale motivo sono stati apposti ritardi sulle linee di reazione, realizzati con due invertitori in cascata. In assenza di tali ritardi, l'effetto dell'alea essenziale sarebbe un grossolano mal funzionamento del circuito (si veda § II.6, ove questo stesso circuito è assunto come esemplificativo dell'alea essenziale).

17. T asincrono con 2 RS

Tipo di circuito: rete sequenziale asincrona
Riferimento: MEI, VIII-6
Obiettivo: Progettazione asincrona, studio di flip-flop

Testo

Progettare un flip-flop T che commuti sul fronte di salita di T con segnali di preset e clear.

Tecnica di progetto

Progetto rete di asincrona con flip-flop su linee di reazione.

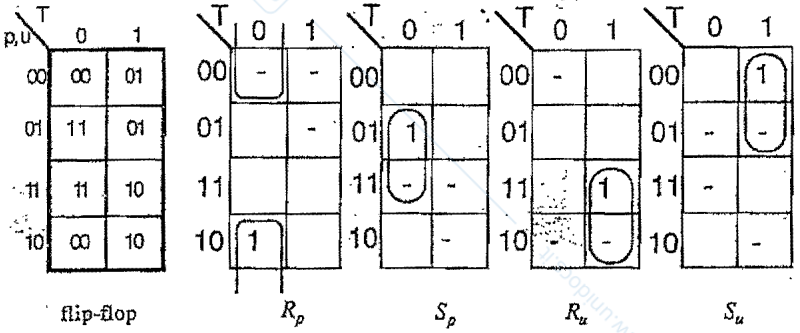
Progetto

La tabella di stato è quella del flip-flop T asincrono (cfr. fig. 13.1c). Si effettua dunque la seguente assegnazione degli stati, priva di corse:

S_0= (00); S_1= (01); S_2= (11); S_3= (10)

Lasciando da parte per il momento i segnali *preset* e *clear* e detti nell'ordine *u*, *p* i 2 flip-flop di stato (il flip-flop u coincide con l'uscita), si hanno le tabelle di fig. 17.1 e i relativi segnali di posizionamento:

$$R_p = \overline{T}\cdot\overline{u} \qquad S_p = \overline{T}\cdot u \qquad R_u = T\cdot p \qquad S_u = T\cdot\overline{p}$$



I segnali *preset* e *clear* si aggiungono poi direttamente sul set e reset dei flip-flop RS componenti.

Descrizione del circuito e tempificazione (fig.17.2)

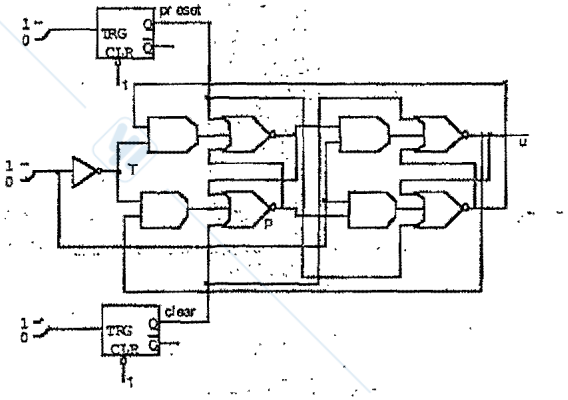


Fig. 17.2a: Flip-flop T asincrono con 2 RS

Il circuito è identico a quello del § 15 (flip-flop sincrono con RS edge), così come la tempificazione: ne è stata sottolineata la caratteristica a livelli - adoperando un switch invece di un monostabile per simulare il segnale T.

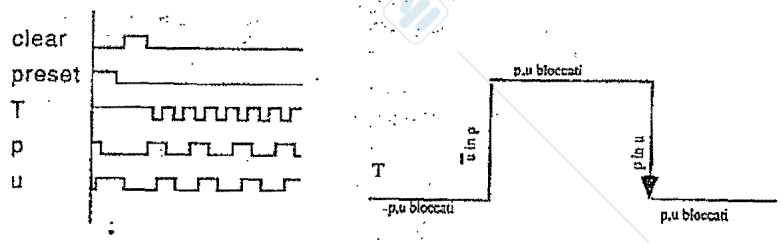


Fig. 17.2:b: Flip-flop T asincrono con 2 RS

18. Il flip-flop JK

Il flip-flop JK riunisce in *sé* i comportamenti dei flip-flop RS e T: si comporta come il T se è $J=K=1$, altrimenti si comporta come lo RS, con *J* come set e *K* come reset. Il flip-flop è sempre abilitato da un segnale di sincronizzazione *a*; *J* e *K* definiscono lo stato che deve raggiungere il flip-flop, *a* individua quando ciò debba avvenire.

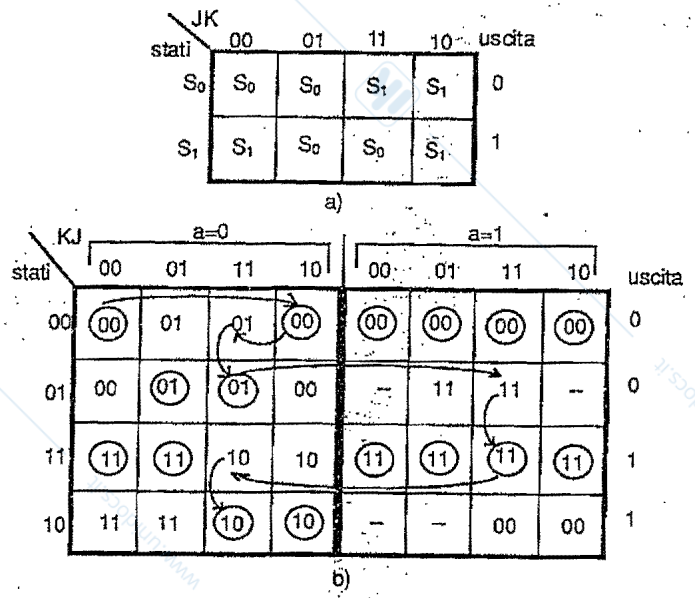


Fig. 18.1: Flip-flop JK a) tabella sincrona; b) edge-triggered

La tabella sincrona del flip-flop è mostrata in fig.18.1a: in essa si intende che ogni transizione avvenga in sincronismo con *a*. In fig.18.1b) è mostrata la tabella del flip-flop edge-triggered sul fronte di salita; in essa i quattro stati necessari sono stati codificati su 2 variabili come 00, 01, 10, 11, in modo analogo a quanto è stato fatto per il flip-flop D edge (cfr. §§ 6, 10); è evidenziata una sequenza di commutazione a partire dallo stato 00 stabile sotto l'ingresso 00:

- con *a*=0, KJ variano 00→10: lo stato resta 00;
- KJ variano 10→11: la macchine si prepara a commutare nello stato 01 (l'uscita non varia);
- *a* varia 0→1 e sul suo fronte il flip-flop si sposta nello stato 11, variando l'uscita;
- alla ulteriore variazione 1→0 di *a*, restando J e K inalterati, il flip-flop si prepara alla nuova transizione nello stato 10.

19. JK sincrono con RS latch

Tipo di circuito: rete sequenziale sincrona
Riferimento: RL, VIII-3C e X-5, esempio 6; MEI, VIII-6
Obiettivo: flip flop JK

Testo

Progettare un flip-flop JK con RS latch.

Scelta di progetto

Si assume il modello di rete sincrona a sincronizzazione esterna con l'abilitazione *a* che svolge il ruolo di impulso di sincronismo. Sulle linee di reazione si pone un flip-flop RS abilitato latch.

Funzioni di posizionamento

La tabella di stato è quella sincrona di fig. 18.1a). Detto F il flip-flop di stato, i due stati sono banalmente assegnati:

$$S_0 \rightarrow F=0 \qquad S_1 \rightarrow F=1$$

Essendosi adottato il modello sincrono a sincronizzazione esterna, essa va letta come tabella per il progetto delle funzioni di posizionamento a livelli.

L'impulso *a* sarà poi applicato direttamente al flip-flop di stato. Le funzioni di posizionamento sono quindi:

$$R=K\cdot F \quad S=J\cdot\bar{F}$$

L'abilitazione può porsi in AND con le funzioni di cui sopra:

$$R=K\cdot F\cdot a \qquad S=J\cdot\bar{F}\cdot a$$

Descrizione del circuito (fig. 19.1)

Il circuito ha la medesima struttura e i medesimi problemi di tempificazione del flip-flop T con RS latch (cfr. § 14). Le difficoltà di tempificazione suggeriscono di adottare la versione edge del flip-flop.

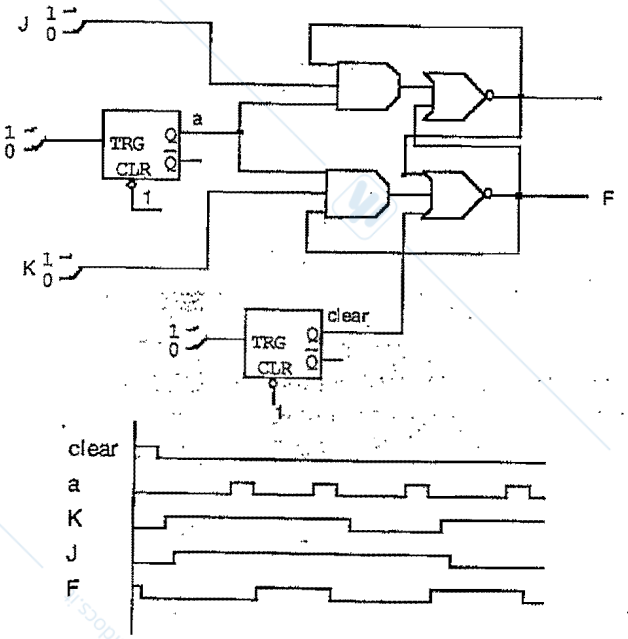


Fig. 19.1: Flip-flop JK sincrono con RS latch

20. JK sincrono con RS edge (master-slave)

Tipo di circuito: rete sequenziale sincrona
Riferimento: MEI, VIII-6
Obiettivo: flip flop JK

Testo

Progettare un flip-flop JK sincrono con RS edge triggered sulle linee di reazione.

Scelta di progetto

Si assuma il modello di rete sincrona a sincronizzazione esterna con l'abilitazione *a* che svolge il ruolo di impulso di sincronismo. Per evitare i problemi di tempificazione di cui al flip-flop con RS latch (cfr. § 19), sulle linee di reazione si pone un flip-flop RS edge triggered.

Funzioni di posizionamento

Come per il caso del flip-flop con RS latch, la tabella di stato è quella sincrona di cui alla fig. 18.1a). I due stati sono assegnati in modo banale, per cui, detto *u* il flip-flop di stato, si ha:

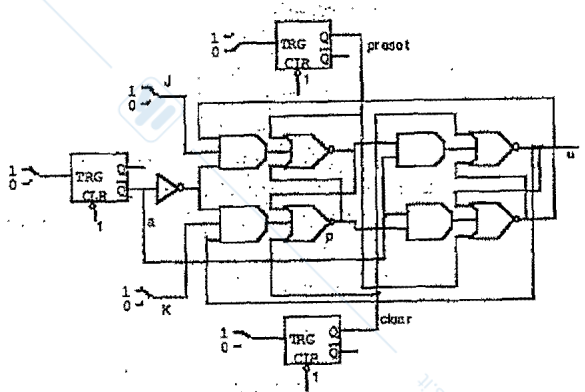


Fig. 20.1a: Flip-flop JK sincrono con RS edge su fronte di salita (master-slave su impulso negativo)

$S_0 \rightarrow u=0$ $S_1 \rightarrow u=1$

Essendosi adottato il modello a sincronizzazione esterna, la tabella va letta come tabella per il progetto delle funzioni di posizionamento a livelli. L'impulso *a* sarà poi applicato direttamente al flip-flop di stato. Le funzioni di posizionamento sono quindi:

$$S = J\cdot\bar{u} \qquad R = K\cdot\bar{u}$$

20.1 JK sul fronte di salita

Assumendo come flip-flop di stato quello RS edge realizzato con 2 RS fondamentali (cfr.§ 5), si ottiene il circuito di figura 20.1 se il flip-flop RS opera sul fronte di salita. Considerando che il flip-flop RS edge-triggered presenta i 2 ingressi R ed S in and con $\bar{a}$ si ha:

$$S = \bar{a}\cdot J\cdot\bar{u} \qquad R = \bar{a}\cdot K\cdot u$$

Considerando poi le funzioni di posizionamento del flip-flop RS e i segnali di *preset* e *clear* si ha:

$$S_p = \bar{a}\cdot J\cdot\bar{u} + \text{preset} \qquad R_p = \bar{a}\cdot K\cdot u + \text{clear}$$
$$S_u = a\cdot p + \text{preset} \qquad R_u = a\cdot\bar{p} + \text{clear}$$

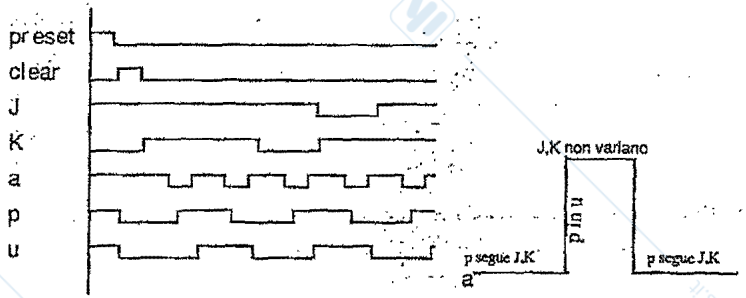


Fig. 20.1b: Flip-flop JK edge sul fronte di salita: tempificazione

20.2 JK su fronte di discesa

Procedendo analogamente si ha:

$$S_p = a\cdot J\cdot\bar{u} + \text{preset} \qquad R_p = a\cdot K\cdot u + \text{clear}$$

$S_u = \bar{a} \cdot p + preset \qquad R_u = \bar{a} \cdot \bar{p} + clear$

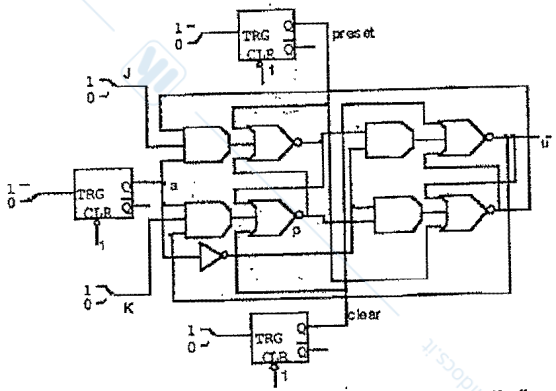


Fig. 20.2 a: Flip-flop JK sincrono con RS edge su fronte di discesa (master-slave su impulso positivo)

20.3 JK master-slave

Con riferimento al flip-flop sul fronte di discesa, si osservi quanto segue (per il flip flop sul fronte di salita valgono le considerazioni duali).

Il flip-flop JK sincrono qui sviluppato ha il comportamento del flip-flop master-slave, ma con una condizione di vincolo sugli ingressi: essi non devono variare all'interno dell'impulso, cioè per $a=1$, come del resto è tipico per le macchine impulsive. La simulazione del segnale a con il monostabile rende appunto impossibile provocare variazioni di J e K durante la presenza dell'impulso.

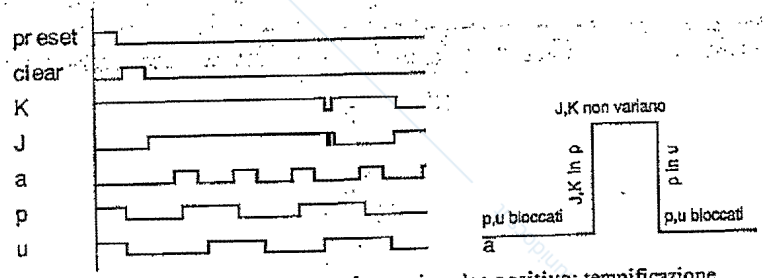


Fig. 20.2 b: Flip-flop JK master-slave su impulso positivo: tempificazione

Il flip-flop sul fronte di discesa opera, dunque, come master-slave per impulso 1-attivo mentre, dualmente, il flip-flop sul fronte di salita opera con impulso 0-attivo (cfr. fig. 20.3).

Sui manuali dei circuiti integrati questo tipo di circuito viene comunque detto *master-slave*.

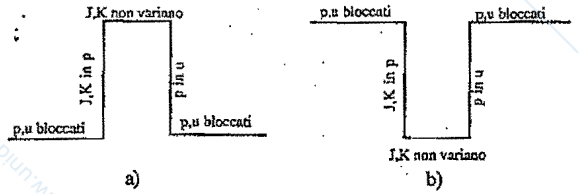


Fig. 20.3: Tempificazione master-slave: a) impulso 1-attivo (circuito di fig.20.2) b) impulso 0-attivo (circuito di fig.20.1)

Confronto con il flip-flop edge triggered

Contrariamente a quanto accade per i flip-flop D e T, il flip-flop impulsivo (master-slave) qui presentato non coincide con il flip-flop edge triggered (cfr. § 22).

Si prenda ad esempio il flip-flop operante con impulso 1-attivo. Se J e K variassero, per $a=1$ ed $u=1$, il flip-flop p potrebbe soltanto essere resettato ($R_p = a \cdot K \cdot u = K$) e non settato ($S_p = a \cdot J \cdot \bar{u} = 0$), mentre in ogni caso p è ricopiato in u sul fronte di discesa. Se quindi in questa fase si ha la sequenza di ingresso JK=00→01→00→10, sul fronte di discesa si avrebbe u=0 mentre l'ultimo valore degli ingressi dovrebbe mantenere il flip-flop a 1.

21. JK master-slave tutte nor (o nand)

Tipo di circuito: rete sequenziale sincrona
Riferimento: § 20; circuito commerciale 7472
Obiettivo: flip-flop JK

Testo

Realizzare il flip-flop master-slave di cui al § 20 adoperando soltanto porte NOR (oppure, dualmente, porte NAND). Si mantengono le condizioni di vincolo per le quali J e K non variano per $a=0$ *(NOR)* oppure per $a=1$ (NAND).

21.1 master-slave a NOR

Si parte dalla versione con impulso 0-attivo (fronte di salita, cfr. § 20):

$S_p = \bar{a}\cdot\bar{u}\cdot J + \text{preset}$ $\qquad R_p = \bar{a}\cdot u\cdot K + \text{clear}$

$S_u = a\cdot p + \text{preset}$ $\qquad R_u = a\cdot\bar{p} + \text{clear}$

e si effettuano le seguenti modifiche:

1) le due AND a monte del flip-flop *p* sono trasformate in NOR; di conseguenza scompare la NOT e J, K diventano 0-attivi:

$S_p = \text{nor}(a, u, \bar{J}) + \text{preset}$ $\qquad R_p = \text{nor}(a, \bar{u}, \bar{K}) + \text{clear}$

2) Le due AND del flip-flop u sono trasformate in NOR; di conseguenza si dovrebbe inserire una NOT su *a* ed invertire i segnali provenienti da *p* (p sul reset e $\bar{p}$ sul set); dovrebbe dunque essere:

$S_u = \text{nor}(\bar{a}, \bar{p}) + \text{preset}$ $\qquad R_u = \text{nor}(\bar{a}, p) + \text{clear}$

(graficamente, invece di invertire le entrate sul set e reset di u, si è disegnato capovolto quest'ultimo flip-flop).

3) Invece di alimentare entrambe le nor di posizionamento con $\bar{a}$, si alimenta il set con il set di p, il reset con il reset di p:

$S_u = \text{nor}(S_p, \bar{p}) + \text{preset}$ $\qquad R_u = \text{nor}(R_p, p) + \text{clear}$

Ciò è lecito in quanto, si ha:

$S_u = \text{nor}(\bar{a}, \bar{p}) = a\cdot p$

$= a\cdot p + u\cdot a\cdot p$ (u·a·p è incluso in a·p)

$= a\cdot p + \bar{u}\cdot a\cdot p + u\cdot\bar{a}\cdot p$ (u·ā·p è don't care: per a=0, u=1, u non deve variare e può anche essere di nuovo settato)

$= a\cdot p + u\cdot p$

$= a\cdot p + u\cdot p + a\cdot p\cdot\bar{J}$ (a p·$\bar{J}$ è incluso in a·p)

$+\bar{a}\cdot p\cdot\bar{J}\cdot u$ (incluso in p·u)

$+\bar{a}\cdot p\cdot\bar{J}\cdot\bar{u}$ (è nullo: per a=0, u=0, p=1 è J=1 perché p o è stato settato da J=1 o ha commutato per K=J=1)

$= a\cdot p + u\cdot p + p\cdot\bar{J}$

$= (a+u+\bar{J})\cdot p = \overline{S_p}\cdot p = \text{nor}(S_p, \bar{p})$

Da cui si ottiene con analogo sviluppo per R_u:

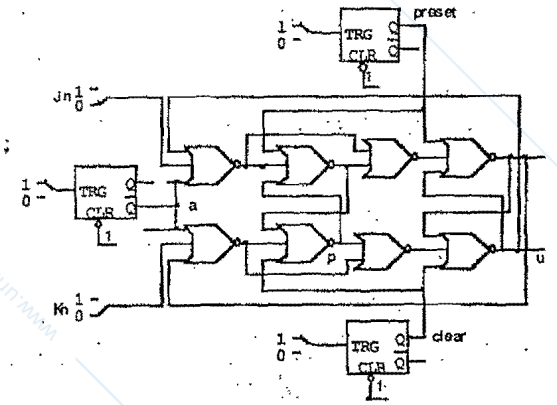


Fig. 21.1a: Flip-flop master-slave "tutte nor" con impulso 0-attivo

$S_p = \text{nor}(a, u, \bar{J}) + \text{preset}$ $\qquad R_p = \text{nor}(a, \bar{u}, \bar{K}) + \text{clear}$

$S_u = \text{nor}(S_p, \bar{p}) + \text{preset}$ $\qquad R_u = \text{nor}(R_p, p) + \text{clear}$

In definitiva, per il flip-flop a tutte NOR, si ha che :

- J, K sono 1-attivi
- set e reset sono 0-attivi.

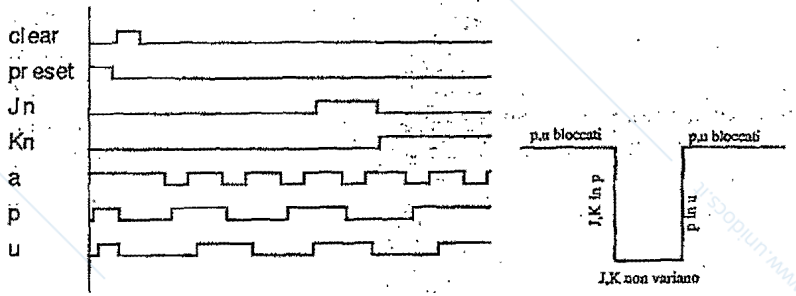


Fig. 21.1b: Flip-flop master-slave "tutte nor": tempificazione

21.2 master-slave a NAND

Si consideri prima di tutto che il flip-flop a NAND ed ingressi di posizionamento 0-attivi è il duale del flip-flop a NOR, con ingressi invertiti di posi-

zione: il set di fronte alla uscita F, il reset di fronte ad $\overline{F}$. Pertanto, tutti i circuiti a NOR già visti diventano per dualità circuiti a NAND ed il flip-flop è con segnali di posizionamento 0-attivi:

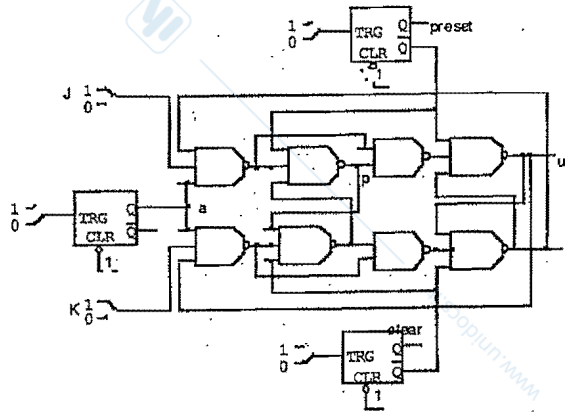


Fig. 21.2a: Flip-flop master-slave "tutte nand" con impulso 1-attivo

$$S_p = (a+u+\overline{J})\cdot\overline{preset}$$
$$R_p = (a+\overline{u}+\overline{K})\cdot\overline{clear}$$
$$S_u = (\overline{a}+\overline{p})\cdot\overline{preset}$$
$$R_u = (\overline{a}+p)\cdot\overline{clear}$$

e quello con tutte NAND:

$$S_p = \text{nand}(\overline{a}, \overline{u}+J)\cdot\overline{preset}$$
$$R_p = \text{nand}(\overline{a}, u, K)\cdot\overline{clear}$$
$$S_u = \text{nand}(S_p, p)\cdot\overline{preset}$$
$$R_u = \text{nand}(R_p, \overline{p})\cdot\overline{clear}$$

con:
- J, K 1-attivi;
- set, preset 0-attivi.

Il flip-flop commerciale 7472 si basa su quest'ultimo schema logico.

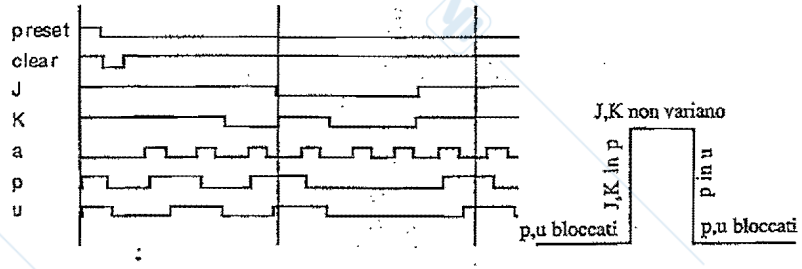


Fig. 21.2b: Flip-flop master-slave "tutte nand" con impulso 1-attivo: tempificazione

22. JK edge con 2 RS

Tipo di circuito: rete sequenziale asincrona
Obiettivo: progetto asincrono

Testo

Progettare un flip-flop JK edge triggered sul fronte di salita.

Tabella di stato

La tabella di stato è quella riportata in fig.18.1b).

Scelte di progetto

Rete asincrona con 2 flip-flop RS (p, u) sulle linee di reazione.

Funzioni di posizionamento

Avendo assegnati gli stati come indicato nella citata tabella 18.1b, ne risulta che uno dei due flip-flop RS (u) rappresenta anche l'uscita del flip-flop; occorre dunque progettare i segnali di posizionamento S_u, R_u, (set e reset di u), S_p, R_p (set e reset di p). Dalle mappe di fig. 22.1 si ha:

$$S_p = \overline{a}\cdot(\overline{u}\cdot J + u\cdot\overline{K})$$
$$R_p = \overline{a}\cdot(u\cdot K + \overline{u}\cdot\overline{J})$$
$$S_u = a\cdot p$$
$$R_u = a\cdot\overline{p}$$

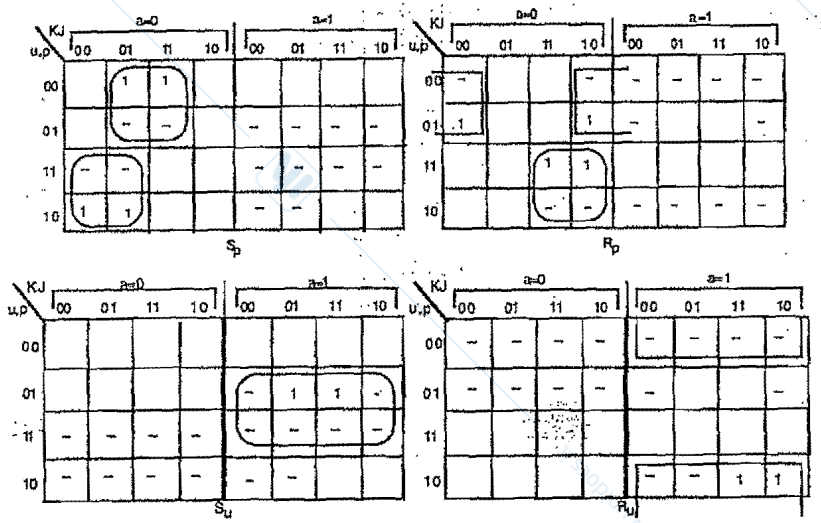


Fig.22.1: Mappe di Karnaugh delle funzioni di posizionamento

Descrizione del circuito (fig. 22.2)

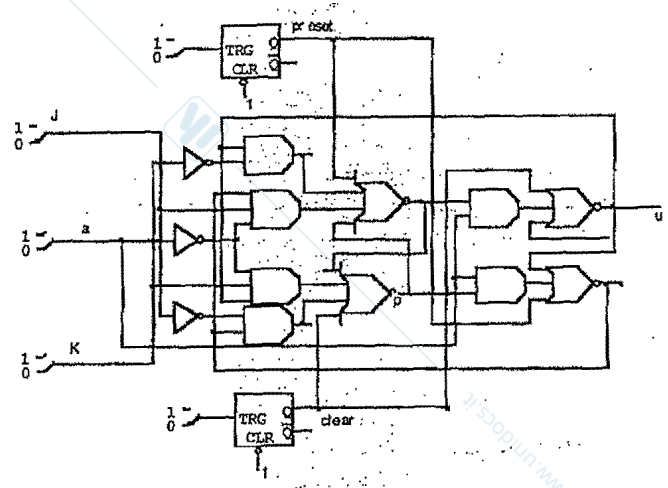


Fig. 22.2a: Flip-flop JK edge triggered

Il circuito è simile a quello "JK master-slave" (cfr. § 20), salvo che aggiunge il termine $\bar{a} \cdot \bar{u} \cdot \bar{J}$ al reset e $\bar{a} \cdot u \cdot \bar{K}$ al set; in tal modo, per a=0 il flip-flop segue interamente J, K.

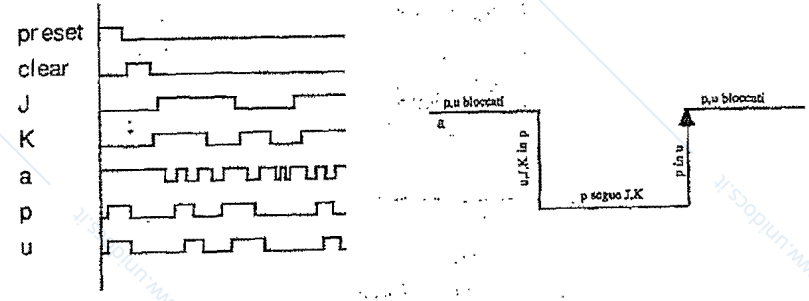


Fig. 22.2b: Flip-flop JK edge triggered: tempificazione

23. JK edge con RS e rete di posizionamento

Tipo di circuito: rete sequenziale asincrona
Riferimento: circuito commerciale 74101
Obiettivo: conoscenza di circuiti commerciali

Testo

Progettare un flip-flop JK edge triggered sul fronte di discesa

Tecnica di progetto

E' adottata una tecnica mista, che in parte si rifà alle metodologie classiche e in parte è basata su intuizioni particolari per questo progetto, tese alla semplificazione della rete.

Impostazione del progetto

Si parte dallo schema latch di fig.19.1, trasformandolo quindi in edge. Lo schema latch è a nand: lo stadio di uscita è un flip-flop e le porte di ingresso, considerando R e S 0-attivi, sono (fig.23.1):

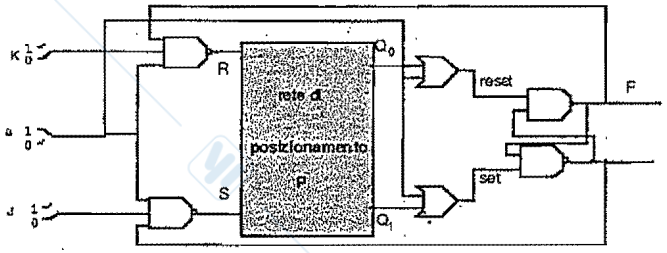


Fig.23.1: Schema del flip-flop

$$R = \overline{K \cdot F \cdot a} \qquad S = \overline{J \cdot \overline{F} \cdot a}$$

Fra le nand di ingresso ed il flip-flop di uscita è interposta una apposita rete P di posizionamento del flip-flop, i cui segnali di set e reset sono:

$$set = Q_1 + a \qquad reset = Q_0 + a$$

Si determina quindi il seguente comportamento:
- per a=1, S e R assumono i valori che, trasformati dalla rete P in Q_0 e Q_1, dovranno determinare sul successivo fronte 1→0 di *a* il valore di F;
- sul fronte 1→0 i valori di Q_0 e Q_1 vengono trasferiti in F e qui memorizzati;
- permanendo a=0, Q_0 e Q_1 vengono mantenuti stabili dalla rete in modo che i segnali di set e reset durino il tempo sufficiente a definire lo stato di F (vedi note sulla tempificazione).

La rete P, dovendo mantenere i valori Q_0 e Q_1, è una rete sequenziale con ingressi R, S e variabili di stato Q_0, Q_1.

Progetto

Sulla base delle scelte di cui sopra, si ha la tabella di stato di fig.23.2:
- per S=R=0, l'ingresso del flip-flop di uscita deve essere neutro: Q_0Q_1=11;
- per SR=01 e Q_0Q_1=01, il flip-flop è di nuovo posto in set: Q_0Q_1=01, τ(01,01)=01, lo stato è stabile;
- analogamente, per SR=10 e Q_0Q_1=10, il flip-flop è resettato: τ(10,10)=10;
- se dagli stati stabili di cui sopra gli ingressi si portano a SR=11, gli stati restano stabili;
- se da Q_0Q_1=01 stabile per SR=11 si passa ad SR=10, lo stato dovrebbe invertisi (Q_0Q_1=01): per evitare la corsa si pone τ(01,10)=11, raggiungendo quindi lo stato stabile 10 con un ciclo di due transizioni;
- analogamente si pone τ(10,01)=11;

- si pone τ(11,11)=11. Si noti peraltro che in funzionamento normale questo punto non dovrebbe mai essere raggiunto; se tuttavia lo fosse per una variazione simultanea di S e R oppure per una errata tempificazione, si deve avere la garanzia che il flip-flop di uscita non sia alterato: il valore Q_0Q_1=11 (piuttosto che non specificato) fornisce tale garanzia.

Q_0Q_1 \ SR	00	01	11	10
00	–	–	–	–
01	11	(01)	(01)	11
11	(11)	01	11	10
10	11	11	(10)	(10)

Fig.23.2: Rete di posizionamento: tabella di stato

Dalla tabella di stato si deduce:

$$Q_0' = \overline{R} + \overline{Q_1} + Q_0 \cdot S = \overline{Q_1 \cdot R} + Q_0 \cdot S$$

$$Q_1' = \overline{S} + \overline{Q_0} + Q_1 \cdot R = \overline{Q_0 \cdot S} + Q_1 \cdot R$$

Descrizione del circuito (fig.23.3)

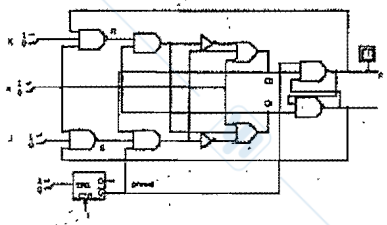


Fig.23.3: Flip-flop JK edge-triggered

Il circuito riproduce il progetto di cui sopra, con le varianti che seguono:
- Piuttosto che calcolare indipendentemente $\overline{Q_1R}$, lo si ottiene da Q_1R che serve per Q_1' (ed analogamente per $\overline{Q_0S}$);
- Piuttosto che realizzare $reset = Q_0' + a = \left(\overline{Q_1R} + Q_0S\right) + a$, si pone direttamente: $Q_0' = \overline{Q_1R} + Q_0S + a$ e $reset = Q_0'$ (ed analogamente per $set = Q_1'$). Il fatto di avere aggiunto a alle variabili di stato della rete P non ne altera il comportamento, in quanto essa è attiva solo per a=0.
- Alla rete è stato aggiunto un preset 0-attivo che pone F=Q_1=1; analogamente si potrebbe aggiungere un clear.

Il circuito presenta lo stesso schema logico del circuito commerciale 74101, che appunto presenta il solo preset in quanto integra su un unico chip

due flip-flop di tal genere e non ha a disposizione il numero di piedini del chip per presentare sia il preset che il clear.

Note sulla tempificazione

La rete P potrebbe essere realizzata anche come rete combinatoria, ma in tal caso occorrerebbe progettare ad hoc i ritardi delle porte, cosa che si preferisce evitare. Comunque, per la rete combinatoria P dovrebbe aversi:

SR	Q_0Q_1
00	11
01	01
10	10
11	11

$$Q_0 = S + \overline{R}$$
$$Q_1 = R + \overline{S}$$

e la rete si semplificherebbe in quella di fig. 23.4.

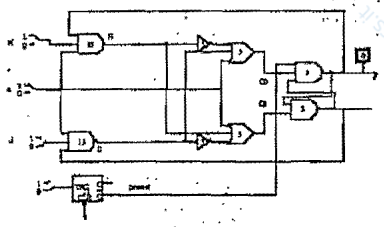


Fig. 23.4: Flip flop di fig.23.3 con rete P combinatoria

In tal caso, però, come accennato, esiste un delicato problema di sincronizzazione: il ritardo delle porte R, S deve essere maggiore del doppio del ritardo delle porte nand del flip-flop finale, per sostenerne la commutazione (cfr. §III. 4). In figura è indicato un esempio di tempificazione corretta; se viceversa il ritardo di tutte le porte fosse eguale, il flip-flop andrebbe in oscillazione, come indicato nella figura 23.5, ottenuta per ritardi tutti uguali a 5 unità di tempo.

In ogni caso, la piccola semplificazione del circuito non giustifica l'adozione della soluzione combinatoria.

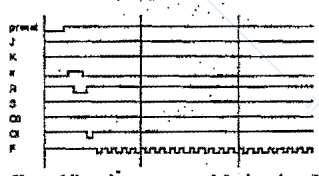


Fig. 23.5: Tempificazione errata del circuito di fig.23.4

Capitolo quarto
Reti sequenziali asincrone

1. Macchine sequenziali

Una macchina sequenziale è caratterizzata da un proprio insieme di *ingressi I*, *uscite U* e *stati interni* (o semplicemente *stati*) *S* e può essere descritta mediante un'apposita tabella, che definisce per alcune coppie ingresso-stato lo *stato seguente* (funzione $\tau: I \times S \to S$) e l'uscita (funzione $\omega: I \times S \to U$ per il modello di Mealy o funzione $\omega: S \to U$ per il modello di Moore).

La macchina è detta *completamente specificata* se τ e ω sono definite per tutte le coppie stato-ingresso, incompletamente altrimenti.

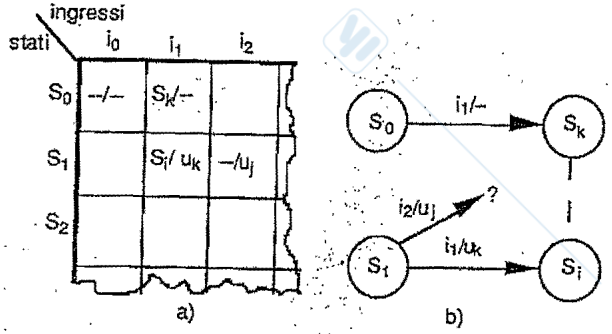


Fig. 1.1: Tabella e grafo di stato per macchina di Mealy

In fig. 1.1 sono abbozzate una tabella di stato (o delle transizioni) ed un *diagramma di stato*, ad essa equivalente, per una macchina di Mealy; è stato in particolare indicato il simbolismo adoperato per una transizione completamente specificata (da S_1, i_1 a S_i con uscita u_k) e una per la quale l'uscita è non specificata (da S_0, i_1), una con stato seguente non specificato (da S_1, i_2) e

una completamente non specificata (da S_0, i_0). Per il modello di Moore le uscite si pongono in corrispondenza delle righe della tabella e all'interno dei nodi del grafo.

Due macchine completamente specificate che hanno il medesimo comportamento terminale (a medesime sequenze di ingresso corrispondono medesime sequenze di uscita), si dicono *equivalenti*. Due macchine, M e M', sono equivalenti se per ogni stato S di M ne esiste almeno uno S' di M' ad esso equivalente, nel senso che per ogni ingresso i due stati presentano la medesima uscita e stati seguenti equivalenti, e viceversa (per ogni stato di M' ne esiste almeno uno di M). Le due macchine possono essere costituite da un numero diverso di stati.

Si può dunque così procedere per la cosiddetta *minimizzazione degli stati*, cioè per la ricerca di una macchina equivalente a quella data M, ma con il minimo numero di stati interni:

- si individuano in M le classi di equivalenza degli stati;
- si costruisce una macchina M' che associ ad ogni classe uno stato; essendo questi equivalenti, ne è definita per ciascun ingresso lo stato seguente e l'uscita (in quanto equivalenti, gli stati della classe hanno stati seguenti equivalenti, e quindi corrispondenti ad un'altra classe, ed una unica uscita),

Per le macchine incomplete, il concetto di equivalenza si trasforma in quello di compatibilità fra stati ed inclusione fra macchine. In particolare:

- due stati sono *compatibili* se presentano uguali uscite e stati seguenti compatibili per quegli ingressi in cui le funzioni τ e ω sono specificate;
- la macchina M' *include* la macchina M se M' ha il medesimo comportamento di M, limitatamente ai comportamenti specificati di quest'ultima.

La costruzione della macchina minima si basa sulla ricerca delle *classi massime di compatibilità* e sulla loro associazione agli stati della macchina.

2. Reti asincrone

Una macchina sequenziale è una macchina *fondamentale o asincrona* se la sua tabella di stato gode di particolari proprietà. Per definire tali proprietà si introduce il concetto di *stato stabile*: lo stato S è stabile per l'ingresso i se è $\tau(S,i)=S$, cioè se per l'ingresso i lo stato seguente è ancora S (lo stato stabile è segnato con una circonferenza sulla tabella).

Una macchina è asincrona se la tabella è tale che, in ogni colonna vi è almeno uno stato stabile e, a partire da uno stato stabile, ogni transizione (ottenuta per una variazione degli ingressi) fa terminare la macchina ancora in uno stato stabile, così come esemplificato in fig. 2.1a) per una transizione

"semplice" (ciclo di lunghezza 1) e in fig. 2.1b) per un ciclo di "lunghezza 3". La tabella della macchina viene "letta" nella continuità del tempo: in ogni istante la macchina è in un punto interno della sua tabella, si sposta in orizzontale sulla stessa per una variazione degli ingressi e quindi si sposta verticalmente fino a raggiungere lo stato stabile.

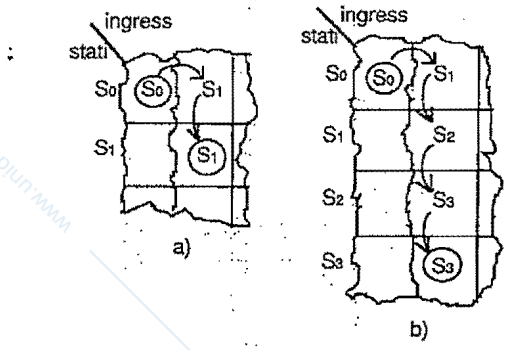


Fig. 2.1: Transizioni fra gli stati in una macchina asincrona

La sincronizzazione della macchina asincrona le deriva, dunque, da proprietà intrinseche della sua stessa tabella e non da artifici esterni: raggiunto lo stato stabile, la macchina si " autosostiene" per la permanenza dell'ingresso e per effetto delle retroazione.

Una *rete sequenziale* è una rete logica che realizza fisicamente una macchina sequenziale. Esistono due principali modelli costruttivi di una rete asincrona: quello fondamentale e quello con flip-flop sulle linee di reazione.

Modello fondamentale (fig. 2.2)

Il modello fondamentale consta soltanto di una rete combinatoria e di ritardi Δ sulle linee di reazione, che a seconda dei casi sono pure idealizzazioni oppure ritardi fisici imposti ad arte.

Il progetto di una rete siffatta è soggetto a molti vincoli:

- la tabella deve essere asincrona;
- l'assegnazione degli stati deve essere priva di corse;
- la rete combinatoria deve essere priva di alee;
- in presenza di alee essenziali (cfr. § II-5) i ritardi Δ devono essere appositi ritardi fisici.

Si ricorda inoltre che:

- i segnali devono permanere un tempo tale da garantire che le transizioni avvengano tra stati stabili;

- il posizionamento iniziale della rete in uno stato specifico può essere associato ad uno specifico segnale e deve essere opportunamente tenuto in conto nella fase di progettazione della rete combinatoria della macchina.

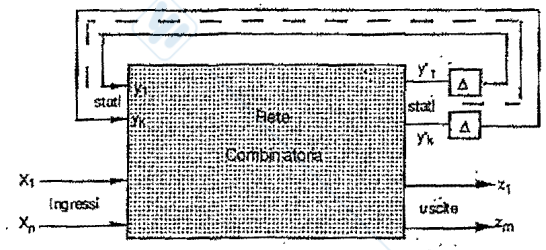


Fig. 2.2: Modello fondamentale di macchina asincrona

Modello con flip-flop (fig.2.3)

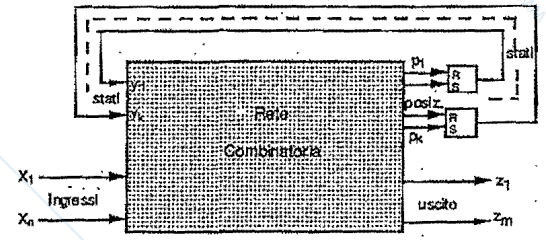


Fig.2.3: Modello sequenziale con flip-flop RS

Il modello consta di una rete combinatoria e di flip-flop RS fondamentali che memorizzano le variabili di stato. I flip-flop risolvono tutti i problemi di alee e, pertanto, rendono meno problematico il progetto.

da	a	R	S
0	0	–	0
0	1	0	1
1	0	1	0
1	1	0	–

Fig. 2.4: Tabella per il progetto del set e reset del flip-flop RS

Per la progettazione della rete occorre costruire i circuiti per 2n "variabili di posizionamento" (set e reset) degli n flip-flop di stato; queste devono essere tali da condurre, per ciascun ingresso, ciascun flip-flop allo stato seguente, in funzione del suo stato precedente e degli ingressi, secondo la tabella di figura 2.4.

Il posizionamento iniziale della rete può essere fatto posizionando opportunamente i flip-flop mediante un segnale esterno.

3. Riconoscitore di parità

Tipo di circuito: rete sequenziale asincrona
Obiettivo: progetto asincrono; minimizzazione degli stati

Testo

Realizzare una rete con due ingressi A e B e due uscite Y e Z che fornisca in uscita:
- Y=1 se l'ultimo segnale variato è A;
- Y=0 se l'ultimo segnale variato è B;
- Z=1 se il numero di variazioni complessive (A e/o B) è dispari;
- Z=0 se il numero di variazioni complessive è pari.

Tabella di transizione

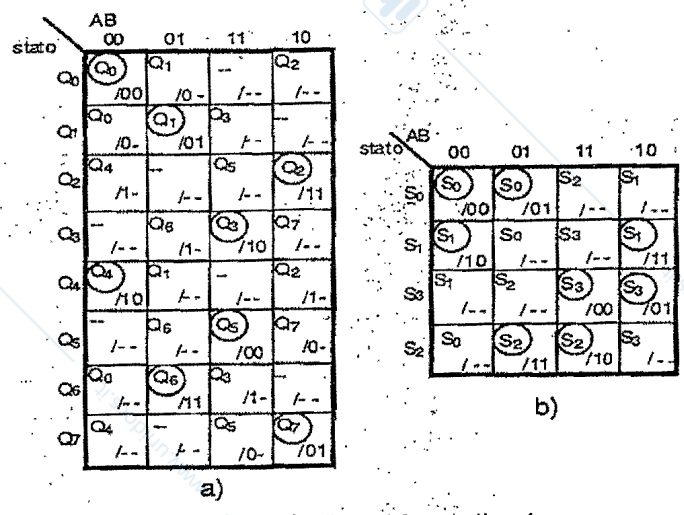


Fig. 3.1: Tabella di transizione degli stati

Dalle specifiche del problema si trae la tabella di transizione di figura 3.1a), costruita nella forma "primitiva", cioè con un solo stato stabile per ogni riga. Questa tecnica descrive il problema in forma estremamente analitica, rinviando ogni problema di sintesi alla successiva minimizzazione degli stati. E' la tecnica da seguire sempre che non si abbiano idee già chiare sulla individuazione di stati equivalenti.

Dalla tabella si evincono i seguenti insiemi di stati compatibili:

$\{Q_0, Q_1\}$ $\{Q_2, Q_4\}$ $\{Q_3, Q_6\}$ $\{Q_5, Q_7\}$

che sono associati rispettivamente agli stati S_0, S_1, S_2 ed S_3 della macchina minimizzata. Ne risulta quindi la tabella minimizzata di figura 3.1b).

Definizione degli stati

La rete ha quindi quattro stati significativi ai quali, riconsiderando le specifiche, può essere dato il seguente significato:
- S_0 (S_3): stato per il quale l'ultima variazione è stata di B con A=0 (A=1);
- S_1 (S_2): stato per il quale l'ultima variazione è stata di A con B=0 (B=1);

mentre la parità delle variazioni complessive risulta ovviamente funzione solo del valore degli ingressi: supposta pari all'inizio per A=B=0 è $A \oplus B$.

Codifica degli stati e progetto combinatorio

Considerando che gli stati della macchina sono quattro e che ogni stato ha al più due stati adiacenti per la codifica, si possono utilizzare 2 bit (q_1q_0). Si è adottata la seguente codifica: 00 per S_0, 01 per S_1, 10 per S_2, 11 per S_3.

A partire dalla tabella di figura 3.1b) è possibile ricavare la rete combinatoria che permette di generare le variabili di stato e le uscite in funzione degli ingressi e dello stato precedente. Nelle figura 3.2 sono riportate le mappe di Karnaugh utili per determinare le funzioni booleane delle variabili di stato e di uscita.

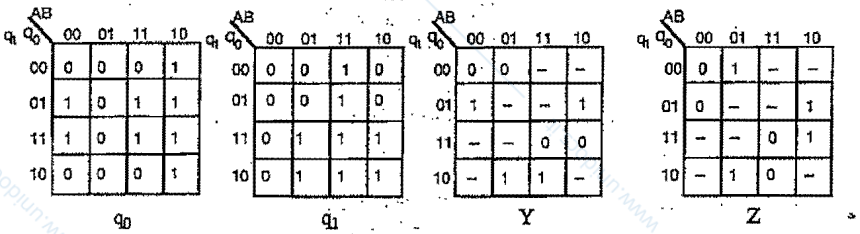


Fig. 3.2: Mappa di Karnaugh per le variabili di stato e di uscita

Dalle mappe riportate si ottengono le funzioni booleane del progetto combinatorio (nelle formule si sono evidenziati con un apice l'uscita delle funzioni booleane relative alle variabili di stato):

$$q_0' = A \cdot q_0 + \overline{B} \cdot q_0 + A \cdot \overline{B}$$
$$q_1' = B \cdot q_1 + A \cdot q_1 + A \cdot B$$
$$Y = \overline{q_0} \cdot q_1 + q_0 \cdot \overline{q_1}$$
$$Z = A \cdot \overline{B} + \overline{A} \cdot B$$

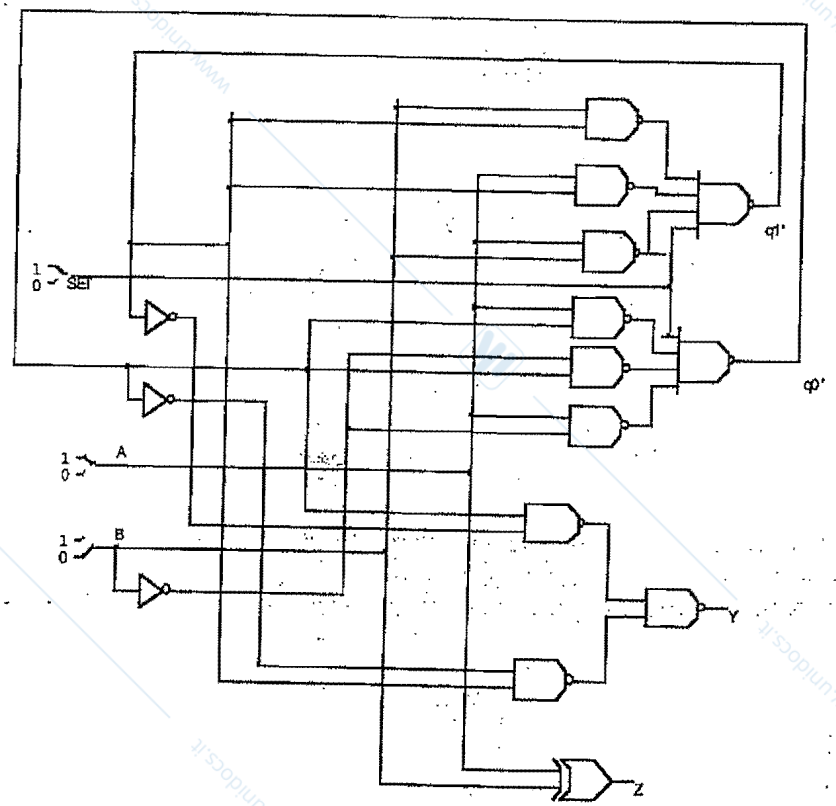


Fig. 3.3: Realizzazione circuitale mediante NAND e XOR

Descrizione del circuito (fig. 3.3)

In figura è riportata una realizzazione circuitale della macchina ottenuta avendo realizzato le funzione booleane descritte mediante porte NAND e XOR, per cui le funzioni precedentemente ricavate sono state trasformate in:

$$q_0' = (A \uparrow q_0) \uparrow (\overline{B} \uparrow q_0) \uparrow (A \uparrow \overline{B})$$
$$q_1' = (B \uparrow q_1) \uparrow (A \uparrow q_1) \uparrow (A \uparrow B)$$
$$Y = (\overline{q_0} \uparrow q_1) \uparrow (q_0 \uparrow \overline{q_1})$$
$$Z = A \oplus B$$

Nel circuito si evidenzia la presenza di un segnale di SET di tipo 0-attivo, che porta il sistema nello stato 11 nella fase di configurazione iniziale. Si fa presente che tale scelta è del tutto arbitraria: sarebbe infatti lecito portare il sistema nella fase iniziale in uno stato qualsiasi di quelli previsti dalla tabella di stato della macchina.

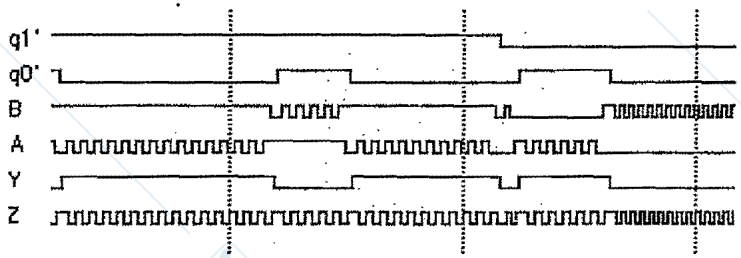


Fig. 3.4: Tempificazione del circuito

4. Riconoscitore di sequenze

Tipo di circuito: rete sequenziale asincrona
Obiettivo: progetto asincrono; eliminazione delle corse

Testo

Si realizzi una rete con due ingressi X e Y ed un'uscita Z tale che:

- Z si alza solo se si alza Y dopo che si è verificata la sequenza 1-0-1 sull'ingresso X (ed X è ancora alto). La variazione 1-0-1-0-1 di X deve essere considerata come una doppia variazione ammissibile;
- Z si abbassa quando ritorna la configurazione X=Y=0.

Tabella di transizione degli stati

Un riconoscitore di sequenza, come quello in esame, è tipicamente caratterizzato da almeno tanti stati quanti sono necessari per individuare tutte le sequenze parziali incluse in quella da riconoscere. Si può allora iniziare il progetto in esame individuando i cinque stati di cui alla figura 4.1, ciascuno dei quali individua una sequenza di lunghezza diversa: A di lunghezza 1, con x=0 ed y qualsiasi (y=-), B di lunghezza 2 (x=01, y qualsiasi) e così via.

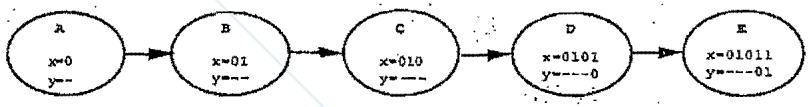


Fig. 4.1 Alcuni stati del riconoscitore

Le specifiche del sistema permettono di identificare alcuni stati significativi attraverso i quali la macchina deve evolvere. Tali stati sono direttamente determinabili considerando le fasi in cui si articola il riconoscimento della stringa di ingresso.

Con lo sviluppo della tabella (cfr. fig. 4.2) si ottengono la transizione fra gli stati e l'individuazione di un nuovo stato (F), dopo che è stato Z=1; a partire da questo, se si abbassa X occorre tornare allo stato C poiché è iniziata una nuova sequenza (10 ...) e non andare ad A come da E.

Nella tabella sono evidenziati gli stati stabili e risulta agevole verificare che per ogni ingresso la macchina termina in uno stato stabile. I punti di non specificazione sono relativi a transizioni che richiederebbero una doppia variazione del segnale di ingresso e quindi un'alea multipla: per le macchine asincrone si ipotizza che ciò non avvenga. La macchina descritta in tabella è una macchina di Moore. Una forma equivalente di descrizione delle transizioni tra stati può essere data mediate un grafo di transizione, in cui gli stati stabili sono evidenziati da autoanelli (cfr. fig. 4.2).

stato	xy 00	01	11	10	Z
A	(A)	(A)	B	B	0
B	C	C	(B)	(B)	0
C	(C)	(C)	B	D	0
D	C	–	E	(D)	0
E	A	(E)	(E)	F	1
F	C	–	E	(F)	1

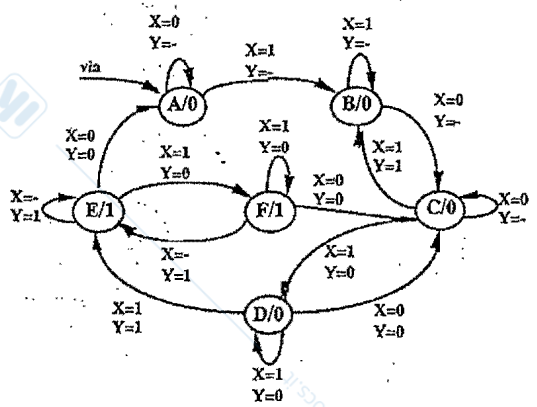


Fig. 4.2: Tabella di transizione degli stati e grafo corrispondente

Codifica degli stati

Il numero di stati presenti in tabella risulta essere minimo e si può procedere alla codifica degli stati, che deve essere tale da garantire l'assenza di fenomeni di alee per corse; per i 6 stati occorrono almeno 3 variabili. Esaminando il grafo delle transizioni, si può notare che lo stato E dista da C sia un numero pari di transizioni (E-F-C o E-D-C) che un numero dispari (E-A-B-C) per cui non è possibile una codifica che non causi variazioni contemporanee di più di una variabile di stato senza aggiungere uno stato fittizio. Si aggiunge, pertanto, tra A e B uno stato instabile S che rende sempre pari il numero di transizioni tra E e C.

Indicando nell'ordine come q_2, q_1, q_0 le variabili di stato, si ottiene la seguente codifica:

A=(100) B=(001) C=(011) D=(010)
E=(110) F=(111) S=(000)

Da cui si ottiene la tabella di transizione ed il grafo di figura 4.3.

Si noti che l'inserimento del nuovo stato richiede che il segale di ingresso, dovendo garantire la transizione tra stati stabili (A e B), debba permanere un tempo maggiore.

stato	xy 00	01	11	10	Z
A (100)	(A)	(A)	S	S	0
B (001)	C	C	(B)	(B)	0
C (011)	(C)	(C)	B	D	0
D (010)	C	–	E	(D)	0
E(110)	A	(E)	(E)	F	1
F(111)	C	–	E	(F)	1
S(000)	--	–	B	B	0

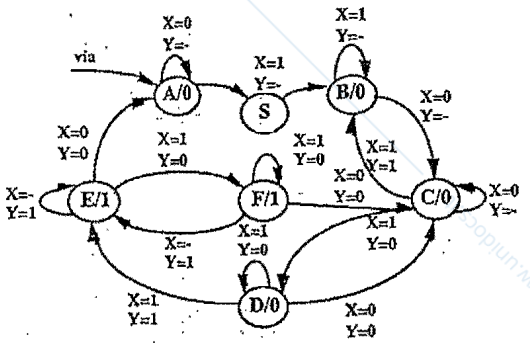


Fig. 4.3 Tabella di transizione e grafo completi

Progetto combinatorio

Dalla tabella di transizione definitiva possiamo trarre le tabelle di verità, che ci consentiranno di trovare le funzioni booleane delle variabili di stato. Queste sono funzioni di 5 variabili per cui si sono adoperate due mappe delle 4 variabili x, y, q_1, q_0, ciascuna per un diverso valore di q_2 (fig.4.4).

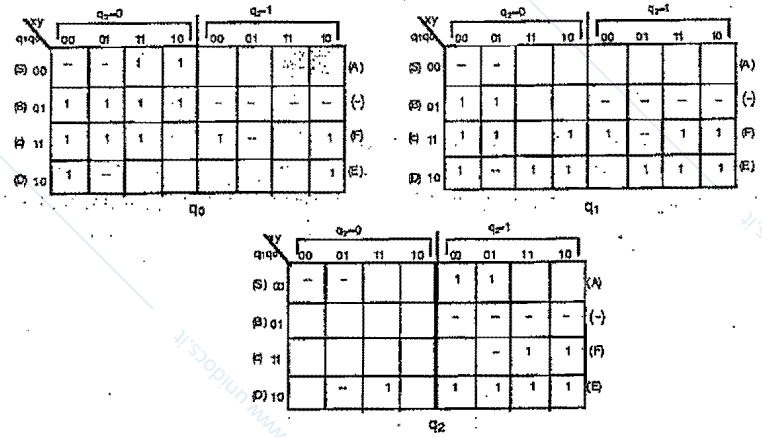


Fig. 4.4: Tabella di verità per le variabili di stato

Dalle mappe si ottengono le funzioni che seguono:

$$q_0' = \overline{q_1}\overline{q_2} + \overline{X}\overline{q_2} + Yq_0\overline{q_2} + \overline{Y}q_0q_2 + X\overline{Y}q_1q_2 + \overline{X}q_0$$
$$q_1' = \overline{X}q_0 + \overline{X}\overline{q_2} + \overline{q_0}q_1\overline{q_2} + X\overline{Y}q_1 + q_0q_2 + Yq_1q_2 + Xq_1q_2 + Xq_0\overline{q_1}$$
$$q_2' = Yq_0\overline{q_1} + \overline{X}\overline{q_0}q_2 + Xq_1q_2$$

Le alee statiche, che nella rete sequenziali potrebbero indurre fenomeni di oscillazione, sono state eliminate considerando una ricopertura della funzione che includesse tutti gli implicanti che contengono transizioni tra ingressi adiacenti per i quali l'uscita è 1. A tale scopo è stato necessario aggiungere gli implicanti ridondanti $\overline{X}q_0$ per la copertura della funzione di posizionamento di q_0 e $\overline{X}q_0$ e $Xq_0\overline{q_1}$ per la copertura di q_1.

La forma NAND delle funzioni che sarà usata per l'implementazione è direttamente ricavabile da quella and-or di cui sopra:

$$q_0' = (\overline{q_1}\uparrow\overline{q_2})\uparrow(\overline{X}\uparrow\overline{q_2})\uparrow(Y\uparrow q_0\uparrow\overline{q_2})\uparrow(\overline{Y}\uparrow q_0\uparrow q_2)\uparrow$$
$$\uparrow(X\uparrow\overline{Y}\uparrow q_1\uparrow q_2)\uparrow(\overline{X}\uparrow q_0)$$
$$q_1' = (\overline{X}\uparrow q_0)\uparrow(\overline{X}\uparrow\overline{q_2})\uparrow(\overline{q_0}\uparrow q_1\uparrow\overline{q_2})\uparrow(X\uparrow\overline{Y}\uparrow q_1)\uparrow(q_0\uparrow q_2)\uparrow$$
$$\uparrow(Y\uparrow q_1\uparrow q_2)\uparrow(X\uparrow q_1\uparrow q_2)\uparrow(X\uparrow\overline{q_0}\uparrow q_1)$$
$$q_2' = (Y\uparrow\overline{q_0}\uparrow q_1)\uparrow(\overline{X}\uparrow\overline{q_0}\uparrow q_2)\uparrow(X\uparrow q_1\uparrow q_2)$$

L'uscita Z è alta solo quando ci troviamo negli stati E ed F, avendo progettato la rete come una macchina di Moore, si ricava facilmente la funzione Z:

$$Z = q_1 \cdot q_2$$

oppure, in forma NAND:

$$Z = \overline{(q_1\uparrow q_2)}$$

Descrizione del circuito (fig. 4.5)

La rete che implementa la macchina è riportata in figura con il relativo diagramma di tempificazione.

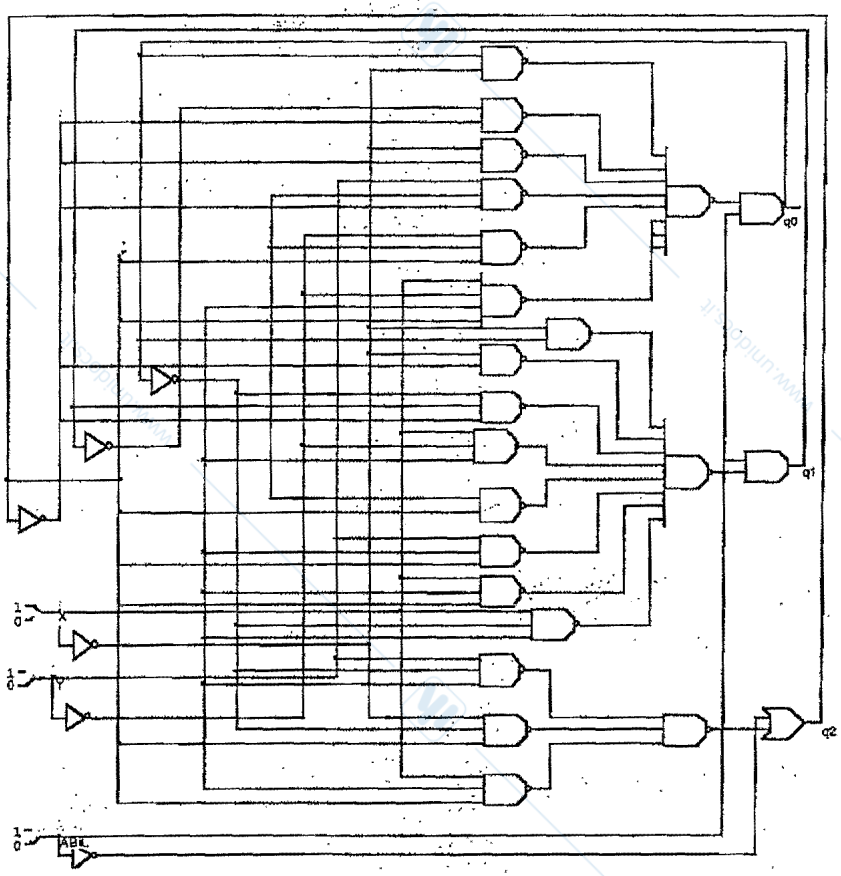


Fig. 4.5a: Rete asincrona e diagramma di tempificazione

Nel circuito si è reso disponibile un segnale di "reset" iniziale (ABIL), che porta la rete nello stato A. Essendo 100 il codice di A, ABIL resetta la rete semplicemente ponendo: q_2=1, q_1=0, q_2=0; ponendo allora:

$$q_2 = q_2' + \overline{ABIL}$$
$$q_1 = q_1' \cdot ABIL$$
$$q_0 = q_0' \cdot ABIL$$

le variabili di stato restano quelle di cui sopra per ABIL=1, mentre assumono il valore di A per ABIL=0.

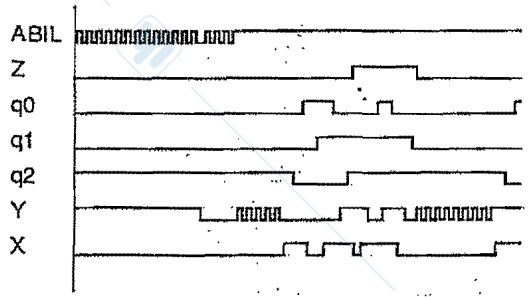


Fig. 4.5b : Diagramma di tempificazione

5. Interruttore ideale

Tipo di circuito: rete sequenziale asincrona
Obiettivo: progetto asincrono

Testo

Un interruttore reale (fig. 5.1) è soggetto al fenomeno del rimbalzo: spostandosi la lamella dal contatto A a B, rimbalza su B aprendo e chiudendo il relativo contatto.

Progettare un dispositivo logico che trasformi in ideale l'interruttore: fornendo un segnale di uscita stabile: il contatto si chiuda non appena la lamella perviene per la prima volta su B.

Impostazione del progetto

Posto uguale ad 1 un contatto chiuso e detto Z il segnale di uscita dal dispositivo che indichi se la lamella è in A (Z=0) o in B (Z=1), il diagramma temporale di fig. 5.1 suggerisce immediatamente la soluzione: il dispositivo è un flip-flop RS con A segnale di reset e B segnale di set.

Si suppone ora di non aver visto subito la soluzione e si procede con la tecnica di progetto delle reti asincrone: progettare una rete con due ingressi a livelli A e B ed un'uscita pure a livelli Z che si alza la prima volta che si alza

B con A basso e si abbassa la prima volta che si alza A con B basso. Sugli ingressi vale la condizione di vincolo A·B=0, poiché è impossibile che il contatto elettrico si trovi contemporaneamente nei punti A e B.

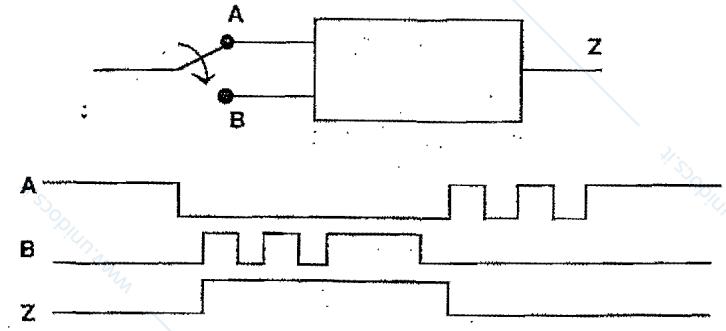


Fig. 5.1: Interruttore ideale

Diagramma temporale

Riprendendo il diagramma di fig. 5.1 si possono assegnare gli stati associati a ciascuna situazione verificatesi nell'evolversi della sequenza, come esemplificato in figura 5.2.

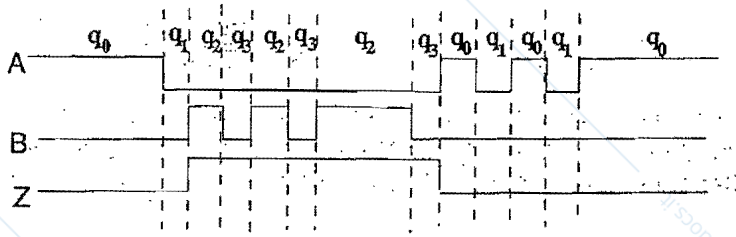


Fig.5.2: Stati nella sequenza

Assegnando agli stati il seguente significato:

q_0: interruttore in A;
q_1: commutazione da A a B o rimbalzo;
q_2: interruttore in B;

q_3: commutazione da B a A o rimbalzo;

si ottiene la tabella di stato di figura 5.3.

stato \ AB	00	01	11	10
q_0	q_1	—	—	(q_0)
q_1	(q_1)	q_2 /-	—	q_0
q_2	q_3 /1	(q_2) /1	—	—
q_3	(q_3)/1	q_2 /1	—	q_0/-

Fig. 5.3: Tabella di stato

In tabella si sono considerati gli ingressi A=1, B=1 non fisicamente irrealizzabili, per cui la 3° colonna (A·B=1) è riempita con tutti don't care e si considera Z=1 se l'interruttore è in B.

Si ha, inoltre, che:
- $\tau(q_1, 10)=q_0$: l'interruttore non ha il tempo di commutare su B perché c'è il nuovo comando su A (rimbalzo);
- $\tau(q_3, 01)=q_2$: è il caso duale di $\tau(q_1, 10)=q_0$.

Minimizzazione degli stati

La tabella è incompletamente specificata per cui bisogna determinare gli insiemi compatibili massimi. Da essa si ricavano le seguenti classi di compatibilità:

$$F_{11}=(q_0, q_1);$$
$$F_{12}=(q_2, q_3)$$

A ciascun elemento della famiglia si associa uno stato della macchina ridotta:

$$S_0=F_{11}$$
$$S_1=F_{12}$$

e si ottiene la tabella di figura 5.4.

stato \ AB	00	01	11	10
S_0	S_0	S_1/—	—	S_0
S_1	S_1/1	S_1/1	—	S_0/—

Fig.5.4: Tabella minima

Si noti che:
- è possibile associare agli stati il seguente significato:
  $S_0 \Leftrightarrow$ interruttore ideale in A; $S_1 \Leftrightarrow$ interruttore ideale in B;
- ritardando leggermente la variazione di stato - $\omega(S_0,01)=0$, $\omega(S_1,10)=1$ - la tabella diventa di Moore;
- la tabella è quella del flip-flop RS.

A titolo di esercizio, comunque si procede con il progetto indipendentemente dall'ultima osservazione.

Codifica degli stati

Gli stati sono codificabili con una sola variabile di stato x:

$$S_0 \Leftrightarrow x=0 \qquad S_1 \Leftrightarrow x=1$$

Progetto combinatorio per le variabili di stato

Dalla tabella di stato si possono ricavare le funzioni per la generazione delle variabili di stato (fig.5.5):

x \ AB	00	01	11	10
0		1	—	
1	1	1	—	

Fig.5.5: Progetto della rete

da cui si ottiene per la variabile di stato (e per l'uscita che coincide con essa):

$$x' = B + \overline{A} \cdot x$$

Il circuito è quello del flip-flop RS dinamico; esso coincide infatti con l'equazione di stato di detto flip-flop:

$$F' = S + \overline{R} \cdot F$$

Descrizione del circuito (fig. 5.6)

Il circuito (flip-flop dinamico) è mostrato in figura. Si ricorda tuttavia che il problema specifico si risolve meglio con un normale flip-flop RS.

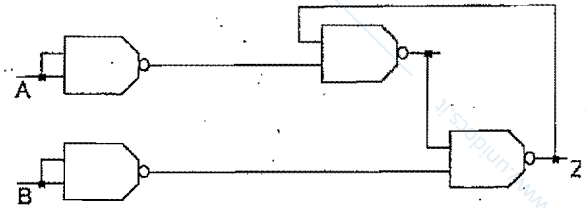


Fig. 5.6: Rete dell'interruttore ideale

6. Simulatore di ritardo inerziale

Tipo di circuito: rete sequenziale asincrona
Obiettivo: approfondimento teorico; progetto asincrono

Testo

Progettare una macchina sequenziale che simuli il comportamento fisico di un sistema avente ritardo inerziale di durata T.

Impostazione del progetto

Detto S il segnale di ingresso, la macchina possiede un'uscita U che segue S dopo un tempo T, sempre che il nuovo valore di S sia rimasto costante almeno per un tempo T; tutte le variazioni di S che durano meno di T sono ignorate in uscita. Un esempio è riportato in figura 6.1, ove il ritardo T è posto uguale a 2 u.t. e sono ignorate le variazioni che durano 1 u.t.

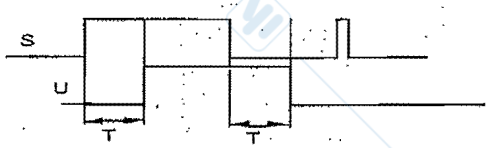


Fig. 6.1: Ritardo inerziale

Per generare l'uscita U occorre comunque generare un segnale R=ΔS, la replica di S ritardata di un tempo T. L'uscita U sarà posta uguale ad R se la variazione di S è durata almeno T u.t., resterà inalterata altrimenti. La sequenza dei valori assunti dalla coppia (S, R) è peraltro indicativa del tempo in cui un valore di S sia rimasto costante, in quanto istante per istante la rete "vede" S e quello che S era T u.t. prima.

Ad esempio, la sequenza SR= 00→10→00→01 indica che S è tornato a 0 prima che R diventasse 1, quindi prima di T u.t. dalla variazione 0→1 di S: questa variazione non va considerata e U permane 0. Invece, la sequenza SR=00→10→11 indica che il valore 1 di S è rimasto tale per T u.t.. Più in generale, una doppia variazione di S mentre R rimane costante equivale ad una variazione avvenuta in un intervallo inferiore al ritardo inerziale T.

La rete è ovviamente sequenziale in quanto deve almeno memorizzare l'uscita da mantenere ed asincrona in quanto a livelli ed operante nella continuità del tempo.

Tabella delle transizioni

Detti A, B, C e D gli stati, la tabella delle transizioni è mostrata in fig. 6.2. La macchina è una macchina di Moore.

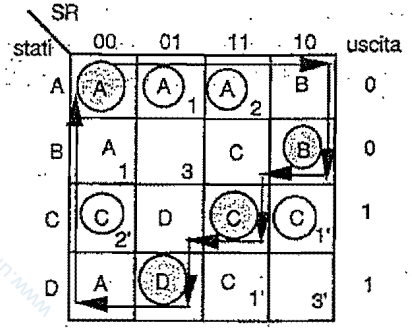


Fig.6.2: Simulazione ritardo inerziale: tabella di stato

Sulla tabella sono prima riportate le transizioni corrispondenti alle sequenze indicative di variazioni che superano l'inerzia del circuito (evidenziate con le frecce e con gli stati stabili anneriti):
- a partire dallo stato (A, SR=00): SR=00→10→11
- a partire dallo stato (C, SR=11): SR=11→01→00

e quindi le transizioni di S ignorate (evidenziate con 1, 1');
- a partire dallo stato (B, SR=10): SR=10→00→01
- a partire dallo stato (D, SR=01): SR=01→11→10

Sono infine considerati gli altri punti della tabella:
- Il punto 2 (SR=11), al quale si arriva da SR=01, indica che S si alza quando R non ha ancora sentito il valore S=0 (è infatti ancora 1, un vecchio valore di S); se R si abbassa prima di S (SR=10, fig.6.3a) si va nello stato B di attesa dell'evento SR=11, come se si provenisse da SR=00; se viceversa si abbassa prima S (SR=01, fig.6.3b), vuol dire che c'è stata un'altra variazione "corta" di S e si ritorna in A. Dualmente si possono ripetere le osservazioni per 2'.
- Il punto 3 indica una variazione SR=10→01: indica cioè che S varia T u.t. dopo la precedente variazione; è una condizione limite, per la quale si può porre indifferente l'uscita alta o bassa. Dualmente si possono ripetere le osservazioni per 3'.

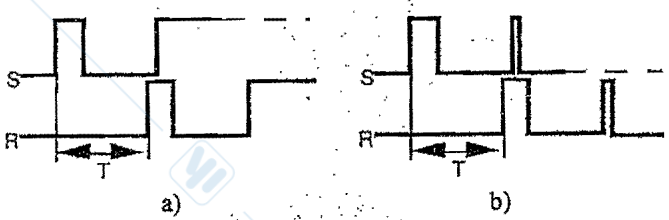


Fig.6.3: possibili variazioni di S

Codifica degli stati

Dette y_0, y_1 nell'ordine le variabili di stato, si effettua la seguente assegnazione priva di corse:

A= (00); B=(01); C=(11); D=(10)

Da questa si ottiene la tabella di stato:

$y_0 y_1$ \ SR	00	01	11	10	uscite (Z)
00	00	00	00	01	0
01	00	--	11	01	0
11	11	10	11	11	1
10	00	10	11	--	1

Dalla tabella si evince che l'uscita z coincide con la variabile di stato y_0.

Progetto combinatorio

Si adotta il modello di rete con flip-flop RS sulle linee di reazione: occorre dunque progettare i segnali di set e reset per ciascun flip-flop: R_1, S_1 per y_1 e R_0, S_0 per y_0 (fig.6.4).

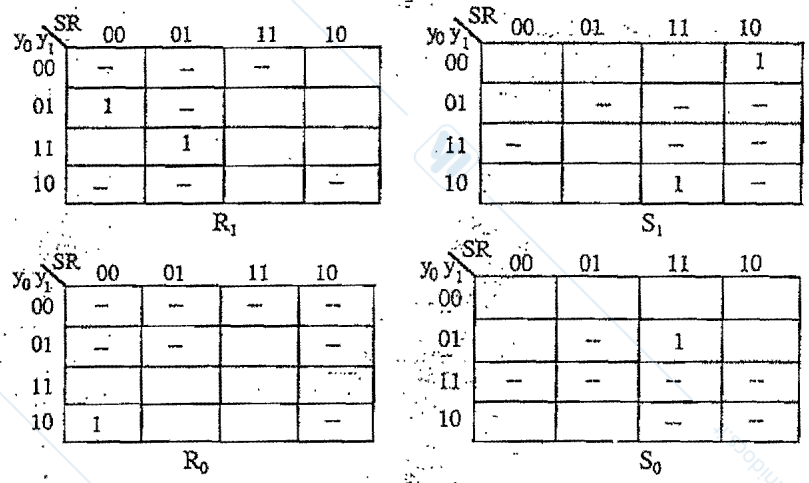


Fig.6.4: Progetto per il posizionamento

Si ottiene dunque:

$$R_1 = \overline{S} \cdot R + \overline{y_0} \cdot \overline{S} \qquad S_1 = S \cdot \overline{R} + S \cdot y_0$$

$$R_0 = \overline{R} \cdot \overline{y_1} \qquad S_0 = R \cdot y_1$$

Descrizione del circuito (fig. 6.5)

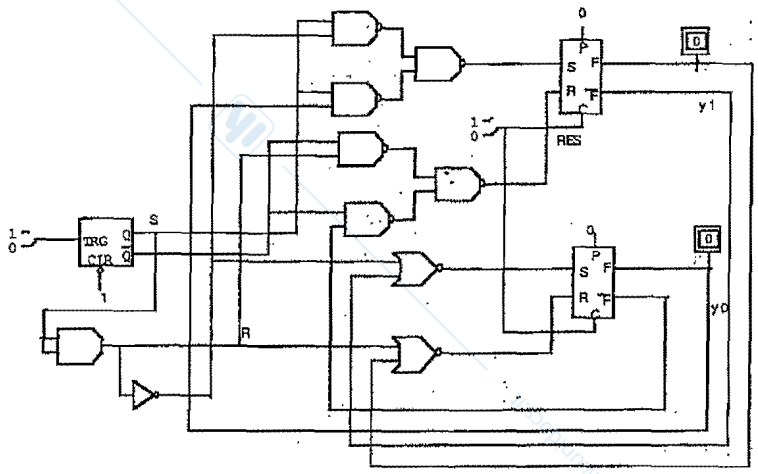


Fig. 6.5a: Simulatore di ritardo inerziale

Il segnale ritardato R è ottenuto ponendo $R = S \cdot S$ e sfruttando il ritardo della porta AND. Per la generazione del segnale S si è usato un monostabile con ritardo programmabile, in modo da poter provare il comportamento del circuito per impulsi di diversa ampiezza. Si ricorda che l'uscita coincide con y_0. Il sistema dispone, inoltre, di un segnale di reset (RES) che porta il sistema nello stato 00.

Nella fase di verifica del circuito, al fine di considerare differenti condizioni di funzionamento del dispositivo, si sono generati i segnali S ed R mediante dei switch. Tale configurazione ha permesso di analizzare il comportamento della macchina per differenti segnali di ingresso S ed R senza variare il ritardo della porta AND usata per la generazione di R.

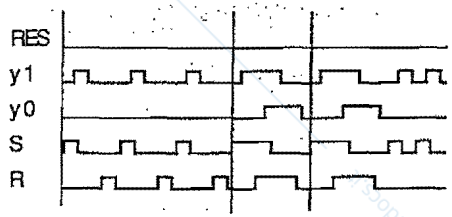


Fig. 6.6: Simulatore di ritardo inerziale: tempificazione

Nella figura 6.6 è illustrato un diagramma di tempificazione della macchina per una possibile tempificazione del segnale di ingresso. Le linee verticali individuano le variazioni del segnale S che hanno una durata maggiore ritardo maggiore del ritardo inerziale e che sono, pertanto, riportate in uscita (y_0) con ritardo T.

Tempificazione

I segnali di ingresso alla macchina asincrona, in accordo con la teoria generale, devono permanere un tempo sufficiente a garantire che la transizione della macchina avvenga tra stati stabili. Per la rete in esame le transizioni tra stati stabili avvengono in un sol passo, per cui considerando i ritardi della rete combinatoria con l'aggiunta della porte di negazione per S, si ha che l'ingresso deve permanere più di 3 u.t.. Ne deriva che non è possibile simulare ritardi inerziali con tempi minori di 3, poiché in tal caso le variazioni dei segnali S ed R di ingresso alla rete avverrebbero in un tempo inferiore al tempo necessario per garantire l'evoluzione della macchina tra stati stabili. Ad esempio, per un ritardo T di 2 u.t. non si ha un corretto funzionamento del circuito poiché la macchina asincrona non riesce a portarsi in uno stato stabile ricevendo due transizioni dell'ingresso (prima S e poi R) distanti 2 u.t. (tempo inferiore al ritardo della rete combinatoria della macchina asincrona). Se aumenta il tempo di ritardo T (ad esempio a 4 u.t.) il circuito funziona correttamente poiché la variazione dell'ingresso avviene in un tempo tale da garantire che le transizioni avvengano tra stati stabili.

7. Flip-flop a tre stati

Tipo di circuito: rete sequenziale asincrona
Obiettivo: progettazione asincrona, assegnazione priva di corse

Testo

Progettare un registro con tre differenti stati di uscita, pilotato da tre segnali A, B e C, che lo posizionano rispettivamente in uno dei tre stati.

Per analogia al flip-flop, il circuito è anche detto "flip-flop a 3 stati". Esso può essere utile in operazioni logiche o aritmetiche; ad esempio quando è necessario ricordare se l'ultimo risultato di un'operazione sia stato maggiore, minore o uguale a 0.

Impostazione del progetto

Per lo svolgimento del progetto si analizzeranno tre soluzioni alternative basate su:
- codifica delle uscite con due variabili;
- codifica delle uscite con due variabili e impiego di flip-flop RS per lo sviluppo del progetto;
- codifica delle uscite con tre variabili.

Prima soluzione: codifica delle uscite con due variabili

Si considerano gli ingressi 0-attivi e mutuamente esclusivi per cui l'ingresso neutro è l'ingresso A=B=C=1. L'uscita viene codificata con due variabili per cui si ha che essa è uguale a:

- 00 se l'ultimo ingresso attivo è stato A (A=0);
- 01 se l'ultimo ingresso attivo è stato B (B=0);
- 11 se l'ultimo ingresso attivo è stato C (C=0).

La macchina è una macchina di Moore (come è ovvio essendo un registro) e si possono identificare i seguenti stati coincidenti con le uscite:
- q_0: uscita 00;
- q_1: uscita 01;
- q_2: uscita 11;

che si codificano con due variabili, y_1 e y_0 nell'ordine. Ne deriva, pertanto, la tabella di figura 7.1.

$y_1 y_0$ \ ABC	000	001	011	010	100	101	111	110
q_0(00)	--	--	(q0)	--	--	q_1	(q0)	q_2
q_1(01)	--	--	q_0	--	--	(q1)	(q1)	q_2
q_2(11)	--	--	q_0	--	--	q_1	(q2)	(q2)

Fig.7.1: Tabella di stato

La codifica adottata in prima istanza produce corse: non tutte le transizioni avvengono tra stati adiacenti ed, essendo 3 gli stati comunque fra loro connessi, non esistono altre codifiche prive di corse su 2 variabili. E', quindi, necessario inserire un ulteriore stato instabile, q_3, codificato con 10, nelle

transizioni tra q_0 e q_2 in modo da "disciplinare" la variazione 00→11 facendola avvenire attraverso 10: 00→10→11 e viceversa. La tabella si modifica in quella di figura 7.2.

$y_1 y_0$ \ ABC	000	001	011	010	100	101	111	110
q_0(00)	--	--	(q0)	--	--	q_1	(q0)	q_3
q_1(01)	--	--	q_0	--	--	(q1)	(q1)	q_2
q_2(11)	--	--	q_3	--	--	q_1	(q2)	(q2)
q_3(10)	--	--	q_0	--	--	--	--	q_2

Fig.7.2: Tabella con assegnazione priva di corse

Dalla tabella di figura 7.2 si ottiene:

$$Y_0 = \overline{B} + A \cdot y_1 + A \cdot y_0$$
$$Y_1 = \overline{C} + y_1 \cdot y_0 \cdot B$$

In figura 7.3 è riportata la rete, realizzata mediante porte NAND.

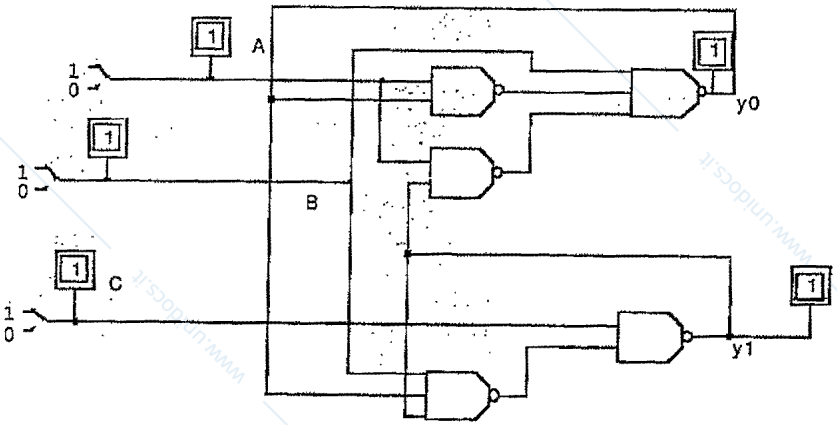


Fig. 7.3a: Flip-flop a tre stati: soluzione 1

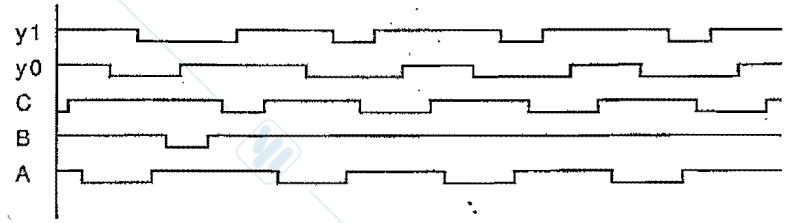


Fig. 7.3b: Flip-flop a tre stati: tempificazione

Nel diagramma temporale si nota che nella transizione da 11 a 00 si ha il passaggio per lo stato intermedio 10, per cui il segnale di ingresso deve durare 4 u. t. in quanto la transizione tra stati stabili avviene in due passi.

Seconda soluzione: impiego di flip-flop RS

Un'altra possibile risoluzione al problema è quella di utilizzare due flip-flop RS per tenere conto delle tre differenti possibilità di memorizzazione. In tal caso si possono facilmente ricavare dalla tabella 7.2 i segnali di posizionamento dei flip-flop in funzione delle variabili di ingresso A, B e C, per le quali si ha che (cfr. tabella per i vincoli di progetto del flip-flop RS):

da	a	R	S
0	0	–	0
0	1	0	1
1	0	1	0
1	1	0	–

Si ha dunque:

$$S_0 = \overline{C} + \overline{B}$$
$$R_0 = \overline{A}$$
$$S_1 = \overline{C}$$
$$R_1 = \overline{A} + \overline{B}$$

Tale macchina è una macchina sequenziale degenere poiché il posizionamento del flip-flop non è funzione dello stato (così come per il flip-flop RS). Alle medesime equazioni si sarebbe arrivati anche sviluppando il progetto come progetto di macchina asincrona con flip-flop sulle linea di retroazione.

Descrizione del circuito (fig. 7.4)

In figura 7.4 è riportato il circuito che implementa la rete secondo lo schema presentato in cui si ricorda che l'uscita coincide con lo stato.

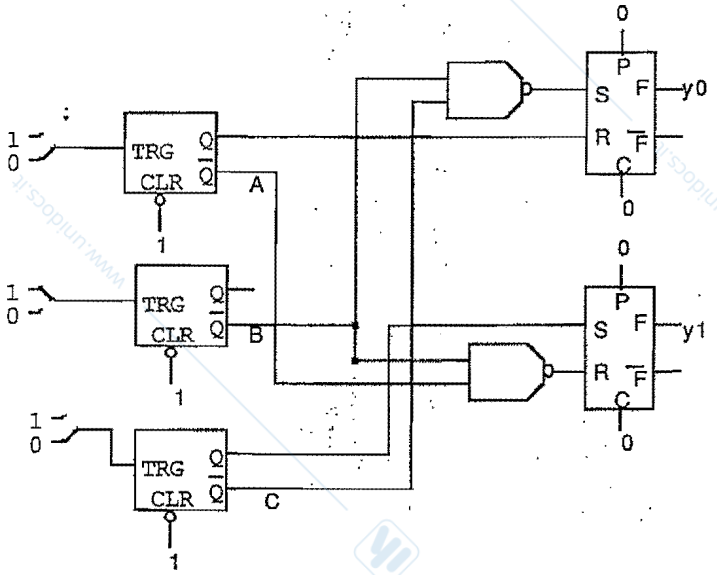


Fig. 7.2a: Flip-flop a tre stati con RS: soluzione 2

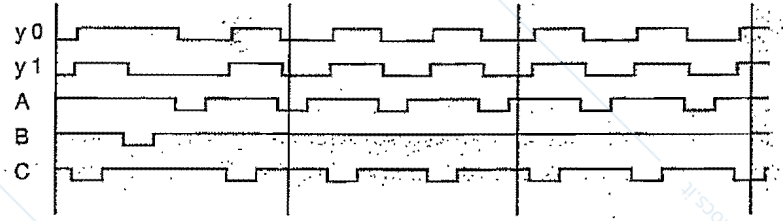


Fig. 7.2b: Flip-flop a tre stati con RS: tempificazione

Nel circuito i segnali A, B e C sono generati mediante un monostabile (si è considerata l'uscita negata in quanto i segnali sono 0-attivi). In tal modo è possibile generare una sequenza di ingressi che rispetta la specifica per la quale un sol segnale alla volta può avere valore pari a 0. Il circuito non dispone, inoltre, di un segnale di *reset* poiché il primo segnale di ingresso

garantisce il corretto posizionamento dei flip-flop indipendentemente dallo stato precedente.

Terza soluzione: impiego di tre variabili di stato

Si considerano gli ingressi 1-attivi e mutuamente esclusivi per cui l'ingresso neutro è l'ingresso A=B=C=0.

Utilizzando tre variabili per la codifica dell'uscita (y_0, y_1, y_2) e, conseguentemente, dello stato, è possibile avere la seguente associazione (considerando anche l'uscita 1-attiva):

- q_1 (001): uscita per A=1;
- q_2 (010): uscita per B=1;
- q_3 (100): uscita per C=1.

Tale associazione risulta simile a quella effettuata nel flip-flop RS per la codifica delle variabili di stato. Nel caso presentato è necessario introdurre uno stato intermedio instabile q_0 (000) per garantire che le transizioni avvengano sempre tra stati adiacenti. Ne deriva, pertanto, la tabella di stato di figura 7.4.

ABC / $y_2y_1y_0$	000	001	011	010	100	101	111	110
q_0 (000)	--	q_3	–	q_2	q_1	–	–	–
q_1 (001)	(q1)	q_0	–	q_0	(q1)	–	–	–
q_4 (011)	–	–	–	–	–	–	–	–
q_2 (010)	(q2)	q_0	–	(q2)	q_0	–	–	–
q_3 (100)	(q3)	(q3)	–	q_0	q_0	–	–	–

Fig.7.4: Tabella di stato: soluzione 3

e le conseguenti funzioni:

$$y_0 = \overline{B}\cdot\overline{C}\cdot\overline{y_2}\cdot\overline{y_1} = \overline{B+C+y_2+y_1}$$
$$y_1 = \overline{A}\cdot\overline{C}\cdot\overline{y_2}\cdot\overline{y_0} = \overline{A+C+y_2+y_0}$$
$$y_2 = \overline{A}\cdot\overline{B}\cdot\overline{y_1}\cdot\overline{y_0} = \overline{A+B+y_1+y_0}$$

Si noti come tale soluzione è migliore della soluzione 1, avendo le tre uscite perfettamente bilanciate e simmetriche. L'espressione ottenuta è simile a quella del flip-flop RS a NOR, che risulta progettato con la stessa metodologia.

Descrizione del circuito (fig. 7.5)

Il circuito è stato realizzato mediante porte NOR. Se si fosse utilizzata una logica 0-attiva si sarebbe pervenuti ad un circuito con lo stesso schema di interconnessione, ma con le porte NAND al posto delle porte NOR. In questo caso le uscite sarebbero 0-attive (011, 101, 110) e sarebbe stato necessario inserire uno stato di transizione instabile 111 per garantire le transizioni tra stati adiacenti.

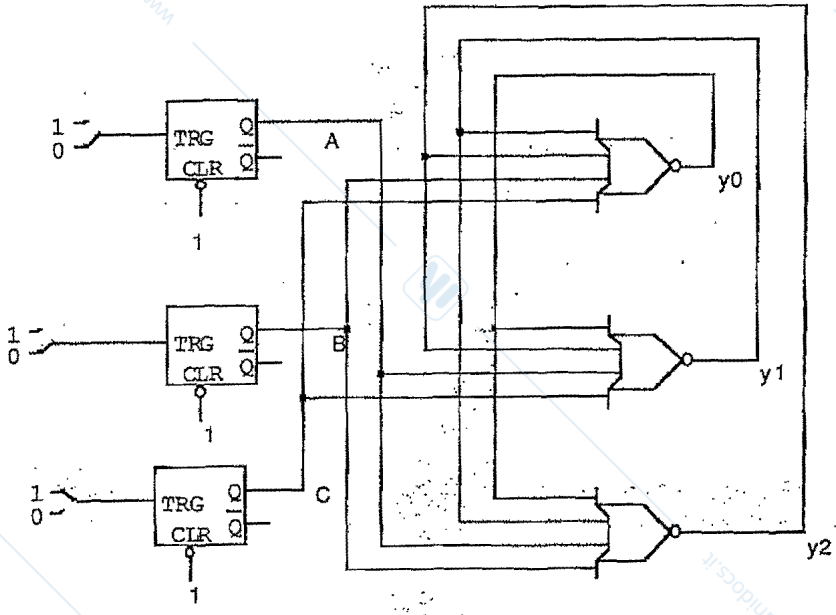


Fig. 7.5a: Flip-flop a tre stati con tre variabili di stato

Nel diagramma temporale le linee verticali indicano le variazioni dello stato dovute rispettivamente ai segnali A, B e C.

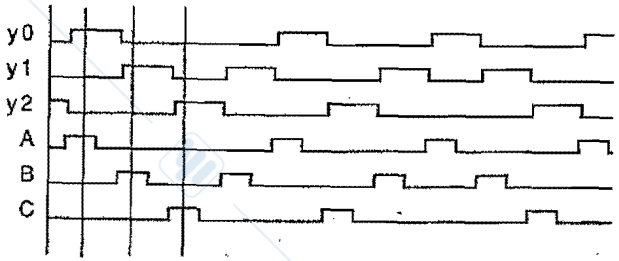


Fig. 7.5b: Flip-flop a tre stati con tre variabili di stato: tempificazione

Capitolo quinto
Reti sincrone

1. Sequenze a livelli e sincrone

Nel progetto delle reti sequenziali, molta importanza assume la tempificazione ed in particolare quella dei segnali di ingresso. Una sequenza di ingresso in una rete è costituita da una sequenza di più segnali binari che, nel loro complesso, costituiscono la sequenza degli *stati di ingresso* alla macchina. Per lo studio della tempificazione occorre far riferimento talora ai singoli segnali che costituiscono l'ingresso (*ingressi binari o variabili booleane*), talora ai valori che può assumere l'ingresso nel suo complesso (*stati di ingresso*). Nella figura 1.1 è rappresentata una sequenza di segnali binari (l_1, l_2, l_3, l_4) e l'associata sequenza Q degli stati di ingresso; questi ultimi sono convenzionalmente individuati attraverso il codice numerico costituito dalle variabili binarie, sicché la sequenza degli stati è 0, 1, 3, etc.

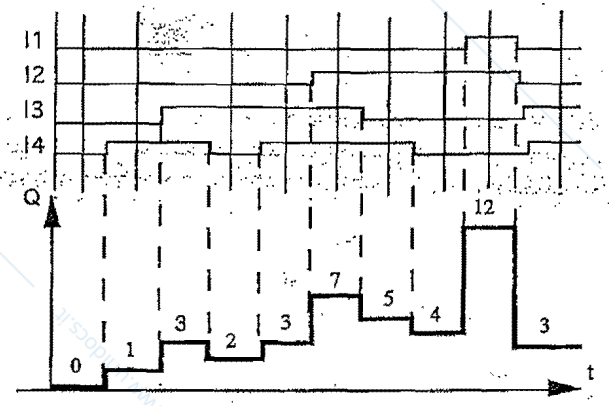


Fig. 1.1: Sequenza a livelli

La sequenza di figura 1.1 è esemplificativa di una classe di possibili sequenze di segnali di ingresso e prende il nome di *sequenza a livelli*. In essa i k stati di ingresso, codificati mediante $n \geq \log_2 k$ segnali binari, sono da ritenere significativi in tutto il periodo di tempo in cui si trovano in ingresso alla rete. Essa è la tipica sequenza di ingresso di una rete sequenziale asincrona, la cui tabella di stato viene letta nella continuità del tempo (cfr. § IV-2): permanendo un determinato ingresso, la rete raggiunge uno stato stabile, ove si "autosostiene" fino ad un ulteriore variazione dell'ingresso. In una sequenza a livelli si devono evitare transizioni multiple (di più di una variabile binaria), che potrebbero essere fonte di alee per la rete: l'ultima transizione esemplificata in figura, dallo stato 12 al 3 è da evitare, poiché ottenuta dalla variazione contemporanea di 4 ingressi binari. La necessità di evitare tali transizioni rende problematica l'adozione delle sequenze a livelli e delle annesse reti asincrone.

Le sequenze di ingresso che considerano come significativo il segnale solo in particolari istanti di tempo prendono il nome di *sequenze impulsive*. In esse è presente almeno un segnale binario che discrimina il tempo in cui considerare significativo l'ingresso. Tale segnale è detto impulsivo.

Dal punto di vista della teoria e del progetto delle reti sequenziali, tuttavia, *un segnale binario non è impulsivo di per sé, ma in quanto inserito in una sequenza impulsiva*. Tali sequenze sono gli ingressi delle *reti sincrone impulsive* (o semplicemente reti sincrone).

Esistono due tipi fondamentali di sequenze impulsive, che danno luogo a due distinti modelli di rete sincrona:

- sequenze (e reti) a sincronizzazione esterna;
- sequenze (e reti) autosincronizzate.

In figura 1.2 è mostrata una *sequenza a sincronizzazione esterna*, la più comunemente usata delle due, costituita da un unico segnale binario impulsivo (x) e da tre segnali binari a livello (l_1, l_2, l_3), per un totale di 16 stati di ingresso (considerando tutte le possibili combinazioni delle 3 variabili a livello con i due valori di quella impulsiva). Il segnale binario x è impulsivo, e la sequenza di stati lo è altrettanto, in quanto, allorché esso è attivo (x=1), nessuno dei segnali binari a livello varia; in corrispondenza del valore attivo di x, lo stato della sequenza si dice *stato impulsivo* (ciò accade per gli stati 1, 7, 9, 11, 5 e 13 dell'esempio). Ne consegue che a sinistra e a destra di uno stato impulsivo la sequenza di stati presenta un unico livello, detto *base* dello stato impulsivo (nell'esempio, 0 è la base di 1, 6 di 7, 8 di 9, 10 di 11, 4 di 5, 12 di 13).

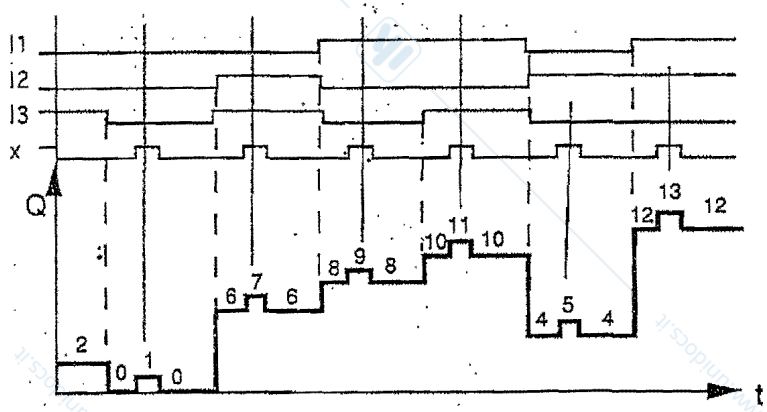


Fig. 1.2: Sequenza impulsiva a sincronizzazione esterna

Dei 16 stati della sequenza esemplificata, 8 sono impulsivi, 8 sono le basi corrispondenti e si distinguono a coppie per la sola presenza o assenza dell'impulso binario.

Nelle sequenze a sincronizzazione esterna l'impulso binario porta con sé soltanto un'informazione temporale; ad esempio, alla base 0 può seguire soltanto lo stato 1 ed x determina quando ciò avvenga.

In figura 1.3 è mostrata una sequenza *autosincronizzata*, costituita da due segnali binari impulsivi (x_1, x_2) e da due segnali binari a livello (l_1, l_2), per un totale di 16 stati di ingresso. I segnali binari x_1, x_2 sono impulsivi, e la sequenza di stati lo è altrettanto, in quanto singolarmente godono della proprietà della x del caso a sincronizzazione esterna. Nell'esempio, gli stati 2, 1, 10, 13, 14, 5 sono impulsivi e 0, 0, 8, 12, 12, 4 ne sono ordinatamente le basi: si noti che una medesima base (p.e. 0) è base di distinti stati impulsivi (p.e. 1 e 2).

In una sequenza autosincronizzata gli impulsi binari non portano con sé la sola informazione temporale come nel caso precedente, ma anche di quale degli impulsi si tratti; ad esempio, sulla base 12 insistono sia lo stato impulsivo 13 (associato a x_2) che il 14 (a x_1) e l'avvento di x_1 o x_2 fornisce appunto l'informazione di quale dei due sia avvenuto oltre che di quando.

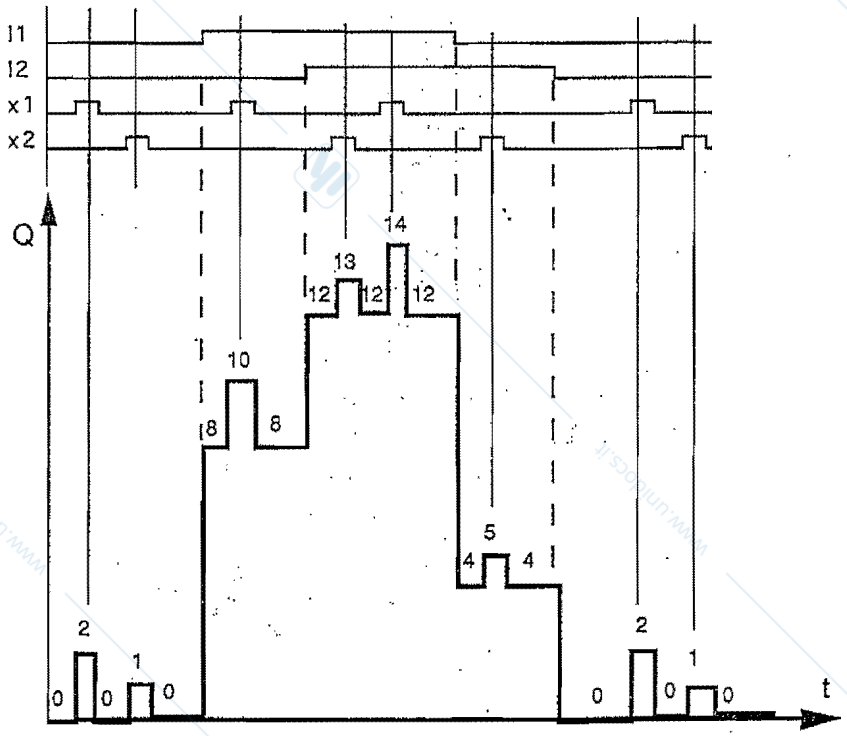


Fig. 1.3: Sequenza sincrona autosincronizzata

2. Modelli di reti sincrone

2.1 Reti sincrone

Una macchina sequenziale è *sincrona* se la sua tabella non soddisfa le proprietà di quella di una macchina asincrona: in termini di definizione, una macchina è sincrona se non è asincrona. Le reti sequenziali che interessano concretamente sono quelle corrispondenti a macchine *sincrone impulsive*, aventi cioè una tabella sincrona e per ingresso una sequenza impulsiva. In figura 2.1 è esemplificata una tabella di stato di una rete sincrona impulsiva (nel seguito detta semplicemente *sincrona*). Essa va letta in modo diverso da quella di una macchina asincrona: la rete riposa in un determinato stato (S_0 nell'esempio), localizzato ai margini della tabella (e non nel suo interno, come nel caso asincrono) e, in presenza di un ingresso impulsivo, si sposta

verso un nuovo stato di riposo (S_2) secondo la transizione definita dalla tabella. Al termine della transizione il sistema permane a riposo nello stato raggiunto. Le transizioni della rete avvengono, quindi, soltanto in corrispondenza degli ingressi impulsivi (in effetti, i "margini" della tabella rappresentano un'altra parte della tabella costituita da stati tutti stabili in corrispondenza delle basi della sequenza di ingresso impulsiva).

In altre parole, la tabella rappresenta soltanto ciò che avviene negli istanti in cui sono presenti impulsi agli ingressi della macchina: la tabella è riferita ad uno spazio discreto del tempo, scandito dagli impulsi in arrivo. La lettura della tabella è, quindi profondamente diversa da quella di una macchina asincrona, che va letta nella continuità del tempo (cfr.§ 1).

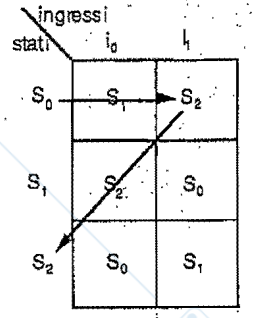


Fig. 2.1: Tabella sincrona

In una rete sincrona, la sincronizzazione non è dunque intrinseca della tabella (come accade per una rete asincrona), ma le deriva dal modello costruttivo adoperato, che memorizza gli stati della rete in appositi flip-flop (anche per le reti asincrone esiste il modello con flip-flop, ma lì questi non hanno funzioni di sincronismo).

A seconda del tipo della sequenza di ingresso, si hanno due distinti modelli costruttivi di reti sincrone:

- reti a sincronizzazione esterna;
- reti autosincronizzate.

2.2 Modello a sincronizzazione esterna

In figura 2.2 è mostrato il modello realizzativo della rete a sincronizzazione esterna: le variabili di stato sono memorizzate in flip-flop qualsiasi (RS, T, JK, D), essi stessi a sincronizzazione esterna e sincronizzati dall'unico impulso binario *x* della sequenza di ingresso; gli altri ingressi sono tutti a livelli

e la rete combinatoria computa i segnali di posizionamento dei flip-flop di stato in funzione soltanto di quest'ultimi.

Le uscite possono essere tanto a livelli (Z) quanto impulsive (Z'), ottenute nel secondo caso con una AND fra un valore a livelli e l'impulso:

$$Z=f(L); \quad Z'=x \cdot f'(L)$$

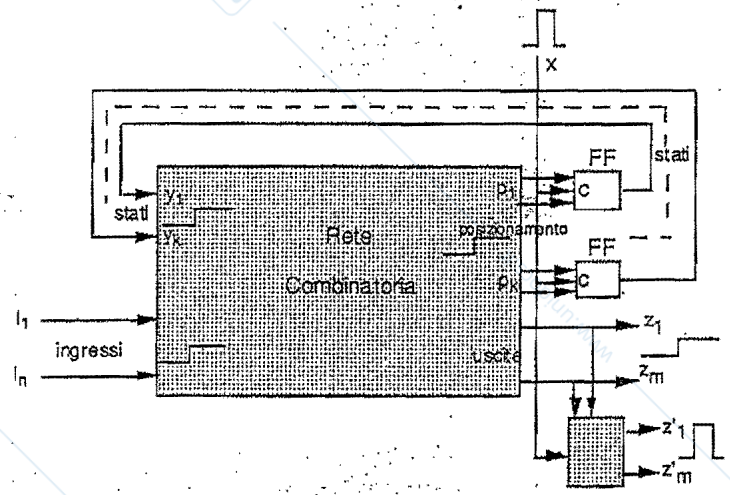


Fig. 2.2: Modello di rete sincrona sincronizzata dall'esterno

Se si adottassero flip-flop latch, la durata di x andrebbe accuratamente dimensionata, ma i problemi di sincronismo sono facilmente risolti con l'adozione di flip-flop edge-triggered o master-slave.

Alla sequenza e alla macchina è stato dato il nome di "sincronizzata dall'esterno" in quanto l'impulso x si sovrappone, appunto dall'esterno, ai segnali a livello che da soli forniscono il contenuto informativo della sequenza e determinano le azioni della rete.

L'impulso ha soltanto funzione di tempificazione (spesso esso è sincrono oppure coincide con un segnale di clock). Esso deve avere un ritardo dall'applicazione del segnale a livelli pari almeno al ritardo della rete combinatoria per garantire la corretta computazione del segnale di posizionamento.

2.3 Modello autosincronizzato

In figura 2.3 è mostrato il modello autosincronizzato, con ingressi misti (livelli ed impulsi), funzioni di posizionamento impulsive della forma

$$x_1 \cdot f_1(L)+x_2 \cdot f_2(L)+\ldots+x_q \cdot f_q(L)$$

ed uscite tanto a livelli che impulsive.

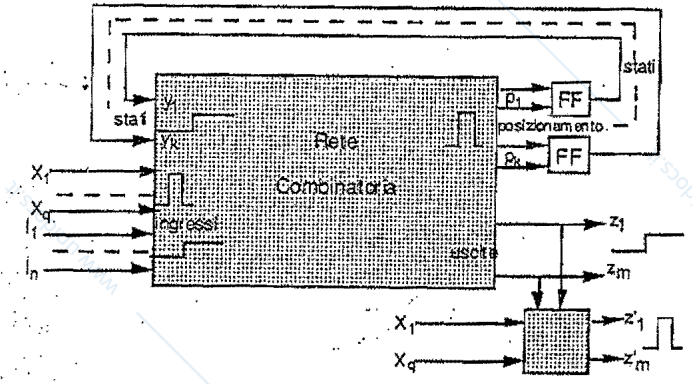


Fig. 2.3: Modello di rete sincrone autosincronizzata

La sequenza di ingresso e la macchina sono dette "autosincronizzate" in quanto gli impulsi di ingresso non hanno soltanto funzione esterna di tempificazione, ma determinano essi stessi il contenuto informativo della sequenza e l'azione che la rete deve compiere: su una determinata base a livelli, a seconda che intervenga l'uno o l'altro impulso, sono diversi i valori assunti dalle funzioni di posizionamento.

La forma assunta dalle funzioni di posizionamento non consente di distinguere, come nelle reti a sincronizzazione esterna, un'unica funzione a livelli che definisce la transizione da effettuare e un unico impulso che ne determina l'istante. Ciò esclude la possibilità di adoperare flip-flop sincronizzati per la memorizzazione degli stati: ne segue che i flip-flop di stato sono o flip-flop RS fondamentali oppure flip-flop T. Il modello, non consentendo l'uso di flip-flop edge-triggered, presenta problemi di tempificazione: il flip-flop RS fondamentale, e a maggior ragione il T latch, posseggono i soliti problemi di dimensionamento della durata dell'impulso (cfr. esempi in §§ 3 e 4). E' viceversa utile l'impiego di un flip-flop T asincrono (cfr.§ III.16) che, avendo un comportamento assimilabile ad un flip-flop edge-triggered, risolve come questo i problemi di tempificazione. All'atto pratico, si usa un flip-flop T abilitato, con $T=I$ (oppure un JK con J=K=1) e *c=funzione di posizionamento* (cfr. esempio §5).

2.4 Trasformazione di sequenze

Le migliori proprietà di sincronizzazione delle reti con un unico impulso binario suggeriscono talora di modificare le caratteristiche temporali della sequenza di ingresso autosincronizzata, trasformandola in sincronizzata dall'esterno. Ciò è realizzato in appositi circuiti a monte della rete.

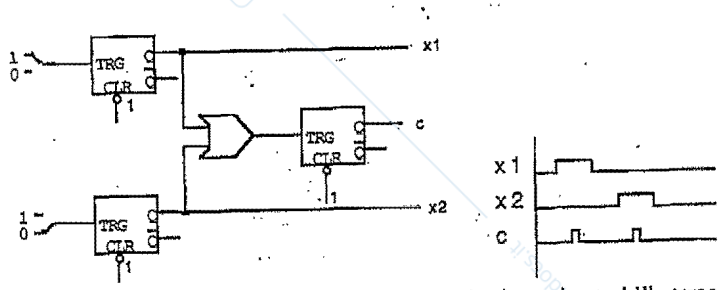


Fig. 2.4: Trasformazione sequenza autosincronizzata in sincronizzata dall'esterno

In figura 2.4 è mostrato un circuito che trasforma una sequenza con due impulsi, x_1 e x_2: sul fronte di salita di ciascuno dei due viene generato un ulteriore impulso *c*, ritardato rispetto al fronte e di durata più breve, in modo che il suo fronte di discesa sia interno all'impulso che lo ha generato. A tale scopo è adoperata una or ed un monostabile che genera l'impulso c dopo il definito ritardo dai fronti di x_1 e x_2. Si noti che in figura i primi due monostabili simulano x_1 e x_2, il terzo fa parte del circuito di trasformazione. La sequenza diventa a sincronizzazione eterna con x_1 e x_2 livelli e c impulso.

Per n>2 impulsi binari, si potrebbe costruire banalmente una OR a n ingressi, ma più in generale gli n impulsi potrebbero essere codificati in $k=\log_2 n$ segnali binari a livelli di una sequenza a sincronizzazione esterna, fermo restando che l'impulso di sincronizzazione è ottenuto dalla OR ritardata degli n impulsi. Inoltre, i k segnali a livelli possono essere mantenuti alti da altrettanti flip-flop, posizionati mediante un'apposita rete che elabora gli n impulsi. In figura 2.5 è riportata una rete di conversione per una sequenza autosincronizzata di 3 impulsi x_1, x_2 e x_3, codificati come segue su due flip-flop l_2, l_1: $x_1 = 00$, $x_2 = 01$, $x_3 = 11$. Con tale codice ed adottando flip-flop dotati di segnali di set e reset asincroni (RS, JK dotati ingresso di clear e set, ecc) per la realizzazione, si ottiene:

$$\text{set-}l_1 = x_2+x_3 \quad \text{reset-}l_1 = x_1 \quad \text{set-}l_2 = x_3 \quad \text{reset-}l_2 = x_1+x_2$$

La rete è stata implementata mediante flip-flop JK dei quali si adoperano i soli ingressi set e reset. Tali ingressi sono 0-attivi per cui sono comandati dai segnali negati del monostabile.

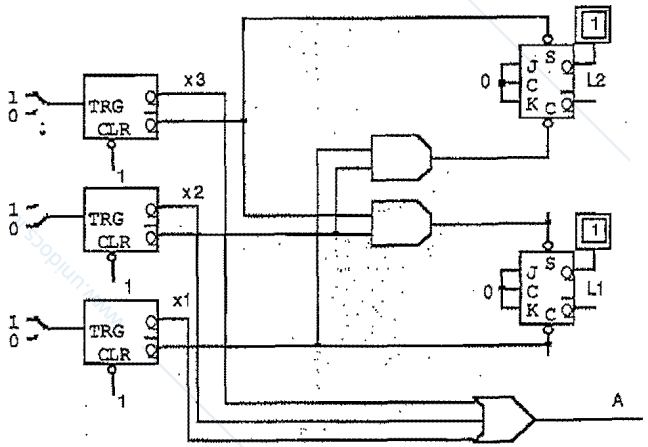


Fig. 2.5a: Convertitore di sequenza autosincronizzata in sincronizzazione esterna

Nella rete ove è applicata la sequenza trasformata, si adopera il fronte di discesa del segnale $A= x_1+x_2+x_3$, presente quando è già avvento il posizionamento corretto dei flip-flop e quindi degli ingressi a livelli l_1 ed l_2. La distanza di tempo fra il posizionamento dei flip-flop (ingressi a livelli) e il fronte di discesa di A (fronte di sincronizzazione) deve essere dimensionata sulle esigenze della rete a sincronizzazione esterna cui la sequenza.

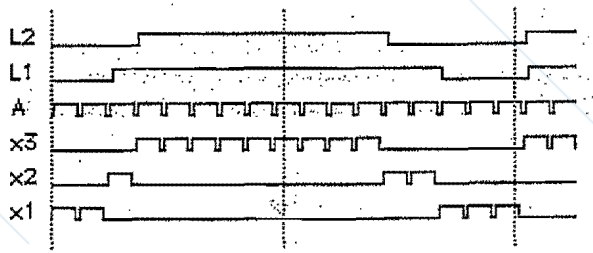


Fig. 2.5b: Convertitore: tempificazione

Altra soluzione potrebbe essere quella di utilizzare lo stesso schema di fig. 2.4 impiegando un trascodificatore per il segnale a livelli.

3. Riconoscitore di evento

Tipo di circuito: rete sequenziale sincrona autosincronizzata
Riferimento: RL, X-esempio 3
Obiettivo: tempificazione per le uscite impulsive

Testo

Costruire una rete che fornisca in uscita un impulso z sincrono con il primo impulso x compreso fra gli impulsi x_a e x_b. Per la risoluzione del problema si considerino e si comparino due differenti realizzazioni: rete autosincronizzata e rete a sincronizzazione esterna.

3.1 Rete autosincronizzata

Tecnica di progetto

Essendo la sequenza di ingresso autosincronizzata, la rete è "naturalmente" tale. Si segue, dunque, la omonima tecnica di progetto.

Tabella di stato ed assegnazione

La tabella di stato della macchina risulta quella di seguito riportata, tipicamente sincrona: per ogni impulso è riportato lo stato seguente, e l'unica uscita impulsiva z (di Mealy) è segnata quando è z=1.

stato	x_a	x_b	x
S_0	S_1	S_0	S_0
S_1	S_1	S_0	$S_0/1$

La macchina ha pertanto due stati, S_0=(0) S_1=(1), che sono codificati mediante un'unica variabile y.

Scelte di progetto e funzioni di posizionamento

Scegliendo come flip-flop di stato un flip-flop RS fondamentale a NOR, si hanno le seguenti funzioni di posizionamento e di uscita:

$$R = x_b + x \qquad S = x_a \qquad z = x \cdot y$$

Descrizione del circuito (fig.3.1)

- Gli impulsi x_a, x_b, x sono realizzati con 3 monostabili ed hanno come ampiezza 10 unità di tempo.
- All'interno delle porte sono segnati i ritardi delle stesse, in convenzionali unità di tempo.
- Per porre in evidenza i problemi di tempificazione, sono state analizzate due implementazioni della rete combinatoria (uscite z e z') con due diversi ritardi delle porte logiche, come evidenziato in figura.

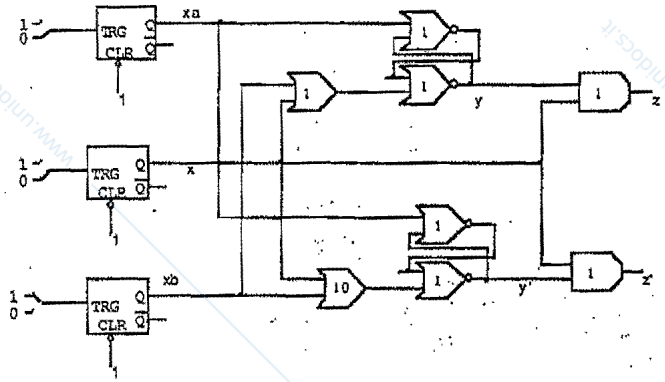


Fig. 3.1a: Riconoscitore di evento

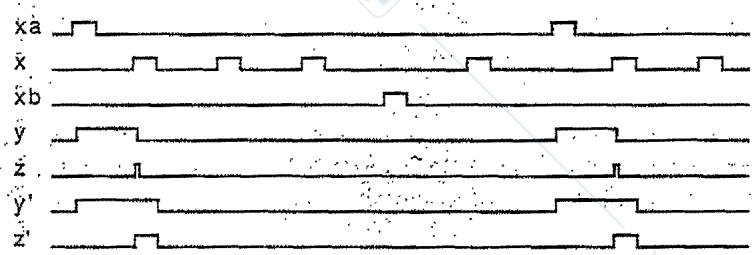


Fig. 3.1b: Riconoscitore di evento: tempificazione

Tempificazione

Allo scopo di descrivere il comportamento temporale del circuito si ponga:
- w l'ampiezza di x (10 nell'esempio);
- R il ritardo di una NOR del flip-flop (1 nell'esempio);
- C il ritardo della rete combinatoria (1 nel primo circuito, 10 nel secondo);

- l_z la durata di z;
- t=0 il tempo corrispondente all'istante in cui, nello stato y=1, si ha la variazione 0→1 di x.

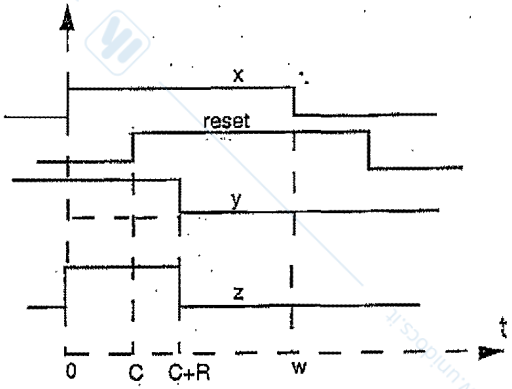


Fig. 3.2: Tempificazione dell'uscita di rete sincrona

Si ha allora (fig.3.2):
- x dura da 0 a w
- al tempo C sale il reset
- al tempo C+R scende y
- l'uscita z=x·y dura il tempo in cui y resta alto entro x e quindi da 0 al minore fra w (durata di x) e C+R (durata di y=1 entro x) per cui si ha:

$$l_z = \min(w, C+R)$$

Senza interventi "ad hoc", C è circa uguale a R (si tratta in entrambi i casi del ritardo di 1 o 2 porte) e si ha dunque $l_z = \min(w, 2R)$. D'altro canto, per il corretto funzionamento del flip-flop, deve essere w>2R e dunque $l_z = 2R$: l'uscita impulsiva è dunque di durata molto limitata, comparabile con i ritardi delle porte del circuito.

L'uscita impulsiva sarebbe invece di ampiezza comparabile con quella degli impulsi di ingresso (w) solo se il circuito combinatorio fosse significativamente lento. Per ovviare ai problemi di tempificazione si può rallentare artificialmente il circuito, apponendo un apposito ritardo tra la porta OR e il flip-flop oppure tra l'uscita del flip-flop e la porta z=x·y.

Si noti che, a causa della banalità del circuito (non è possibile avere 2 transizioni permanendo lo stesso ingresso), non esistono limiti superiori alla durata di w, così come avviene nel caso generale di rete autosincronizzata.

3.2 Rete a sincronizzazione esterna

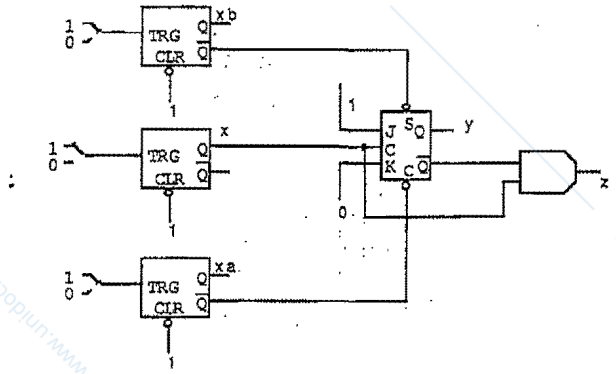


Fig. 3.3a: Riconoscitore di evento a sincronizzazione esterna

I problemi di tempificazione di cui sopra si risolvono radicalmente se si trasforma la sequenza in una sincronizzata dall'esterno e si applica poi il corrispondente modello di rete con l'uso di flip-flop edge-triggered o master-slave. Per la sua semplicità, il circuito costituisce un caso particolare: è ben vero che le specifiche della rete prevedono 3 impulsi distinti, ma i problemi di tempificazione si hanno soltanto in corrispondenza di x, mentre x_a e x_b possono operare anche come segnali a livello. Se allora si adotta un flip-flop JK misto, si possono porre x_a e x_b nel preset e clear e trattare x come impulso esterno. Il segnale x deve settare, invece, il flip-flop e pertanto si ha:

$$\text{preset} = x_a;\quad \text{clear} = x_b;\quad J = 1;\quad K = 0;\quad c = x$$

Ne deriva il circuito banale di figura 3.3a e la corrispondente tempificazione di figura 3.3b.

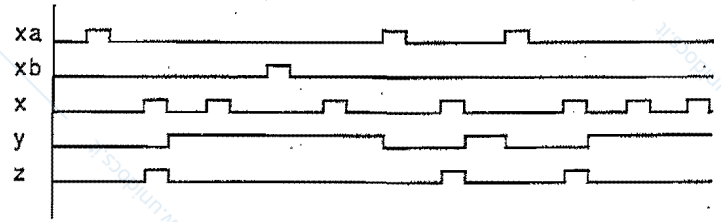


Fig. 3.3b: Riconoscitore di evento: tempificazione

4. Riconoscitore di sequenza con uscita impulsiva

Tipo di circuito: rete sequenziale sincrona autosincronizzata
Riferimento: RL, X-esempio 4
Obiettivo: progettazione sincrona; tempificazione per corretta evoluzione

Testo

Una rete con due ingressi impulsivi, x_1 e x_2, debba fornire un'uscita impulsiva z, sincrona con il secondo impulso x_1 della sequenza x_2-x_1-x_1; in nessun altro caso debba essere presente l'impulso z e i due impulsi di ingresso non possano essere simultanei.

Tecnica di progetto

Progetto di rete sincrona autosincronizzata.

Tabella e assegnazione degli stati

stati (y_1, y_2)	x_1	x_2
q_0	q_0	q_1
q_1	q_2	q_1
q_2	$q_0/1$	q_1

Si ottengono tre stati:

$q_0 = (00)$; $q_1 = (01)$; $q_2 = (11)$

che sono codificati con due variabili, y_1 e y_2 rispettivamente memorizzate in flip-flop RS.

Funzioni di posizionamento

Dalla tabella e dalla codifica adottata si ricava:

$S_1 = \overline{y_1} \cdot y_2 \cdot x_1$ $\quad R_1 = y_1 \cdot x_1 + x_2$
$S_2 = x_2$ $\quad R_2 = y_1 \cdot x_1$
$z = y_1 \cdot x_1$

Descrizione del circuito (fig. 4.1)

Lo stato della rete è posto in evidenza, oltre che con le variabili di stato che si leggono sul disegno della tempificazione, anche attraverso un display esadecimale.

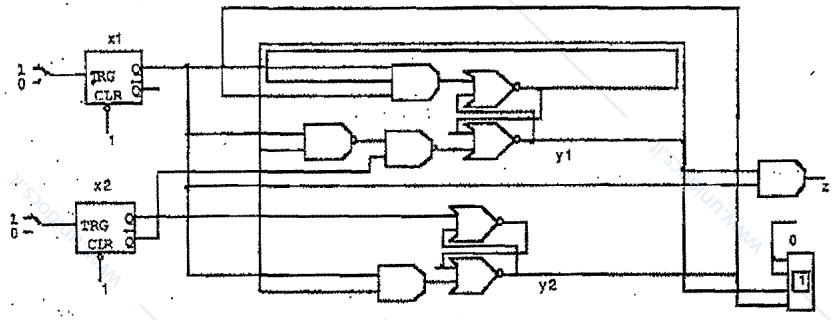


Fig. 4.1: Riconoscitore di sequenza autosincronizzato

Tempificazione

Il ritardo di tutte le porte è di 1 u.t., ad eccezione degli impulsi, sulla cui durata si discuterà in seguito.

I problemi di tempificazione sono quelli già visti al § III-14 (T sincrono con RS latch): per il corretto funzionamento dei flip flop RS componenti, l'ampiezza w dell'impulso deve essere tale da mantenerne la commutazione, cioè maggiore del ritardo complessivo del flip-flop. Per cui, detto R il ritardo di ciascuna delle porte NOR, deve essere: $w > 2R$.

Per il corretto funzionamento della rete sincrona, l'ampiezza w dell'impulso deve essere tale da evitare più transizioni di stato. Detto C il ritardo della rete combinatoria, deve dunque essere $w < 2R+C$. Si ricava pertanto dalle due relazioni:

$$2R < w < 2R + C$$

Nel caso specifico, la doppia transizione potrebbe avvenire sull'impulso x_1 della sequenza x_2-x_1, provocando la transizione $q_1 \rightarrow q_2 \rightarrow q_0$ (per x_1, viceversa, le due transizioni non provocherebbero effetti). La rete combinatoria è a 2 livelli e dunque, considerando che ogni unità ritarda 1 unità di tempo, deve essere $2 < w < 4$.

Nell'esempio, la rete funziona correttamente in quanto è w=3. Si noti che:
- per w=1 e w=2 la rete oscilla (cfr. § III - 4);
- per w=3 la rete funziona correttamente, come evidenziato in figura 4.1b;
- per w=4 oscilla;
- per w > 4 si ha la doppia transizione fra gli stati 01-11-00, come evidenziato in figura 4.2.

Il comportamento del circuito può essere spiegato osservando che, se la durata dell'impulso è maggiore di 3 (per i ritardi della rete), il circuito reagisce nuovamente come se fosse presente un nuovo impulso in ingresso; si ha dunque:
- con w=4 il tempo in cui è presente il segnale all'ingresso non è tale da garantire la seconda commutazione del flip-flop, che pertanto oscilla;
- con w=5 si ha che la durata dell'impulso è tale da garantire una doppia commutazione: $q_1 \rightarrow q_2 \rightarrow q_0$;
- con w>5 si potrebbero teoricamente avere ancora fenomeni di oscillazione ed ulteriori commutazione dello stato, ma allo stato q_0 nel caso specifico segue ancora q_0 e quindi il fenomeno si blocca dopo la doppia transizione.

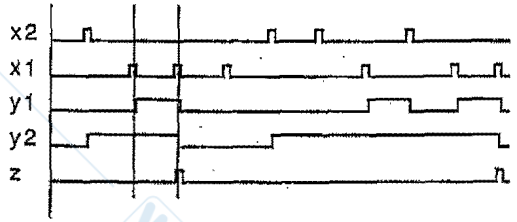


Fig. 4.1b: Riconoscitore di sequenza autosincronizzato con w=3

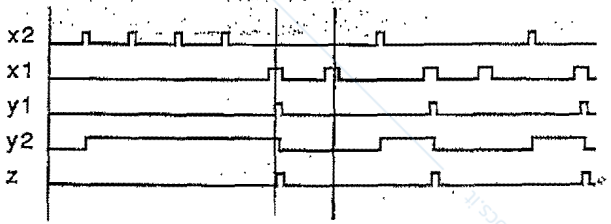


Fig. 4.2: Riconoscitore di sequenza autosincronizzato con w=5

5. Riconoscitore di sequenza con uscita a livelli

Tipo di circuito: rete sequenziale sincrona
Riferimento: RL, X-esempio 5
Obiettivo: trasformazione di sequenza autosincronizzata in sequenza sincronizzata dall'esterno

Testo :

Una rete con due ingressi impulsivi, x_1 e x_2, ed una uscita a livelli, z, si debba comportare come segue:
- se l'uscita è alta (z=1), questa si debba abbassare con il primo impulso x_2 che segue un impulso x_1;
- se l'uscita è bassa, questa si debba alzare con il secondo impulso x_1 della sequenza x_2-x_1-x_1;
- nessun'altra sequenza di impulsi debba alterare lo stato dell'uscita.

Tecnica di progetto

Progetto di rete sincrona autosincronizzata; successiva trasformazione in rete a sincronizzazione esterna.

Tabella ed assegnazione degli stati

stati	x_1	x_2	z
q_0	q_1	q_0	1
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_4	q_3	0
q_4	q_0	q_3	0

Fig. 5.1: Tabella di stato

Si ottengono 5 stati:

q_0= (000), q_1= (001), q_2= (010), q_3=(011), q_4=(100)

che sono codificati mediante 3 variabili di stato y_1, y_2 e y_3.

Scelte di progetto

Per la loro diffusione sul mercato, si scelgono flip flop JK, ma questi sono del tipo a sincronizzazione esterna. Ciò richiede dunque di trasformare gli ingressi da autosincronizzati (come il testo del problema) in sincronizzati dall'esterno (cfr.§ 2.4, fig. 2.4). A tale scopo, sul fronte di salita di ciascuno degli impulsi x_1 e x_2 viene generato un ulteriore impulso *c*, ritardato rispetto al fronte e di durata tale che il suo fronte di discesa sia interno all'impulso che lo ha generato. L'impulso *c* è adoperato come impulso per la sincronizzazione esterna, mentre i due impulsi x_1 e x_2 sono considerati come livelli.

Funzioni di posizionamento

$$J_1=y_2 y_3 x_1 \qquad K_1=x_1+x_2$$
$$J_2=(y_1+y_3)x_2 \qquad K_2=y_3 x_1$$
$$J_3=\overline{y_1}\,\overline{y_2}\,x_1+(y_1+y_2)x_2 \qquad K_3=y_2 x_1+\overline{y_2}\,x_2$$
$$Z=\overline{y_2}\,\overline{y_3}$$

Descrizione del circuito (fig. 5.2)

Si noti che:
- lo stato della rete è evidenziato attraverso un display esadecimale;
- la rete è posta nello stato iniziale 0 mediante il switch *reset*.

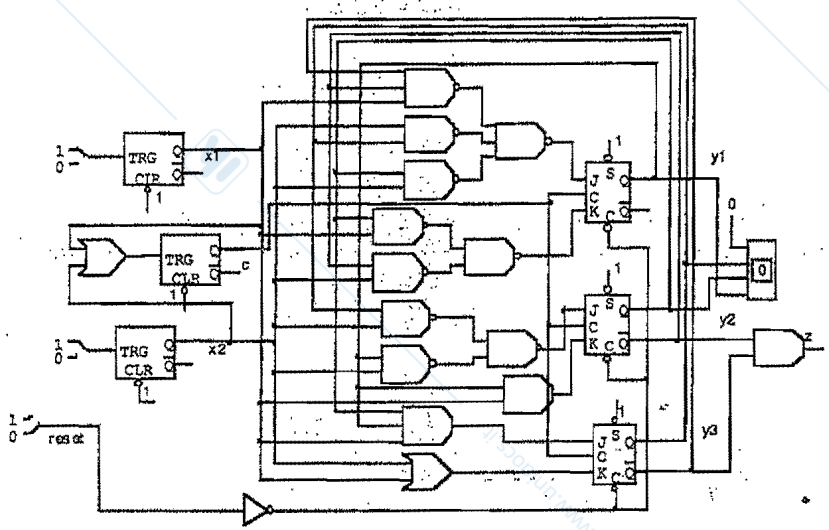


Fig. 5.2a: Riconoscitore di sequenza a sincronizzazione esterna

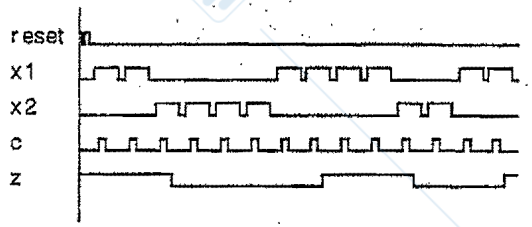


Fig. 5.2b: Riconoscitore di sequenza: tempificazione

Rete autosincronizzata con flip-flop T

I flip-flop T asincroni (sul fronte) sono gli unici che consentono di realizzare una rete autosincronizzata ben tempificata (cfr. § 2.2) e, quindi, senza la necessità di trasformare la sequenza in sincronizzata dall'esterno. Partendo dunque dalla tabella in fig. 5.1 si ottiene:

$$t_1=(\overline{y_1}\,\overline{y_2}\,\overline{y_3}+y_1 y_2\,\overline{y_3})x_1+(y_1\,\overline{y_2}\,\overline{y_3}+\overline{y_1}\,y_2\,\overline{y_3}+\overline{y_1}\,\overline{y_2}\,y_3)x_2$$
$$t_2=y_1 y_2\,\overline{y_3}\,x_1+(\overline{y_1}\,\overline{y_2}\,y_3+y_1\,\overline{y_2}\,\overline{y_3})x_2$$
$$t_3=(y_1 y_2\,\overline{y_3}+\overline{y_1}\,\overline{y_2}\,y_3)x_1+\overline{y_1}\,\overline{y_2}\,y_3 x_2$$

In pratica, si possono adoperare flip-flop JK con J=K=1 e le funzioni t_i applicate all'ingresso impulsivo.

6. Riconoscitore di codice 8421

Tipo di circuito: rete sequenziale sincrona
Riferimento: RL, X-esempio 7
Obiettivo: progettazione sincrona

Testo

Costruire una rete nella quale entrano serialmente i bit di un codice decimale 8-4-2-1 a partire dal bit meno significativo e dalla quale esce un segnale impulsivo che individua se i quattro bit costituiscono o meno una delle 10 parole-codice previste.

Tecnica di progetto

Una sequenza di bit, come l'ingresso di questa rete, è tipicamente sincronizzata da un clock esterno. Si assume, dunque, la tecnica di progetto di una rete a sincronizzazione esterna.

Tabella, grafo ed assegnazione degli stati

stati \ x	1	0
S_0	S_1	S_1
S_1	S_2	S_3
S_2	S_4	S_4
S_3	S_4	S_5
S_4	S_0	$S_0/1$
S_5	$S_0/1$	$S_0/1$

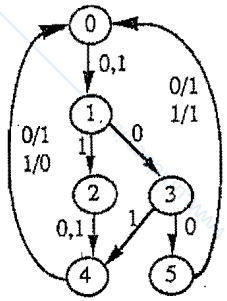


Fig. 6.1: Tabella e grafico di stato

Si ottengono 6 stati, codificati con le tre variabili y_1, y_2 e y_3 nell'ordine:

S_0= (101); S_1= (010); S_2= (111); S_3=(011); S_4=(110); S_5=(000).

Scelte di progetto

Per la memorizzazione delle variabili di stato si utilizzano 3 flip-flop T realizzati con JK.

Funzioni di posizionamento e di uscita

- per il posizionamento dello stato si ricava:

$$t_1=1 \qquad t_2= \overline{y_1}\cdot y_3+y_1\cdot\overline{y_3}\,\overline{x}+y_1\cdot\overline{y_2} \qquad t_3=\overline{y_2}+\overline{y_3}\cdot x$$

- per la componente a livelli dell'uscita:

$$z= y_3\cdot\overline{y_1}\cdot\overline{x}+\overline{y_1}\cdot\overline{y_2}$$

- per l'uscita impulsiva:

$$z' = z\cdot c$$

Descrizione del circuito (fig. 6.2)

Il circuito che implementa la rete è presentato in figura . Si noti che:

- si sono scelti flip-flop JK a fronte di discesa, tipico componente di mercato: in tal modo l'impulso di uscita z dura quanto l'impulso di sincronizzazione x;
- la rete è posta nello stato iniziale S_0=101 mediante un segnale *reset*, che agisce sui *set* e *clear* dei flip flop.

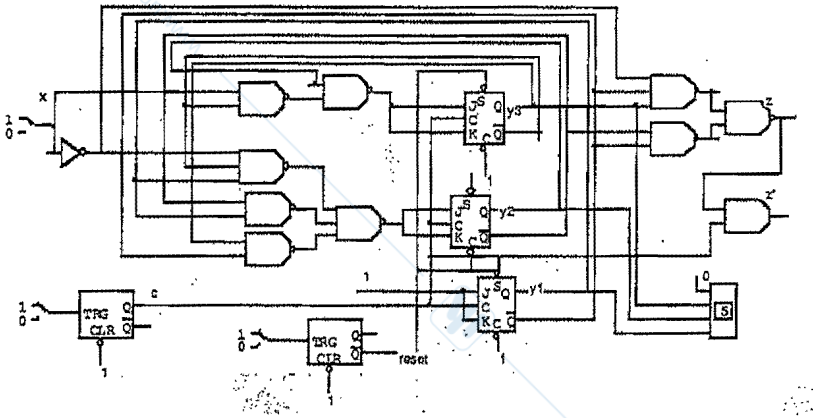


Fig. 6.2: Riconoscitore del codice 8421

Tempificazione degli ingressi

Per simulare una sequenza, il bit di ingresso è simulato dal switch e l'impulso di clock dal monostabile. Questa tempificazione permette di simulare correttamente una possibile segnale di ingresso al dispositivo.

Tempificazione delle uscite

I flip flop sono attivi sul fronte di discesa del clock (secondo fronte). Quindi, l'impulso z', ottenuto con la AND fra stato precedente, ingresso x e clock, ha la medesima durata di quest'ultimo.

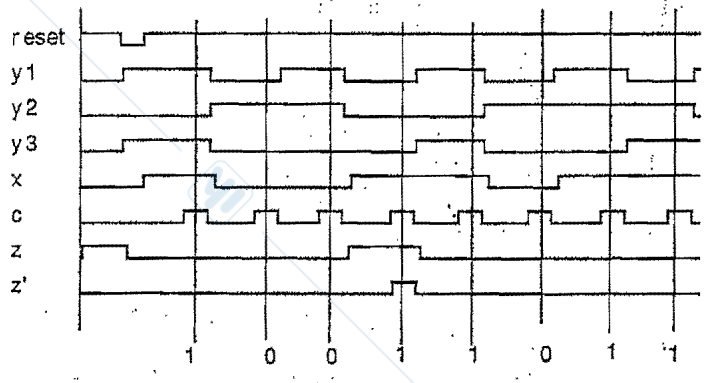


Fig. 6.2b: Riconoscitore del codice 8421: tempificazione

Si noti che l'uscita impulsiva è significativa, mentre la componente a livelli tiene conto di tutte le eventuali oscillazioni di valori del segnale x fra il terzo e il quarto bit. In figura 6.3 è mostrata, ad esempio, la sequenza 111 per i primi tre bit: la rete si porta nello stato S_4 (=110: il visualizzatore segna 6). Se, simulando così un disturbo sulla linea x, si considera una oscillazione da 0 a 1 di x, si ha che l'uscita z oscilla di conseguenza, mentre z' si forma correttamente in sincronismo con il quarto clock, quando x è significativa. In particolare, z' si alza solo se è x=0, cioè per la sequenza 1110 (indicativa del codice della cifra 7) e non se è x=1, codice 1111 (corrispondente ad F in esadecimale).

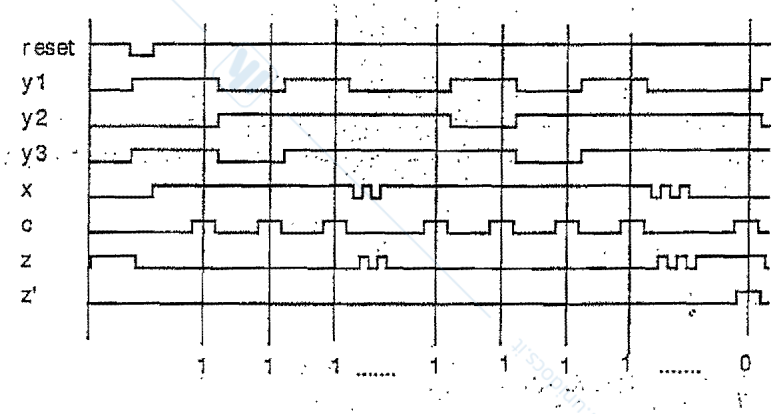


Fig. 6.3: Riconoscitore del codice 8421: analisi dell'uscita

7. Riconoscitore di due sequenze

Tipo di circuito: rete sequenziale sincrona
Obiettivo: minimizzazione degli stati

Testo

Progettare una rete sincrona avente un ingresso X che funzioni da riconoscitore di una delle 2 sequenze: 0110–10, dove con il simbolo " - " si individuano indistintamente i valori 0 o 1.

Specifiche di ingresso-uscita

L'ingresso è quello tipico per una sequenza di bit: un segnale a livelli, X, per il bit ed un clock *cp* che identifica gli istanti in cui X va letto. La rete è quindi a sincronizzazione esterna.

Le uscite debbono essere codificate in modo da distinguere i tre casi di possibile riconoscimento di sequenze: 0110010, 0110110 e tutte le altre. Detti dunque Y, Z due segnali a livelli, si pone:
- Y=0, Z=0: non è stata trovata nessuna sequenza;
- Y=1: riconosciuta la sequenza 0110010;
- Z=1: riconosciuta la sequenza 0110110;
- Y=Z=1: uscita non utilizzata.

Tabella degli stati

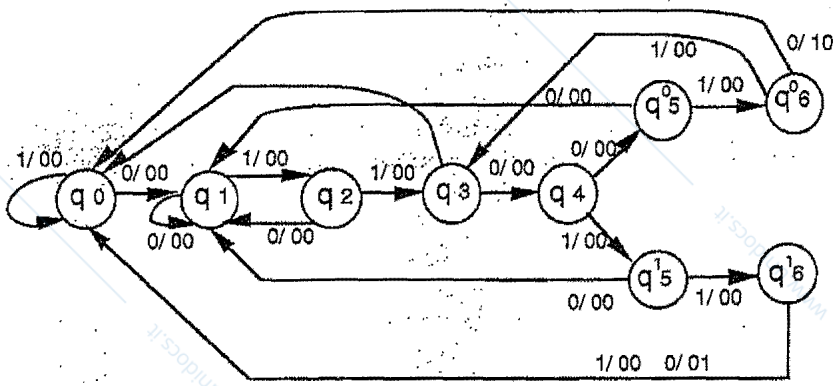


Fig. 7.1: Diagramma di stato

Sulla base delle specifiche del problema, si assumono i seguenti stati:

q_0: stato iniziale;
q_i ($i < 5$): riconosciuto un numero *i* di caratteri della sequenza 0110–10;
q_i^0 ($i \geq 5$): riconosciuto un numero *i* di caratteri della sequenza 0110010;
q_i^1 ($i \geq 5$): riconosciuto un numero *i* di caratteri della sequenza 0110110;

e si ottiene quindi il diagramma di stato di figura 7.1 e la corrispondente tabella di fig. 7.2a.

Minimizzazione degli stati

stati	X = 0	X = 1
q_0	q_1 / 00	q_0 / 00
q_1	q_1 / 00	q_2 / 00
q_2	q_1 / 00	q_3 / 00
q_3	q_4 / 00	q_0 / 00
q_4	q_5^0 / 00	q_5^1 / 00
q_5^0	q_1 / 00	q_6^0 / 00
q_5^1	q_1 / 00	q_6^1 / 00
q_6^0	q_0 / 10	q_3 / 00
q_6^1	q_0 / 01	q_0 / 00

a)

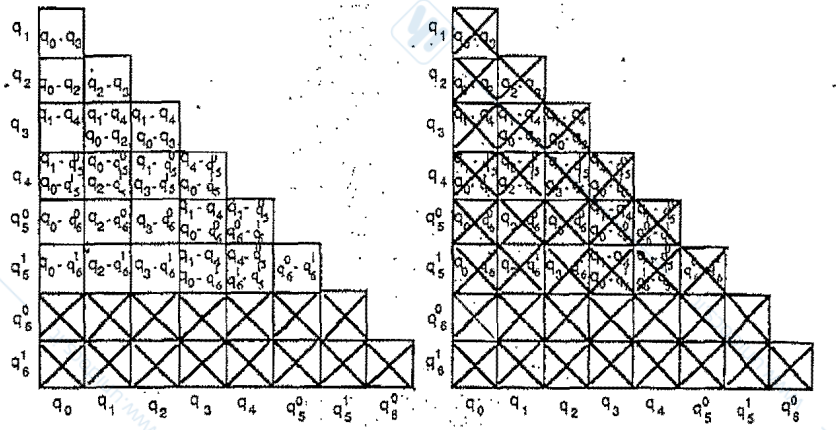
stati	X = 0	X = 1
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_3
q_3	q_4	q_0
q_4	q_5^0	q_5^1
q_5^0	q_1	q_6^0
q_5^1	q_1	q_6^1
q_6^0	q_0	q_3
q_6^1	q_0	q_0

b)

Fig. 7.2: a: Diagramma di stato; b: Partizione

Trattandosi di una macchina completamente specificata, conviene adoperare il metodo tabellare di Paull-Unger per effettuare la minimizzazione.

Considerando le sole uscite, si può effettuare la partizione di cui alla fig. 7.1b), in quanto esiste incompatibilità sulle uscite tra q_6^0 e tutti gli altri stati, tra q_6^1 e tutti gli altri stati e, infine, tra q_6^0 e q_6^1. Dalla tabella risulta evidente che non vi sono righe uguali e, pertanto, si può passare al passo successivo che consiste nella costruzione di una matrice triangolare, nella quale vengono indicate per ciascuna coppia di stati le compatibilità condizionanti. Si ottiene dunque la matrice di figura 7.3a, ove le ultime due righe sono indicate incompatibili con tutte le altre per quanto detto.



a) b)

Fig. 7.3: Minimizzazione con metodo di Paull e Unger

Sulla tabella di fig. 7.3a si barrano via via le caselle contenenti coppie incompatibili e si prosegue iterativamente, ottenendo la tabella di fig. 7.3b. Da essa si evince che non esistono stati compatibili: la tabella di flusso è già minima.

A tale conclusione si poteva pervenire direttamente e più semplicemente considerando che un riconoscitore di sequenze di n bit ha un minimo di n stati e che le due sequenze hanno i primi 4 bit comuni (stati da q_0 a q_4), mentre i due stati per ciascuna sequenza dopo la loro biforcazione sono necessari per distinguere fra loro le due.

Codifica degli stati e funzioni di posizionamento

Per la codifica dei 9 stati della rete occorrono $\{[\log_2 9]\}=4$ variabili binarie; si fissa la tabella di codifica di fig. 7.4a, che porta alla tabella di stato di fig. 7.4b.

Si scelgono flip-flop JK per la memorizzazione delle variabili di stato (di seguito indicate con Q_i) e si ottengono quindi le seguenti funzioni per il posizionamento dei registri di stato:

- per Q_1: $J_1 = X \cdot Q_2 \cdot \overline{Q_3} \cdot \overline{Q_4}$ $\quad K_1 = \overline{X} + Q_3$
- per Q_2: $J_2 = \overline{X} \cdot Q_3 \cdot Q_4$ $\quad K_2 = \overline{X} \cdot Q_4 + Q_3$
- per Q_3: $J_3 = X \cdot Q_4$ $\quad K_3 = \overline{X} + Q_1 + Q_4$
- per Q_4: $J_4 = \overline{X} \cdot \overline{Q_2} + Q_2 \cdot \overline{Q_3} + X \cdot \overline{Q_1} \cdot \overline{Q_3}$ $\quad K_4 = X + Q_3$

e per le uscite:
- per Y: $Y = \overline{X} \cdot \overline{Q_1} \cdot Q_2 \cdot Q_3$
- per Z: $Z = \overline{X} \cdot Q_1 \cdot Q_3$

stato	$Q_1 Q_2 Q_3 Q_4$
q_0	0 0 0 0
q_1	0 0 0 1
q_2	0 0 1 0
q_3	0 0 1 1
q_4	0 1 0 0
q^0_5	0 1 0 1
q^1_5	1 1 0 1
q^0_6	0 1 1 0
q^1_6	1 1 1 0

a)

stati	X = 0	X = 1
0000	0001 / 00	0000 / 00
0001	0001 / 00	0010 / 00
0010	0001 / 00	0011 / 00
0011	0100 / 00	0000 / 00
0100	0101 / 00	1010 / 00
0101	0001 / 00	0110 / 00
1010	0001 / 00	1110 / 00
0110	0000 / 10	0011 / 00
1110	0000 / 01	0000 / 00

b)

Fig. 7.4: a): Codifica degli stati; b): Tabella di stato codificata

Descrizione del circuito (fig. 7.5)

Nel progetto della rete combinatoria della macchina sequenziale si sono utilizzate porte NAND. Nel circuito è presente un segnale di reset per portare la rete nello stato iniziale 0000.

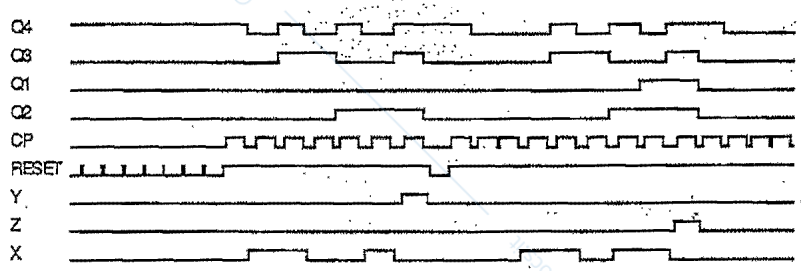


Fig. 7.5b: Rete del riconoscitore: tempificazione

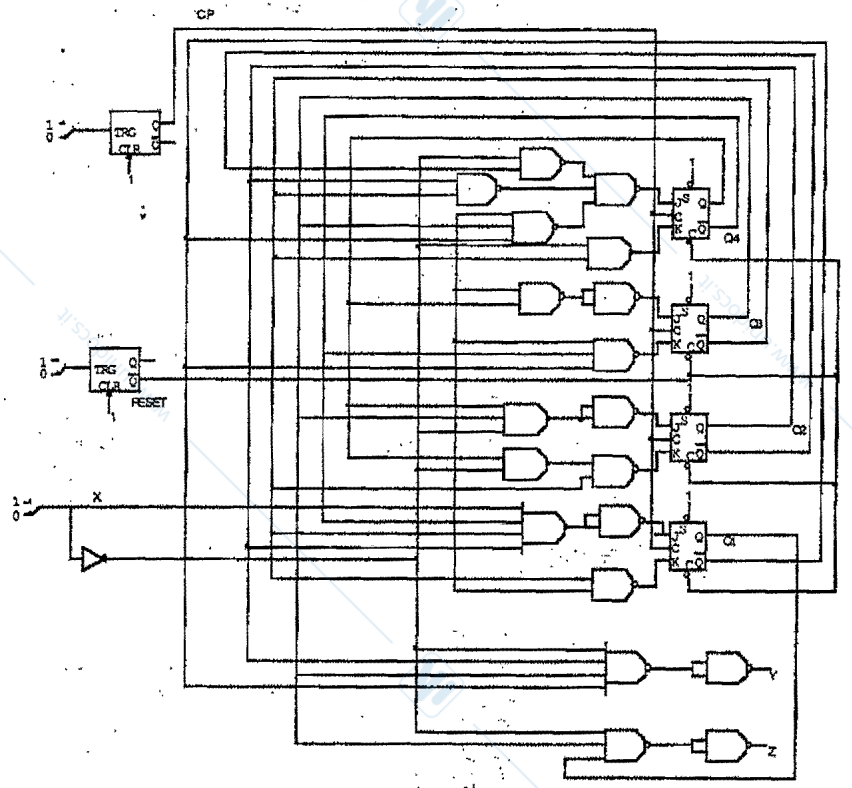


Fig. 7.5a: Rete del riconoscitore

8. I contatori

Un contatore modulo-N è una macchina che conta in modulo N gli impulsi o le variazioni di livello di un segnale assunto come segnale di conteggio. In generale, esso riceve in ingresso i segnali di:
- incremento al conteggio (un impulso o un fronte);
- abilitazione al conteggio (a livelli);
- segnali ausiliari: quali:
  - un segnale per individuare se il conteggio è a crescere o a decrescere;
  - un segnale di reset che azzera il contatore;

- un valore iniziale del contatore e l'associato segnale che ne abilita il caricamento

Inoltre, fornisce in uscita i segnali per:

- Il valore del conteggio (a livelli), tipicamente, ma non necessariamente, un valore codificato con un numero da 0 a N-1.
- Un segnale (*div*) per indicare che il contatore ha un valore di conteggio pari a N-1 per i contatori a crescere (uscita massima) o di 0 per i contatori a decrescere (uscita minima). Il segnale può essere 1-attivo o 0-attivo ed è a livelli: si mantiene attivo per tutto il tempo in cui il conteggio è ai valori di cui sopra.
- Un segnale (*ripple*) sincrono con la transizione del conteggio (N-1)→0 per i contatori a crescere o 0→(N-1) per quelli a decrescere; il segnale, di tipo impulsivo, è utile nella progettazione di contatori di modulo M>N poiché permette di gestire la propagazione del conteggio (come sarà posto in evidenza in seguito). Si noti che l'informazione fornita dal ripple è anche ricavabile dalla variazione del segnale *div*, ad esempio per un div 1-attivo, dal suo fronte 1→0.

Un contatore modulo-N è tipicamente realizzato mediante contatori modulo-k, con k<N, in particolare con contatori modulo-2, per i quali (supposto il contatore a crescere):

- l'uscita di conteggio coincide con l'uscita *div*;
- il segnale *ripple* è associato alla variazione dell'uscita da 1 a 0 e quindi al fronte di discesa dell'uscita.

Si ricorda (cfr.§ III-13), che un contatore modulo-2 è un flip-flop T, che è dunque il componente fondamentale di tutti i contatori; in pratica, si adopera come flip-flop T un flip-flop JK.

Per la realizzazione di un contatore modulo-N attraverso contatori modulo-k (in particolare modulo-2) si faccia idealmente riferimento ad un contatore modulo 10^q progettato a partire da contatori di cifra modulo-10. Ad esempio, un contatore modulo-1000 che conti 0,1,....,9,10,....20, ...,99,...,100,...,999,0 attraverso i contatori delle unità, decine, centinaia. Si supponga, inoltre, che lo stato interno dei contatori coincida con il valore di conteggio. Si può allora far riferimento a due diverse tecniche per esprimere lo stato del contatore, considerando rispettivamente:

- *lo stato complessivo del contatore*: lo stato è funzione dello stato precedente (a 98 segue 99, a 99 segue 100, etc.); allora si ha che le unità variano sempre, le decine solo quando le unità sono a 9, le centinaia solo quando sono a 99, le migliaia a 999, etc.

- *lo stato dei contatori di cifra componenti e la loro variazione*: lo stato di ciascun contatore di cifra varia in funzione della variazione della cifra adiacente di minor peso; allora le unità variano sempre, le decine quando le unità passano da 9 a 0, le centinaia quando le decine passano da 9 a 0 etc.

Quanto detto in particolare per un contatore decimale vale in generale per qualsiasi contatore modulo-N con $N=k^q$, considerando N espresso con q cifre della numerazione di base k e il contatore composto da q "contatori di cifra" (modulo-k).

Dalle considerazioni di cui sopra derivano rispettivamente le due tecniche cosiddette di contatori *sincroni* ed *asincroni* (qui i termini sincrono e asincrono sono usati in una accezione diversa da quella delle macchine sequenziali).

Nei contatori *sincroni* il segnale di conteggio è fornito in parallelo a tutte le unità e l'abilitazione alla commutazione è funzione degli stati precedenti Se i contatori componenti sono flip-flop, lo schema è quello di una rete sequenziale sincrona a sincronizzazione esterna, sincronizzata dall'impulso di conteggio (o da un suo fronte).

Nei contatori *asincroni* o *ripple-counter* o *a propagazione* il segnale di conteggio viene fornito solo alla prima unità, mentre le altre sono poste in serie e ricevono in ingresso ciascuna il ripple dell'unita di peso immediatamente inferiore; il modello è quello di una rete realizzata con collegamento in serie di più reti componenti.

I contatori asincroni, essendo basati sulla propagazione di un segnale, presentano in generale un maggior tempo di risposta e possono portare ad uscite spurie per la differente velocità di commutazione dei singoli contatori componenti. Essi, tuttavia, consentono una notevole semplificazione delle interconnessioni dei moduli nello schema classico di collegamento di n contatori modulo-k per la realizzazione di contatori modulo k^n.

8.1 Contatore binario sincrono

Il contatore sincrono modulo 2^n con flip-flop T come componenti è, per quanto detto, una rete sincrona a sincronizzazione esterna.

Dette Q_0, Q_1, Q_2 le variabili di stato e trasferendo alla numerazione binaria le considerazioni prima fatte per quella decimale, si ha:

$$t_0 = 1 \qquad t_1 = Q_0 \qquad t_2 = Q_1 \cdot Q_0 \qquad t_3 = Q_2 \cdot Q_1 \cdot Q_0$$

• • • • • •

Alle medesime conclusioni si perverrebbe sviluppando il progetto del contatore secondo la metodologia classica.

In figura 8.1 è presentato lo schema fondamentale di un contatore modulo-8, realizzato con flip-flop JK ($J_i = K_i = T_i$) e completato con un segnale RESET che opera sull'ingresso clear dei flip-flop.

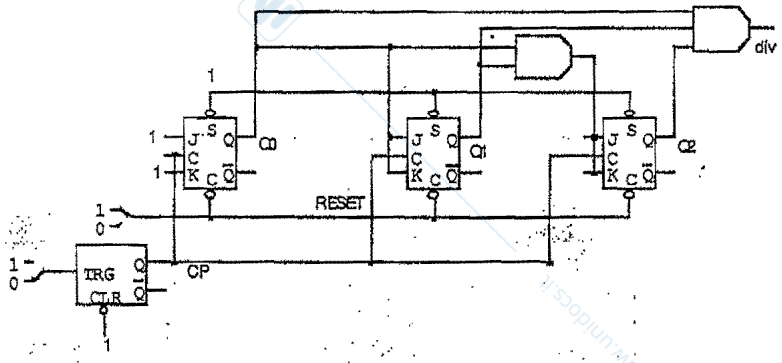


Fig. 8.1a: Contatore modulo 8 sincrono

Uno schema comprendente la possibilità di caricare il contatore ad un valore esterno è mostrato in fig. 8.2: detto LOAD un segnale binario 1-attivo che provoca il precaricamento del contatore ai valori definiti dai bit L_i e adoperando gli ingressi di preset e clear dei flip-flop (0-attivi) si ha:

$$SET_i = \overline{LOAD \cdot L_i} \qquad CLEAR_i = \overline{LOAD \cdot \overline{L_i}}$$

Questo tipo di precaricamento è detto *asincrono* in quanto avviene indipendentemente da un segnale di sincronizzazione.

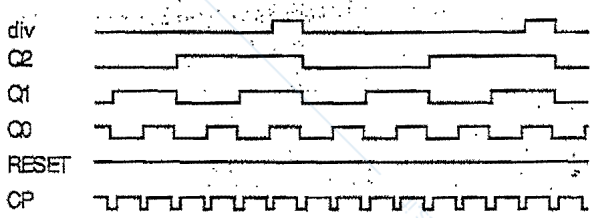


Fig. 8.1b: Contatore modulo 8 sincrono: tempificazione

Un precaricamento *sincrono* può invece essere realizzato adoperando gli ingressi J e K dei flip-flop come abilitazioni e con il caricamento che avviene in sincronismo con l'impulso *cp*:

$$J_i = \overline{load} \cdot t_i + load \cdot L_i \qquad K_i = \overline{load} \cdot t_i + load \cdot \overline{L_i}$$

In tal modo, sugli ingressi clear può essere applicato un reset asincrono:

$$clear_i = RESET$$

Su questo schema si basano i contatori commerciali modulo-16 74161 e 74163, fra loro differenti per una diversa tempificazione del reset: il 161 ha un reset asincrono (agisce sugli ingressi reset dei flip-flop), il 163 un reset sincrono (agisce sugli ingressi K dei flip-flop).

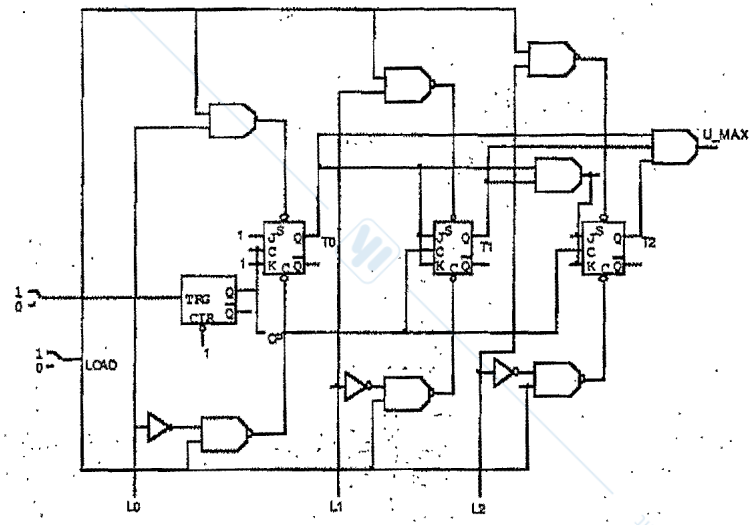


Fig. 8.2: Contatore modulo 8 sincrono con precaricamento

La cella elementare del contatore potrebbe essere realizzata anche con flip-flop di tipo D, come avviene ad esempio nelle versioni 74LS161 e 163 dei contatori commerciali. In generale, la scelta fra le due soluzioni va valutata in termini di complessità della struttura interna dei flip-flop (a vantaggio dei flip-flop D) e dell'architettura complessiva del contatore (a vantaggio dei flip-flop JK); un'analisi di dettaglio porta a concludere che i flip-flop D sono da preferirsi solo per contatori di limitato modulo.

8.2 Contatore binario asincrono (ripple counter)

In fig. 8.3 è mostrato un ripple counter binario modulo-8, ottenuto interconnettendo 3 flip-flop di tipo T (realizzati con JK) edge triggered sul fronte di discesa: il flip-flop i-esimo commuta sul fronte 1→0 del flip-flop (i-1)-esimo, condizione che equivale al segnale *ripple*. Ad esempio (fig. 8.3b), a partire dallo stato 011 (3) si passa allo stato 100 (4) come segue: sul fronte 1→0 del flip-flop di peso 0 si ha la commutazione 1→0 di quello di peso 1 e sul fronte di questo la commutazione 0→1 del flip-flop di peso 2.

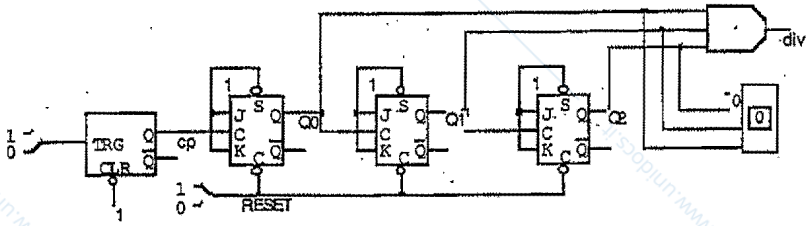


Fig. 8.3a: Ripple counter modulo 8

Per porre in evidenza il funzionamento in cascata del contatore, si sono assunti elevati i tempi di commutazione dei flip-flop del contatore: ciascuno ha un ritardo di 10 u.t.. Si noti come i tempi di commutazione siano dipendenti dal valore del conteggio e il fatto che fra due stati stabili si transiti attraverso stati che determinano uscite improprie: ad esempio la commutazione da 3 a 4 avviene con la sequenza 3→2→0→4.

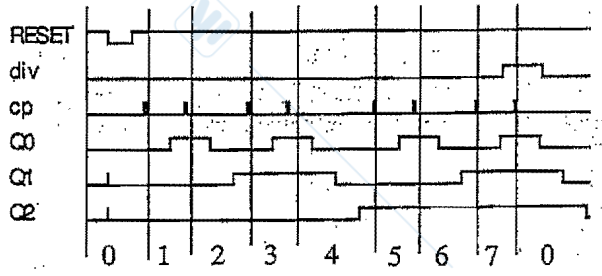


Fig. 8.3b: Ripple counter: tempificazione

9. Contatore bidirezionale

Tipo di circuito: rete sequenziale sincrona
Riferimento: MEI, V-1
Obiettivo: Progettazione sincrona; conoscenza di contatori

Testo

Progettare un contatore modulo 16 dotato di un segnale a livello U di controllo che conti a crescere se è U=1, a decrescere se è U=0.

Progetto

Se il contatore è nello stato di conteggio K e perviene l'impulso di conteggio *cp*, esso transita in uno dei seguenti stati:

- se è U=1, $K = |K+1|_{16}$ (resto modulo-16 di K+1), cioè K=K+1 se è K<15, K=0 se è K=15;
- se è U=0, $K = |K-1|_{16}$, cioè K=K-1 se è K>0, K=15 se è K=0.

Scelta di progetto

Il contatore viene progettato secondo il modello sincrono, utilizzando per la realizzazione 4 flip-flop di tipo JK montati in configurazione T.

Funzioni di posizionamento

Le funzioni per il posizionamento dei flip-flop sono ricavate considerando separatamente le commutazioni necessarie per le due modalità di funzionamento per cui si ha:

- per conteggio a crescere (U):

$$t_0 = 1 \qquad t_1 = Q_0 \qquad t_2 = Q_1 \cdot Q_0 \qquad t_3 = Q_2 \cdot Q_1 \cdot Q_0$$

- per conteggio a decrescere ($\overline{U}$):

$$t_0 = 1 \qquad t_1 = \overline{Q_0} \qquad t_2 = \overline{Q_1} \cdot \overline{Q_0} \qquad t_3 = \overline{Q_2} \cdot \overline{Q_1} \cdot \overline{Q_0}$$

e sovrapponendo gli effetti:

$$t_0 = U + \overline{U} = 1 \qquad t_2 = U \cdot Q_1 \cdot Q_0 + \overline{U} \cdot \overline{Q_1} \cdot \overline{Q_0}$$

$$t_1 = U \cdot Q_0 + \overline{U} \cdot \overline{Q_0} \qquad t_3 = U \cdot Q_2 \cdot Q_1 \cdot Q_0 + \overline{U} \cdot \overline{Q_2} \cdot \overline{Q_1} \cdot \overline{Q_0}$$

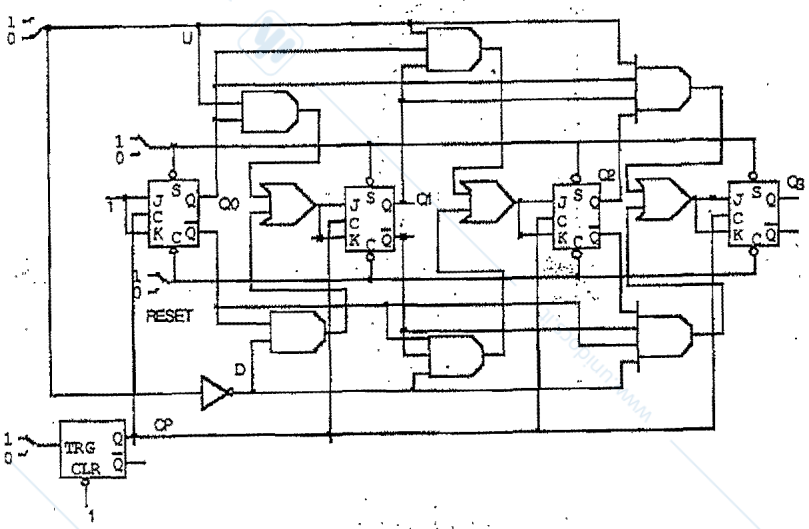


Fig. 9.1a: Contatore modulo 16 bidirezionale

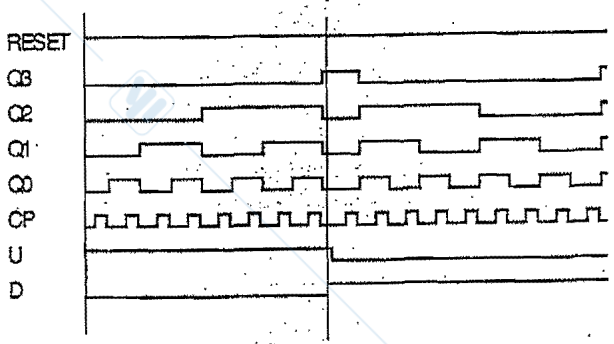


Fig. 9.1b: Contatore modulo 16 bidirezionale: tempificazione

Descrizione del circuito (fig. 9.1)

Il contatore può ritenersi diviso in due sezioni: quella inferiore che lavora come contatore a decrescere, quella superiore che lavora come contatore a crescere. L'abilitazione dei segnali di conteggio, funzione dello stato e del segnale U, è gestita ponendo $J_i = K_i = T_i$ mentre *cp* è inviato in parallelo a tutti i flip-flop nell'ingresso C ed opera sul fronte di discesa. Il segnale $\overline{U}$ è chiamato D per rendere più agevole la lettura del diagramma di tempificazione, ove U è associato ad *up* e D a *down*.

10. Contatore composto

Tipo di circuito: rete sequenziale sincrona
Riferimento: MEI, V-1, circuiti commerciali 74161
Obiettivo: Progettazione sincrona; conoscenza di contatori

Testo

Progettare un contatore modulo 256.

Impostazione del progetto

Il progetto può essere condotto utilizzando due contatori commerciali modulo 16 di tipo 74161. I due contatori sono connessi secondo il modello sincrono presentato nel paragrafo 8.

Progetto

In analogia con la tecnica adoperata per il caso binario, entrambi i contatori ricevono l'impulso di conteggio (ingresso parallelo), ma il contatore di maggior peso è abilitato a commutare solo quando quello di minor peso è al suo conteggio massimo. Si ha, pertanto, che l'uscita U_MAX di quest'ultimo abilita al funzionamento il primo contatore.

La macchina complessiva risultante è un'unica macchina sequenziale a sincronizzazione esterna: l'abilitazione del primo contatore in funzione dello stato del secondo (oltre che di se stesso) è infatti una delle funzioni combinatorie della macchina sequenziale complessiva; basta infatti considerare che le funzioni di posizionamento dei quattro bit più significativi sono:

$$t_4 = Q_3 \cdot Q_2 \cdot Q_1 \cdot Q_0 = U\_MAX$$
$$t_5 = Q_4 \cdot Q_3 \cdot Q_2 \cdot Q_1 \cdot Q_0 = Q_4 \cdot U\_MAX$$
$$t_6 = Q_5 \cdot Q_4 \cdot Q_3 \cdot Q_2 \cdot Q_1 \cdot Q_0 = Q_5 \cdot Q_4 \cdot U\_MAX$$
$$t_7 = Q_6 \cdot Q_5 \cdot Q_4 \cdot Q_3 \cdot Q_2 \cdot Q_1 \cdot Q_0 = Q_6 \cdot Q_5 \cdot Q_4 \cdot U\_MAX$$

I contatori utilizzati commutano sul fronte di salita e quindi anche la macchina risultante possiede tale tempificazione.

Descrizione del circuito (fig. 10.1)

Il contatore 74161 dispone di due ingressi P e T per abilitare rispettivamente il conteggio e l'uscita di conteggio massima. In particolare si ha:

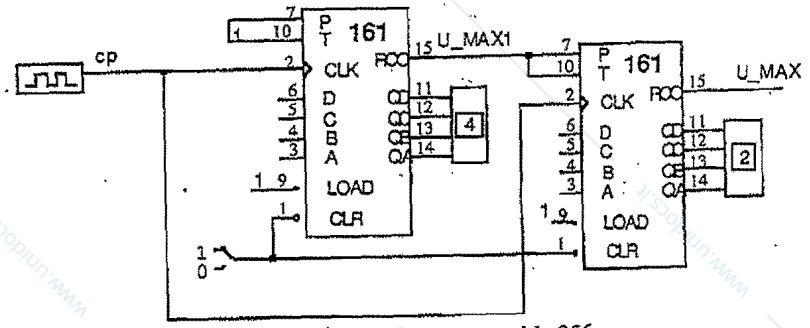


Fig. 10.1a: Contatore modulo 256

$$RCO = T \cdot Q_A Q_B Q_C Q_D$$

Detto dunque U_MAX1 l'uscita div del contatore di minor peso e posto:

P = T = 1 per il contatore di minor peso
P = T = U_MAX1 per il contatore di peso maggiore

si abilita quest'ultimo con U_MAX1 e si ha inoltre che la sua uscita RCO è anche il segnale U_MAX del contatore modulo-256.

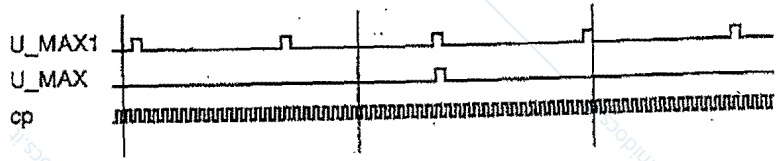


Fig. 10.1b: Contatore modulo 256: tempificazione

11. Contatore modulo 10

Tipo di circuito: rete sequenziale sincrona
Riferimento: MEI, V-1; circuiti commerciali 74163 e 74160
Obiettivo: progettazione di macchine sequenziali

Testo

Progettare un contatore modulo 10.

Impostazione del progetto

Il progetto può essere condotto secondo due differenti approcci: progettando direttamente il contatore o utilizzando per lo sviluppo contatori commerciali modulo 16, opportunamente pilotati attraverso i segnali di reset e precaricamento.

11.1 Contatore indipendente

Le dieci differenti configurazioni che può assumere un contatore a crescere sono:

0000→0001→0010→0011→0100→0101→0110→0111→1000→1001.

Il contatore sincrono (cfr. § 8) richiede 4 flip-flop le cui funzioni di posizionamento dipendono dal tipo di flip-flop e vanno determinate in funzione dello stato precedente del contatore medesimo.

Supposti di tipo JK i flip-flop, si ottengono le mappe di Karnaugh di fig. 11.1, dalle quali si evince:

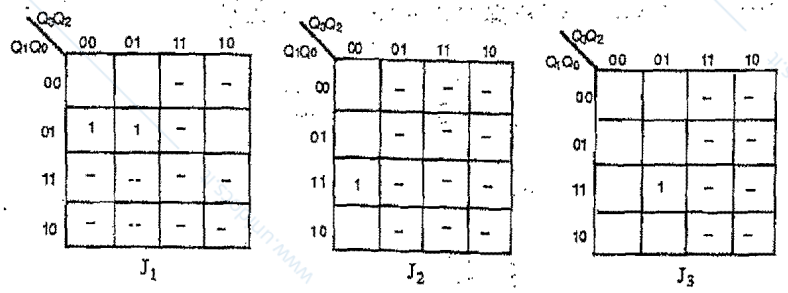


Fig. 11.1a: Mappe per le funzioni di posizionamento

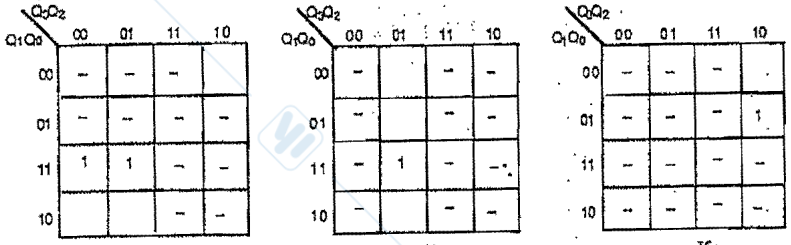


Fig.11.1b: Mappe per le funzioni di posizionamento

$$\begin{aligned} J_0 &= 1 & K_0 &= 1 \\ J_1 &= Q_0 \cdot \overline{Q_3} & K_1 &= Q_0 \\ J_2 &= Q_1 \cdot Q_0 & K_2 &= Q_1 \cdot Q_0 \\ J_3 &= Q_2 \cdot Q_1 \cdot Q_0 & K_3 &= Q_0 \end{aligned} \qquad (1)$$

Considerando i punti di non specificazione si ha poi per l'uscita di conteggio massimo:

$$div = Q_3 \cdot Q_0$$

Si noti che le funzioni di posizionamento per il primo ed il terzo bit coincidono con quelle di tipo T del contatore modulo-16, mentre per il secondo e il quarto si tiene in conto la transizione da 9 a 0:

- il bit Q_1 commuta con Q_0 solo in assenza di $Q_3 (J_1 = Q_0 \overline{Q_3} = Q_0 = K_1)$, mentre resta in reset al conteggio 9 $(J_1 = Q_0 \overline{Q_3} = 0)$;
- il bit Q_3 è settato per $Q_2Q_1Q_0$, cioè nel passaggio da 7 a 8 ed è resettato ogni volta che si abbia il solo Q_0, cioè nei punti 1, 3, 5 e 9.

Una soluzione per la quale tutti i quattro flip-flop operano come T si ha con:

$$J_1 = K_1 = \overline{Q_3} \cdot Q_0 \qquad J_3 = K_3 = Q_3 \cdot \overline{Q_0} + Q_2 \cdot Q_1 \cdot Q_0$$

e quindi con Q_1 che commuta nei punti 1, 3, 5, 7 (e non 9) e con Q_3 che commuta nei punti 9 e 7. Questa soluzione è adottata dal flip-flop commerciale 74160.

E' anche possibile una soluzione con flip-flop componenti di tipo D:

$$\begin{aligned} d_0 &= \overline{Q_0} & d_2 &= Q_2 \cdot \overline{Q_1} + Q_2 \cdot \overline{Q_0} + \overline{Q_2} \cdot Q_1 \cdot Q_0 \\ d_1 &= Q_1 \cdot \overline{Q_0} + \overline{Q_3} \cdot \overline{Q_1} \cdot Q_0 & d_3 &= Q_3 \cdot \overline{Q_0} + Q_2 \cdot Q_1 \cdot Q_0 \end{aligned}$$

La soluzione è più complessa delle precedenti, ma i flip-flop D rendono meno complessi i circuiti per un eventuale precaricamento (load) del contatore; la soluzione è adottata in una versione alternativa del 160 (74LS160).

Descrizione de circuito (fig.11.2)

Il circuito ed il diagramma di tempificazione di figura sono corrispondenti alle (1).

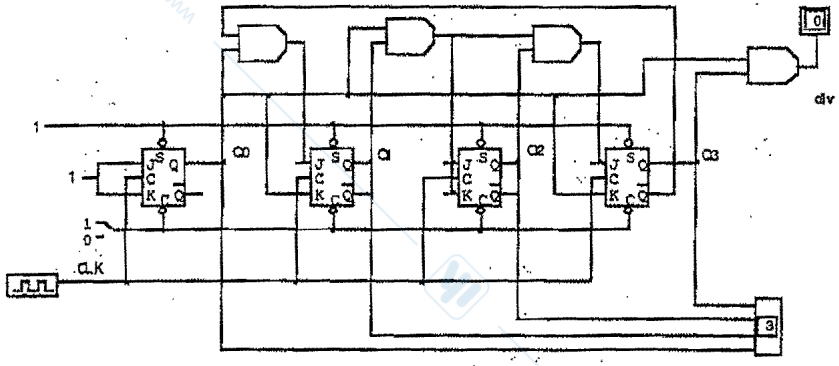


Fig. 11.2a: Contatore modulo 10

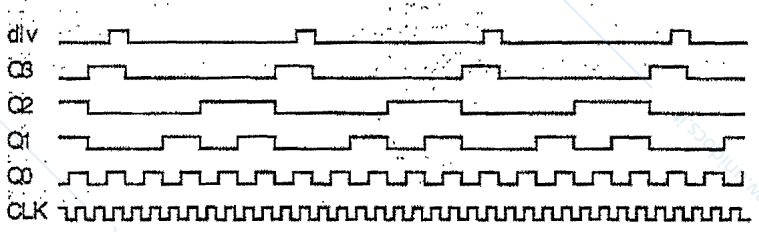


Fig. 11.2b: Contatore modulo 10: tempificazione

11.2 Contatore basato su uno modulo 16

Il contatore modulo 10 può essere progettato a partire dal contatore modulo 16. Tale metodologia di sviluppo di un contatore di modulo M a partire da un contatore di modulo P con P>M è del tutto generale e può essere condotta secondo due differenti modalità:

a) resettando il contatore di modulo P in presenza del segnale di conteggio se si è raggiunto il valore M-1: il conteggio è del tipo 0, 1,...,(M-1), 0, ..(M-1)..;
b) precaricando il valore del contatore a P-M all'atto iniziale ed ogni volta che si avrebbe la commutazione da P-1 a 0: il conteggio è (P-M), (P-M+1),..., (P-1), (P-M),..(P-M+1)...

Nel primo caso le uscite di conteggio dei contatori modulo P e M coincidono, ma non coincidono le uscite di ripple e di conteggio massimo che vanno opportunamente ricalcolate. Nel secondo caso si ha la coincidenza delle uscite di ripple e di conteggio massimo, ma non del valore del conteggio.

Per il metodo a), è necessario resettare il contatore modulo 16, all'arrivo del segnale di conteggio, allorché il contatore è nello stato 9 (1001), corrispondente allo stato di conteggio massimo *(div)* del contatore modulo-10; detto *clr* il segnale di reset e considerando anche la presenza di un reset autonomo (*res*) si ha:

$$div = Q_3 \cdot Q_0 \qquad CLR = div + res \tag{2}$$

Per il metodo con precaricamento (metodo b) deve essere caricato il valore 6 (0110) ogni volta che si avrebbe la commutazione da 15 a 0, quindi in concomitanza con il *ripple* oppure (se li load di ingresso al contatore è a livelli) con il *div*. Detto ancora *res* un segnale di reset esterno, si ha:

$$LOAD = res + ripple \quad \text{se impulsivo} \qquad LOAD = res + div \quad \text{se a livelli} \tag{3}$$

Qualora si voglia un'uscita coincidente con il codice BCD del conteggio è, inoltre, necessaria una rete di trascodifica dell'uscita per associare ad ogni valore in uscita la rappresentazione dello stesso valore diminuita di 6 (p. e. a 6 viene associato 0, a 7 viene associato 1, ecc.). Le equazioni della rete di trascodifica (D, C, B, A) in funzione delle uscite del contatore sono:

$$A = Q_0 \qquad B = \overline{Q_1} \cdot Q_0 + \overline{Q_1} \cdot Q_2$$
$$C = \overline{Q_1} \cdot Q_2 + \overline{Q_2} \cdot Q_1 \qquad D = Q_3 \cdot Q_2 \cdot Q_1 \tag{4}$$

Descrizione del circuito con reset (fig. 11.3)

Si è utilizzato come contatore modulo 16 il componente commerciale 74163, nel quale il segnale di reset (*CLR*) è 0-attivo ed il reset avviene in sincronismo con l'impulso di conteggio. Creando un *div* 0-attivo ed essendo tale il segnale esterno *res*, si ha

$$div = \overline{Q_3 \cdot Q_0} \qquad clr = div \cdot res$$

Essendo *div* 0-attivo l'uscita di ripple coincide con il suo fronte 0→1.

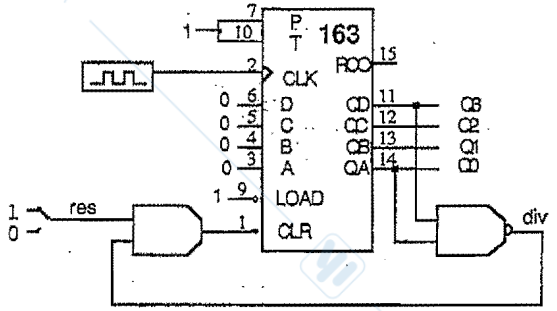


Fig. 11.3a: Contatore modulo 10 realizzato con contatore modulo 16

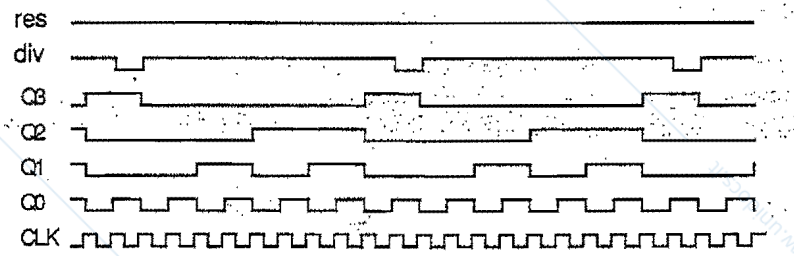


Fig. 11.3b: Contatore modulo 10 realizzato con contatore modulo 16: tempificazione

Descrizione del circuito con load (fig. 11.4)

In figura è riportato il circuito del contatore, sempre basato sul 74163, e la rete di trascodifica associata. Come *div* è disponibile l'uscita RCO (0-attiva); l'ingresso LOAD è a livelli e anch'esso 0-attivo (il caricamento avviene

con l'impulso di conteggio); supposto anche l'ingresso di reset autonomo 0-attivo, dalla (4) si ha:

$$LOAD = \overline{RCO} \cdot res$$

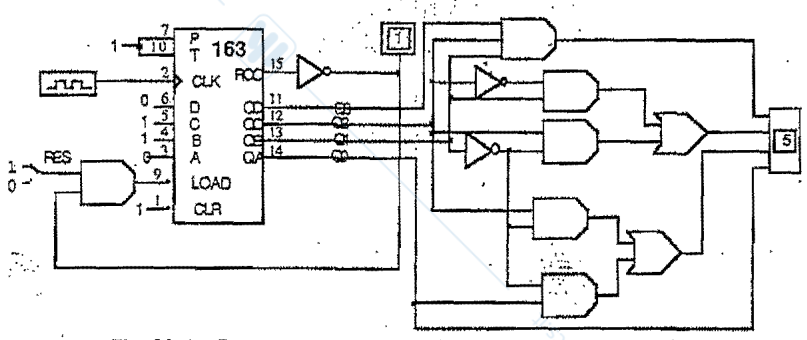


Fig. 11.4a: Contatore modulo 10 realizzato con contatore modulo 16

12. Contatore ad incremento variabile

Tipo di circuito: rete sequenziale sincrona
Obiettivo: progetto sequenziale

Testo

Progettare una macchina M che funzioni da contatore in modulo e che incrementi il valore del conteggio di 1, -1 o +2 in funzione degli ingressi. La macchina disponga inoltre di un segnale di reset che riporti a 0 il valore del conteggio.

Codifica di ingressi e uscite

Nel suo ciclo più lungo (incremento +1 o -1), il contatore è modulo-4 e tale è la macchina base. Per lo sviluppo di dettaglio del progetto occorre codificare gli ingressi e caratterizzarli dal punto di vista della tempificazione. A tale scopo, l'ingresso può essere strutturato con un segnale a livelli (L), che reca l'informazione del valore di incremento, e da un impulso di conteggio (A), che identifica gli istanti in cui viene "letto" il segnale di ingresso e si ha la transizione di stato. La macchina dispone inoltre di un segnale impulsivo di reset (R) che riporta il contatore in uno stato "iniziale". Si ritene R mutuamente esclusivo con A, salvo a rimuovere questa ipotesi in seguito.

Il segnale a livelli L deve essere codificato con 2 bit ($l_1 l_2$), essendo 3 le possibili configurazioni che può assumere; ne deriva, scegliendo i codici arbitrariamente, la seguente codifica: +1→00, +2→01, -1→10.

Il codice ha di fatto il seguente significato:
- l_1 : conteggio doppio (=1) o unitario (=0);
- l_2 : conteggio up (=0) o down (=1) in caso di conteggio unitario (l_1=0).

In definitiva, la macchina risulta essere sollecitata in ingresso da un segnale costituito da tre basi ($l_1 l_2$ = 00, 01, 10) e da due segnali impulsivi (A, R). La sequenza è dunque del tipo "autosincronizzata".

L'uscita della macchina è, invece, a livelli, codificata su due bit per tenere conto dei 4 possibili valori del conteggio (00, 01, 10, 11).

Impostazione del progetto

Essendo la sequenza di ingresso autosincronizzata, tale è la macchina sincrona, che può dunque essere studiata mediante sovrapposizione degli effetti dovuti separatamente ai due impulsi binari A e R, avendo i segnali di posizionamento la forma (cfr.§ 2.3):

$$Af_1(L) + Rf_2(L)$$

Peraltro, l'effetto di R è banale: deve in ogni caso resettare la macchina, indipendentemente dai segnali a livello; la $f_2(L)$ è, pertanto, uguale ad 1 ($f_2(L)$=1). Progetteremo dunque la rete trattando prima l'effetto del solo A, per ricavare la $f_1(L)$.

Tabella di stato

La macchina effettua la transizione fra gli stati in funzione delle variabili di ingresso l_1, l_2 e degli stati, secondo la tabella di fig. 12.1. In sincronismo con R, invece, la macchina viene portata comunque nello stato q_0.

		$l_2 l_1$ 00	01	11	10	uscite $z_2 z_1$
stati	q_0	q_1	q_2	--	q_3	00
	q_1	q_2	q_3	--	q_0	01
	q_2	q_3	q_0	--	q_1	10
	q_3	q_0	q_1	--	q_2	11

Fig. 12.1: Tabella di stato

Per il valore $l_1=l_2=1$, non previsto dal codice adottato, si sono utilizzati punti di non specificazione che potranno essere utili in fase di minimizzazione.

Codifica degli stati

Essendo 4 gli stati, si hanno 2 variabili di stato, che diremo y_2, y_1. La codifica può essere condotta in modo tale che l'uscita risulti direttamente coincidente con il valore dello stato. Si pone dunque: $y_1=z_1$, $y_2=z_2$ e si ottiene la codifica:

stato	variabili y_2y_1	stato	variabili y_2y_1
q_0	00	q_2	10
q_1	01	q_3	11

Scelta dei flip-flop di stato

Per definire completamente il registro di stato occorre effettuare le seguenti scelte:
- tipo degli ingressi dei flip-flop: a memorizzazione (RS, D), a commutazione (T, JK) o misti;
- tempificazione dei flip-flop (latch, edge-triggered, master-slave).

Per quanto attiene al tipo degli ingressi, nel seguito si studieranno e compareranno i seguenti casi alternativi, che saranno quindi commentati:
- flip-flop RS
- flip-flop D
- flip-flop T

La scelta della tempificazione, se i flip-flop sono abilitati, sarà commentata alla voce "tempificazione". Si tratterà il caso del modello autosincronizzato, dettato dalla natura dell'ingresso, ed anche il caso di sincronizzazione esterna, ottenuto con apposita trasformazione della sequenza o con l'uso di appositi flip-flop.

Soluzione n.1: flip-flop RS (modello autosincronizzato)

Dette:
- y_1, y_2 le variabile di stato
- S_1, R_1 il set e reset del flip-flop y_1;
- S_2, R_2 il set e reset del flip-flop y_2;

si ha (fig. 12.2):

$$S_1 = \overline{y_1 l_1} \qquad R_1 = y_1 l_1$$
$$S_2 = \overline{l_2} l_1 y_2 + \overline{y_2} y_1 \overline{l_2} + \overline{y_2 y_1} l_2 \qquad R_2 = y_2 y_1 \overline{l_2} + y_2 \overline{l_1} l_2 + y_2 \overline{y_1} l_2$$

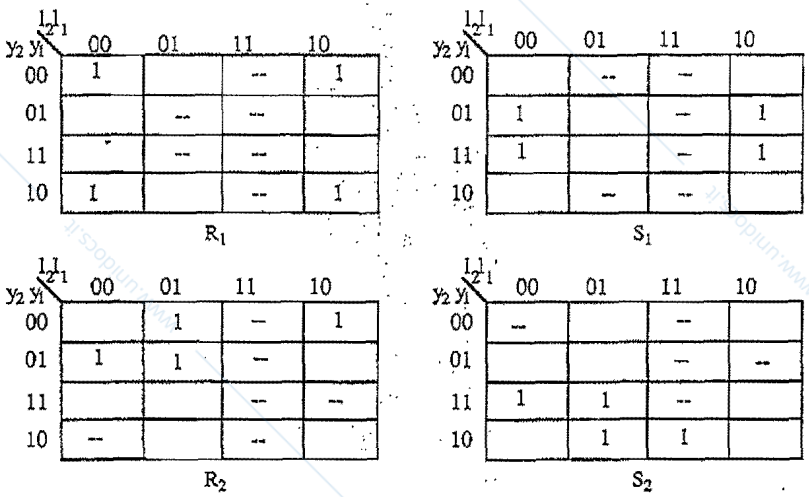


Fig. 12.2: Progetto delle funzioni di posizionamento

Tenendo conto di R e supponendo i flip-flop fondamentali (non abilitati), si ha dunque (soluzione autosincronizzata "pura"):

$$set_1 = A \cdot S_1 \qquad set_2 = A \cdot S_2$$
$$reset_1 = A \cdot R_1 + R \qquad reset_2 = A \cdot R_2 + R$$

Soluzione n.2: trasformazione della sequenza

La soluzione 1 presenta forti problemi di tempificazione. Mantenendo i flip-flop di tipo RS, si può allora trasformare l'ingresso in sincronizzato dall'esterno (cfr. § 2.4), adottando flip-flop sincronizzati; si possono così scegliere flip-flop edge-triggered o master-slave, risolvendo ogni problema di progettazione di durata degli impulsi e di ritardo dei flip-flop.

La trasformazione degli ingressi richiede la costruzione di un unico impulso binario da applicare ai flip-flop sull'ingresso di sincronizzazione. Si può, allora, procedere come segue (cfr. § 2, fig. 2.4): si crea un segnale c=A+R con funzioni di sincronizzazione. Il segnale c, in accordo con la teoria generale delle macchine a sincronizzazione esterna, deve essere ritardato rispetto ad A+R e deve terminare prima di A+R. Se poi il flip-flop è

edge-triggered, è sufficiente che il fronte attivo di c sia interno al segnale A+R. In ogni caso, quindi, questa soluzione richiede appositi circuiti di tempificazione ausiliari: ad esempio, un monostabile che venga eccitato dal primo fronte di A+R e produca un impulso ritardato di d rispetto a questo e di durata t, tale che sia d+t < D.

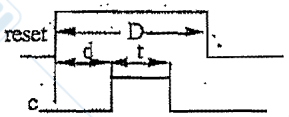


Fig. 12.3: Tempificazione di c rispetto al reset

Soluzione n.3 : flip-flop D (modello a sincronizzazione esterna)

Utilizzando flip-flop D commerciali dotati di segnali aggiuntivi di clear e preset asincroni (si tratta in effetti di flip-flop misti D+RS, cfr.§ 11, circuito 7474), si può utilmente operare come segue:

- il segnale *R* viene applicato direttamente agli ingressi asincroni dei flip-flop;
- il segnale *A* viene applicato agli ingressi di sincronizzazione;
- i segnali l_1 e l_2 sono applicati agli ingressi "dato".

La macchina diventa così a sincronizzazione esterna. in quanto la sequenza che occorre prendere in esame ai fini del progetto dei flip-flop D (segnali di posizionamento "dato") è soltanto quella con l'unico impulso binario A. Dette allora d_1, d_2 le variabili di posizionamento, si ha (fig. 12.4):

$y_2\,y_1$ \ $l_2\,l_1$	00	01	11	10
00	1		–	1
01		1	–	
11		1	–	
10	1		–	1

d_1

$y_2\,y_1$ \ $l_2\,l_1$	00	01	11	10
00		1	–	1
01	1	1	–	
11			–	1
10	1		–	

d_2

Fig. 12.4: Progetto di d_1, d_2

$$d_1 = y_1 \cdot l_1 + \overline{y_1} \cdot \overline{l_1} \qquad d_2 = \overline{y_2} \cdot l_1 + \overline{y_2} \cdot y_1 \cdot \overline{l_2} + y_2 \cdot \overline{y_1} \cdot \overline{l_2} \cdot \overline{l_1} + \overline{y_2} \cdot \overline{y_1} \cdot l_2 + y_2 \cdot y_1 \cdot l_2$$

Se, come accade in genere nei flip-flop commerciali, l'ingresso di *clear* è sempre prioritario rispetto agli altri ingressi, si può anche rimuovere l'ipotesi di mutua esclusione fra *A* ed *R*.

Soluzione n.4 : flip-flop T o JK (modello a sincronizzazione esterna)

Si assume un modello, come nella soluzione 3, a sincronizzazione esterna con l'unico impulso A, e si adottano flip-flop T, Detti quindi t_1, t_2 i segnali di trigger, si ha (fig. 12.5):

$y_2\,y_1$ \ $l_2\,l_1$	00	01	11	10
00	1		–	1
01	1		–	1
11	1		–	1
10	1		–	1

t_1

$y_2\,y_1$ \ $l_2\,l_1$	00	01	11	10
00		1	–	1
01	1	1	–	
11	1	1	–	
10		1	–	1

t_2

Fig. 12.5: Progetto di t_1, t_2

$$t_1 = \overline{l_1} \qquad t_2 = l_1 + y_1 \cdot \overline{l_2} + \overline{y_1} \cdot l_2$$

La realizzazione concreta della rete può avvenire con flip-flop commerciali JK sincronizzati da un ingresso c, che hanno l'effetto dei flip-flop T per J=K=T. Detti flip-flop hanno, similmente ai flip-flop D, ingressi separati di preset e clear asincroni (= non sincronizzati). Si può dunque porre:

$J_1 = K_1 = t_1$ $\qquad$ $J_2 = K_2 = t_2$
$c_1 = A$ $\qquad$ $c_2 = A$
$clear_1 = R$ $\qquad$ $clear_2 = R$

Se l'ingresso di clear è sempre prioritario rispetto agli altri ingressi, anche qui si può rimuovere l'ipotesi di mutua esclusione fra *A* ed *R*.

Si fa presente che le espressioni di t_1 e t_2, ricavate secondo la metodologia classica delle macchine sequenziali, possono anche essere ricavate considerando direttamente le funzioni di posizionamento dei contatori. Si ha infatti che:

- il flip-flop di peso 1 (y_1) commuta sempre che si conti per 1 (l_1=0);
- il flip-flop di peso 2 (y_2) commuta sempre nel conteggio per 2 (l_1=1), oppure nel conteggio per 1:

- se è y_1=1 nel conteggio a crescere ($\overline{I_2}\overline{I_1}$); $y_1\overline{I_2}\overline{I_1}$ si semplifica poi in $y_1\overline{I_2}$, nell'espressione completa, essendo $y_1\overline{I_2}I_1$ incluso in I_1 (condizione di incremento nel conteggio di peso 2);
- se è y_1=0 nel conteggio a decrescere ($I_2\overline{I_1}$); $\overline{y_1}I_2\overline{I_1}$ si semplifica poi nell'espressione completa in $\overline{y_1}I_2$,essendo $y_1I_2I_1$ incluso in I_1.

Confronto fra le soluzioni

Nel confronto fra le soluzioni, risulta nettamente più conveniente quella con flip-flop JK, sia per la ridotta complessità della rete combinatoria, sia per la disponibilità dei segnali di *clear*. Ciò era d'altronde prevedibile, in quanto la macchina è un contatore, anche se a conteggio variabile, ed è noto che i contatori si realizzano utilmente mediante contatori elementari modulo-2, cioè mediante flip-flop T.

Fra le altre soluzioni, è senz'altro da scartare quella con RS fondamentali (soluzione "autosincronizzata pura"), non tanto per la maggiore complessità combinatoria (2+5 porte per il D, 1+1+3+3 porte per lo RS), quanto per le complicazioni derivanti dalla tempificazione. In realtà, il costo teorico della soluzione JK è più elevato (il flip-flop misto commerciale ha una notevole complessità interna), ma quello di mercato è viceversa conveniente.

Descrizione del circuito (fig. 12.6)

Il circuito, corrispondente alla soluzione 4, comprende due monostabili che simulano gli impulsi *A*, *R*. Lo stato è posto in evidenza attraverso un display.

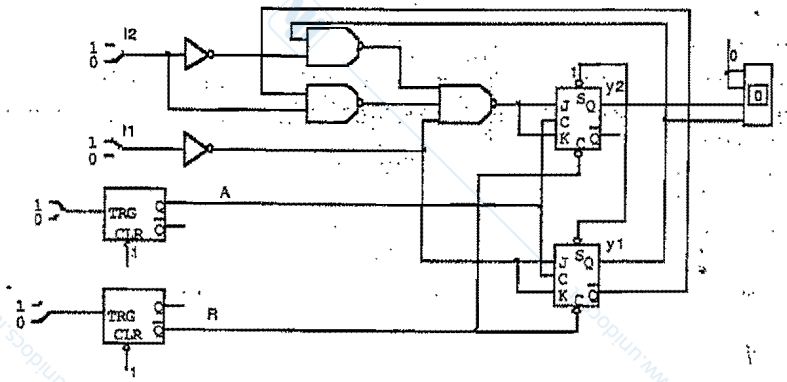


Fig. 12.6a: Schema circuitale realizzazione con flip-flop JK

Tempificazione

Assumendo ritardi unitari per le porte logiche, la rete combinatoria di posizionamento dei flip-flop ritarda di 2 unità di tempo, per cui è necessario che l'impulso *A* arrivi con un ritardo di almeno 2 u.t. rispetto al segnale a livelli L. In realtà, i segnali a livello variano in tempi ben distanti da quelli nei quali insorgono gli impulsi, tipicamente nella mezzeria fra due impulsi.

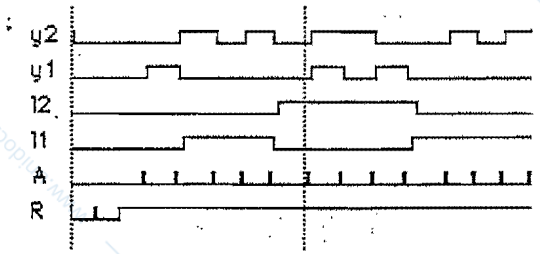


Fig. 12.6b: Schema circuitale realizzazione con flip-flop JK: tempificazione

La tempificazione della macchina suggerisce l'adozione di flip-flop edge triggered o master-slave. Nel caso esemplificato sono stati adottati flip-flop edge sul fronte di discesa . Più in generale, occorre considerare che:

a) se i flip-flop sono master-slave, il segnale di posizionamento dei registri deve essere stabile sul primo fronte di *A* ed il nuovo stato si ripresenterà in uscita sul secondo;
b) se i flip-flop sono edge triggered, il segnale di posizionamento dei registri deve essere stabile sul fronte attivo di *A* ed il nuovo stato si ripresenterà in uscita dopo il ritardo proprio dei flip-flop.

13. Arbitro per la gestione di risorse

Tipo di circuito: rete sequenziale a sincronizzazione esterna
Obiettivo: progettazione sequenziale

Testo

Progettare una macchina M che consenta di gestire l'assegnazione di una risorsa in mutua esclusione tra due unità richiedenti (A e B). Tale macchina, che viene definita arbitro, implementi un algoritmo di assegnazione della risorsa cosiddetto di "round-robin", secondo il quale, in caso di conflitto nella richiesta della risorsa, questa è assegnata con un criterio di priorità che

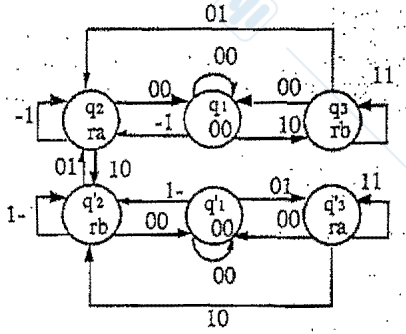
varia in maniera circolare (round-robin). Ad esempio, se in un passo viene assegnata prima ad A e poi a B, nel successivo passo verrà assegnata prima a B e poi ad A. Al rilascio della risorsa la priorità deve variare solo se si vi era una richiesta contemporanea delle unità prima dell'assegnazione della risorsa o se si è verificata una richiesta successiva da parte dell'unità non prioritaria mentre quella prioritaria occupava la risorsa. Si faccia inoltre l'ipotesi che l'unità non perda il diritto di priorità in mancanza di conflitto e che il segnale di richiesta da parte di A (*a*) e di B (*b*) siano alti fino al rilascio della risorsa.

Analisi e completamento delle specifiche

La macchina M può essere sviluppata come una macchina a sincronizzazione esterna: *a* e *b* sono a livelli e un segnale di sincronizzazione identifica gli istanti in cui considerare valida la richiesta. La sequenza è pertanto costituita da un segnale a livelli codificato su due variabili e da un segnale di sincronizzazione ricavato da un clock, che periodicamente permette di considerare gli ingressi per verificare l'avvenuta ricezione di una richiesta o il rilascio di una risorsa. Si definiscono contemporanee (e quindi conflittuali) due richieste attive in uno stesso intervallo di clock.

Esistono, inoltre, due appositi impulsi, *ca* e *cb*, che permettono di configurare la rete negli stati iniziali per i quali se vi è un conflitto la risorsa è assegnata rispettivamente ad A o a B; usando dei flip-flop dotati di clear e set asincroni, tali segnali saranno trattati a parte. La rete ha in uscita due variabili di Moore, *ra* e *rb*, che rappresentano il segnale di abilitazione all'uso della risorsa da parte delle unità A e B rispettivamente e che hanno il vincolo ra·rb=0 (non usano simultaneamente la risorsa).

Tabella e grafo di stato



stati	ba 00	01	11	10	uscita rb, ra
q1	q1	q2	q2	q3	00
q1'	q1'	q3'	q2'	q2'	00
q2	q1	q2	q2	q2'	01
q2'	q1'	q2	q2'	q2'	10
q3	q1	q2	q3	q3	10
q3'	q1'	q3'	q3'	q2'	11

Fig. 13.1: Tabella di transizione degli stati

La rete evolve attraverso i seguenti stati:

q_1: risorsa non assegnata (*ra*=0, *rb*=0) in caso di conflitto assegnazione a A;

q_1': risorsa non assegnata (*ra*=0, *rb*=0) in caso di conflitto assegnazione a B;

q_2: risorsa assegnata ad A (*ra*=1), solo se si è in presenza di conflitto (b=1) prima o dopo l'assegnazione della risorsa la priorità varia da A a B;

q_2': risorsa assegnata a B (*rb*=1), solo se si è in presenza di conflitto (a=1) prima o dopo l'assegnazione della risorsa la priorità varia da B a A;

q_3: risorsa assegnata a B (*rb*=1) in assenza di conflitto, la priorità non varia al rilascio della risorsa (rimane prioritario A);

q_3': risorsa assegnata ad A (*ra*=1) in assenza di conflitto, la priorità non varia al rilascio della risorsa (rimane prioritario B).

Si ottengono così il grafo e la tabella di stato (di Moore) di fig. 13.1. Codificando gli stati con i bit Q_0, Q_1 e Q_2 nell'ordine e ponendo:

q_1=(000), q_1'=(001), q_2=(010), q_2'=(011), q_3=(100), q_3'=(101)

si ottiene la tabella codificata di fig. 13.2.

$Q_0 Q_1 Q_2$	ba 00	01	11	10	uscita rb, ra
000	000	010	010	100	00
001	001	101	011	011	00
010	000	010	010	011	01
011	001	010	011	011	10
100	000	010	100	100	10
101	001	101	101	011	11

Fig. 13.2: Tabella di transizione degli stati con codifica

Progetto combinatorio

Avendo scelto i flip flop D per la memorizzazione delle variabili di stato, dalla tabella di fig. 13.3 si ottengono i segnali di posizionamento d_0, d_1, d_2, in funzione degli ingressi *a* e *b* e di Q_0, Q_1 e Q_2.

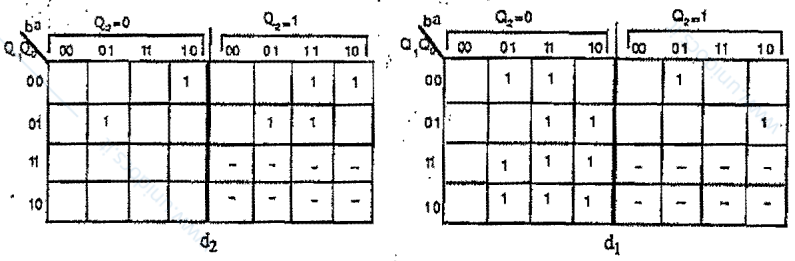


Fig. 13.3 a: Progetto di d_2, d_1

Q_1Q_0 \ ba	$Q_2=0$ 00	01	11	10	$Q_2=1$ 00	01	11	10
00					-	-	-	-
01	1	1	1	1	1	1	1	1
11	1		1	1	-	-	-	-
10				1	-	-	-	-

Fig. 13.3 b: Progetto di d_0

Si ottiene dunque:

$$d_2 = b \cdot a \cdot Q_2 + \bar{b} \cdot a \cdot \overline{Q_1} \cdot Q_0 + b \cdot \bar{a} \cdot \overline{Q_1} \cdot \overline{Q_0}$$
$$d_1 = b \cdot Q_1 + a \cdot Q_1 + b \cdot \bar{a} \cdot Q_0 + \bar{b} \cdot a \cdot \overline{Q_0} + b \cdot a \cdot \overline{Q_2}$$
$$d_0 = b \cdot \bar{a} \cdot Q_1 + \bar{a} \cdot Q_0 + \overline{Q_1} \cdot Q_0 + b \cdot Q_0$$

Le uscite ra e rb, essendo la macchina di Moore, sono funzioni delle sole variabili di stato Q_0, Q_1, Q_2:

$$ra = Q_2 \cdot \overline{Q_1} \cdot Q_0 + \overline{Q_2} \cdot Q_1 \cdot \overline{Q_0}$$
$$rb = \overline{Q_2} \cdot Q_1 \cdot Q_0 + Q_2 \cdot \overline{Q_1} \cdot \overline{Q_0}$$

Occorre ancora definire i segnali reset e clear dei flip-flop in funzione degli impulsi di inizializzazione *ca* e *cb*. Considerando che per il segnale *ca* il sistema si porta nello stato 000 (q_1) mentre per *cb* si porta in 001 (q_1'), si ha che i flip-flop Q_1 e Q_2 vanno posti a 0; mentre il valore di Q_0 dipende dal segnale *ca* e *cb*:

reset-Q_2= ca + cb	set-Q_2= 0
reset-Q_1= ca + cb	set-Q_1= 0
reset-Q_0= ca	set-Q_0= cb

In figura 13.4 è riportato lo schema del circuito con il relativo diagramma di tempificazione. Nel circuito i segnali ca e cb sono 0-attivi per cui le espressioni dei segnali di set e reset dei flip-flop sono state negate.

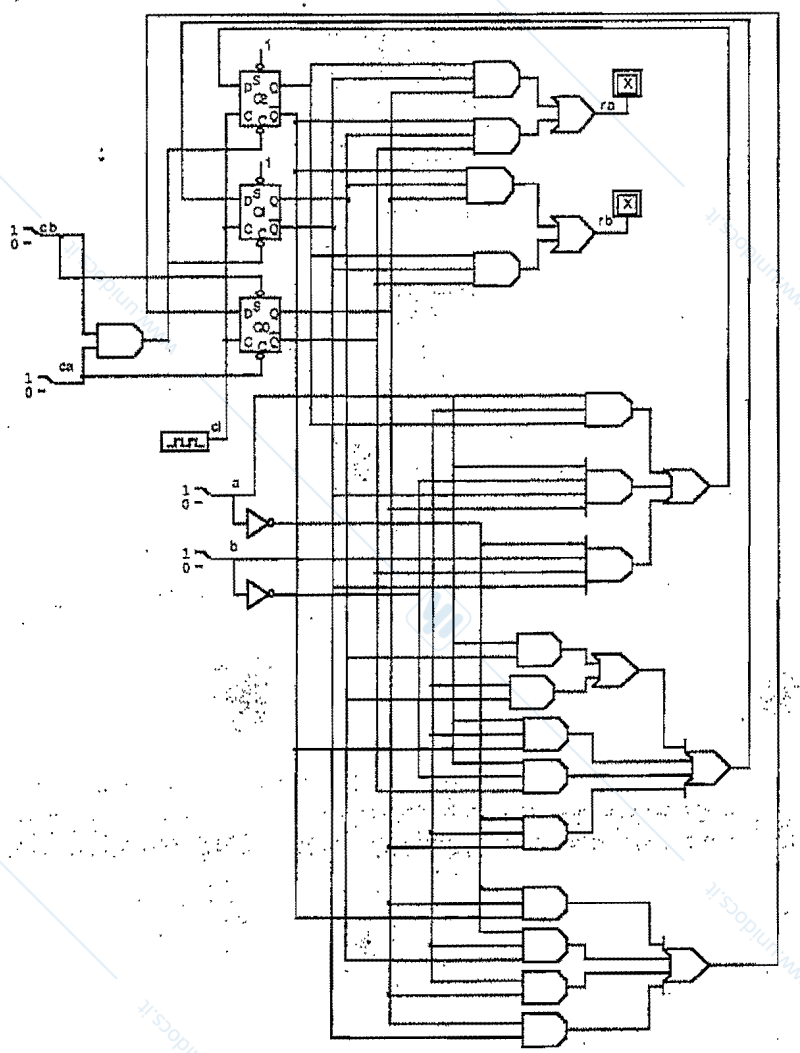


Fig. 13.4a: Rete sequenziale per la gestione della politica di arbitraggio

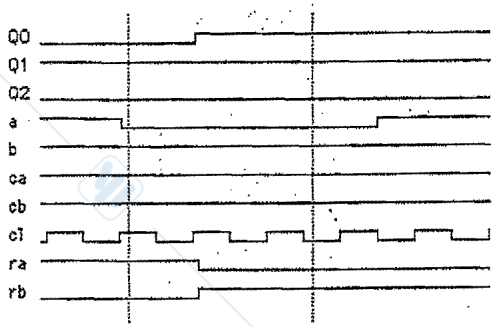


Fig. 13.4b: Rete sequenziale per la gestione della politica di arbitraggio: tempificazione

14. Registri a scorrimento

Un registro a scorrimento binario è un macchina sequenziale a sincronizzazione esterna composta di n flip-flop (Q_i) che, in presenza di un segnale di abilitazione (cp), acquisisce serialmente un dato (D) nella prima posizione di destra (sinistra) del registro, detta Q_0 (Q_{n-1}) ed esegue uno scorrimento verso sinistra (destra) dei dati di una posizione:
- registro a scorrimento verso sinistra (left), detto *shl(Q,D)*:

$$Q_i \rightarrow Q_{i+1} \quad \forall i:\ 0 \le i \le n-2;\ D \rightarrow Q_0 \text{ (si perde il bit } Q_{n-1});$$

- registro a scorrimento verso destra (right), detto *shr(Q,D)*:

$$Q_i \rightarrow Q_{i-1} \quad \forall i:\ 1 \le i \le n-1;\ D \rightarrow Q_{n-1} \text{ (si perde il bit } Q_0).$$

Il registro è "circolare" se il dato immesso ad un estremo del registro è il bit contenuto all'altro estremo, salvando così il bit che andrebbe altrimenti perso:
- registro circolare verso sinistra (left), detto *cil(Q)*:

$$Q_i \rightarrow Q_{i+1} \quad \forall i:\ 0 \le i \le n-2;\ Q_{n-1} \rightarrow Q_0$$

- registro circolare verso destra (right): detto *cir(Q)*:

$$Q_i \rightarrow Q_{i-1} \quad \forall i:\ 1 \le i \le n-1;\ Q_0 \rightarrow Q_{n-1}$$

Sul piano aritmetico, un registro a scorrimento implementa il prodotto per 2 (se si ha uno scorrimento a destra con immissione di un dato pari a 0) o il quoziente intero (se si ha uno scorrimento a sinistra con immissione di un dato pari a 0). Questa proprietà è estesa anche per la rappresentazione in complementi dei numeri: si parla in questo caso di "shift aritmetico" a sinistra -*ashl(Q)*- o a destra -*ashr(Q)*-; in particolare si dimostra (cfr.MEI, § IV, 7, 8):

- per i complementi alla base (complementi a 2):
  $ashl(Q) = shl(Q,0)$
  $ashr(Q) = shr(Q,Q_{n-1})$ (viene ovviamente mantenuto il bit-segno)
- per i complementi diminuiti (complementi ad 1):
  $ashl(Q) = cil(Q)$
  $ashr(Q) = shr(Q,Q_{n-1})$ (realizza il quoziente aritmetico approssimato, non quello troncato)

Il registro a scorrimento è una macchina a sincronizzazione esterna: il banco di flip-flop $Q_0,\ldots,Q_{n-1}$ è il registro di stato della macchina e le espressioni che definiscono lo shift suggeriscono le funzioni di posizionamento dei flip-flop; il segnale di posizionamento di un flip-flop è soltanto funzione di quello immediatamente adiacente. E' ovviamente preferibile che i flip-flop siano edge-triggered o master-slave per i noti problemi di tempificazione delle macchine sincrone. Se, ad esempio, i flip-flop sono di tipo D, le funzioni di posizionamento sono per lo *shl(Q,D)*:

$$d_{i+1} = Q_i \quad \forall i:\ 0 \le i \le n-2; \quad d_0 = D$$

I registri a scorrimento sono tipicamente utilizzati come:
- convertitori serie-parallelo, se dotati di uscita parallela, cioè se gli n flip-flop sono simultaneamente accessibili dall'esterno;
- convertitori parallelo-serie, se dotati di un ingresso di "precaricamento" che opera in parallelo su tutti i flip-flop;
- macchine componenti reti sequenziali;
- contatori, se precaricati con particolari configurazioni; ad esempio, se precaricato con una stringa composta da un "1" e da n-1 caratteri di "0", un registro circolare ad n bit opera come contatore modulo-n;
- macchine aritmetiche, per generare prodotti e quozienti per 2;
- macchine poste a sostegno di algoritmi di elaborazione che operano su stringhe di bit.

Di seguito, per esemplificare il comportamento dei registri a scorrimento, ne viene riportato uno ispirato al prodotto commerciale 74164, con ingresso seriale e uscita parallela (il 74165, all'inverso, presenta ingresso parallelo ed uscita seriale).

Descrizione del circuito (fig. 14.1)

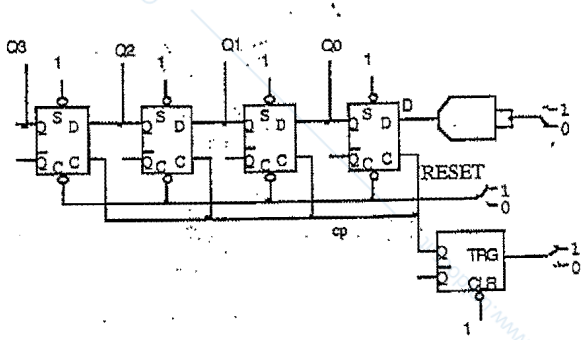


Fig. 14.1a: Registro a scorrimento

Il registro a scorrimento è "ridotto" rispetto a quello commerciale, a 8 bit, avendone solo 4 ed ha le seguenti particolarità:
- possiede un segnale RESET che azzera tutto il registro;
- possiede due morsetti di ingresso a e b, posti in and (D= a·b), per effetto dei quali può essere flessibilmente usato in vari modi:
  - ponendo a=b (come in figura) è il registro fondamentale;
  - ponendo a=b= Q_3 è un registro circolare;
  - ponendo a=*dato*, b=*controllo* si ha D=a oppure D=0 a seconda del valore di *controllo* e, quindi, si può adoperare il registro con *shl*(R, a) oppure *shl*(R, 0)=*ashl*(R).

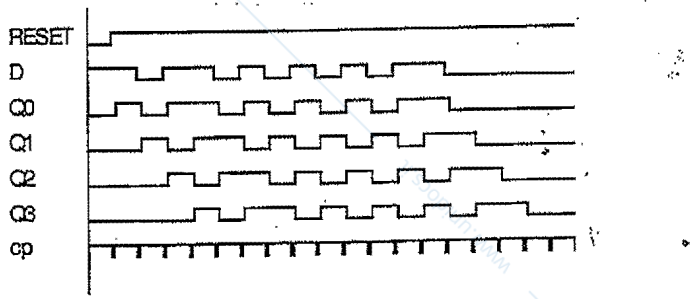


Fig. 14.1b: Registro a scorrimento

15. Registro a scorrimento bidirezionale

Tipo di circuito: rete sequenziale a sincronizzazione esterna
Riferimento: circuito commerciale 74194
Obiettivo: progettazione di apparato

Testo

Progettare un registro a scorrimento che, in funzione di due segnali binari a livelli (S_0, S_1), in presenza di un segnale di sincronizzazione cp, abbia quattro differenti funzionalità:
- S_0=0, S_1=0: il registro è disabilitato (resta inalterato);
- S_0=1, S_1=0: il registro opera come registro a scorrimento verso sinistra;
- S_0=0, S_1=1: l registro opera come registro a scorrimento verso destra;
- S_0=1, S_1=1: il registro viene caricato in parallelo dall'esterno.

Impostazione del progetto

Delle 4 operazioni, le ultime 3 comportano, per ciascun flip-flop del registro, il caricamento di un differente dato, la prima nessun caricamento. L'economia globale del progetto suggerisce tuttavia di realizzare comunque un caricamento del flip-flop, anche nel primo caso, benché fittizio: il flip-flop viene caricato da se stesso. Il progetto può allora essere svolto considerando che la linea dato del singolo flip-flop sia caricata da 4 differenti dati, selezionati da un multiplexer indirizzato dai 2 bit S_0 e S_1. Gli n multiplexer binari hanno quindi gli ingressi-indirizzo collegati in parallelo, mentre ciascuno ha le linee-dato collegate ai bit omologhi. In particolare, le linee-dato del multiplexer i-esimo sono così collegate:
- la prima, selezionata da S_0=0, S_1=0, è dedicata al mantenimento del dato: è collegata all'uscita del bit i-esimo stesso;
- la seconda, selezionata da S_0=1, S_1=0, produce uno scorrimento verso sinistra: è collegata all'uscita dell'(i-1)-esimo bit;
- la terza, selezionata da S_0=0, S_1=1, produce uno scorrimento verso destra: è collegata all'uscita dell'(i+1)-esimo bit;
- la quarta, selezionata da S_0=1, S_1=1, produce il caricamento dall'esterno di un dato: è collegata allo i-esimo bit del "parallel-in".

Descrizione del circuito (fig. 15.1)

Per semplicità si è realizzato lo schema di un registro a 3 bit e per la sua realizzazione si sono usati come componenti flip-flop D e un multiplexer indirizzabile ad 8 ingressi. Se il circuito andasse realizzato ex-novo, al loro

posto potrebbero andare un flip-flop RS con S=D, R=$\overline{D}$ ed un circuito a due livelli per il multiplexer. Su questa base è realizzato il componente commerciale 74194.

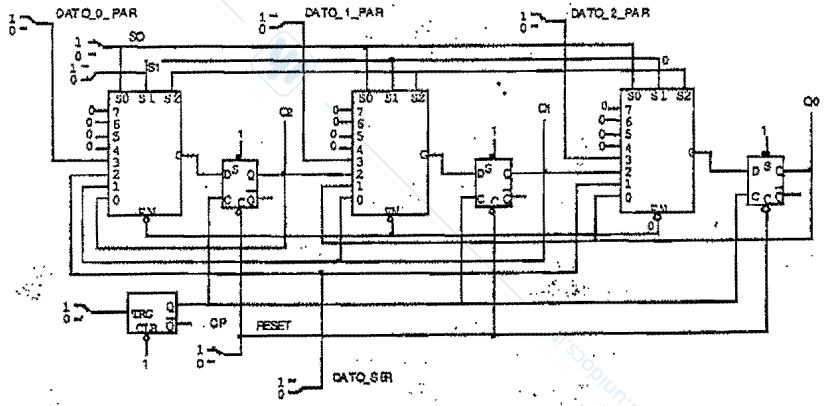


Fig. 15.1a: Registro a scorrimento multifunzione

Si noti che:

- non disponendo di un multiplexer a 4 ingressi, se ne è utilizzato uno ad 8, utilizzandone solo 4; le linee non utilizzate non possono essere selezionate, avendo posto la linea di selezione S_2=0;
- le linee dato non utilizzate del multiplexer sono poste ad un valore di riferimento pari a 0 unicamente per fissare il loro potenziale;

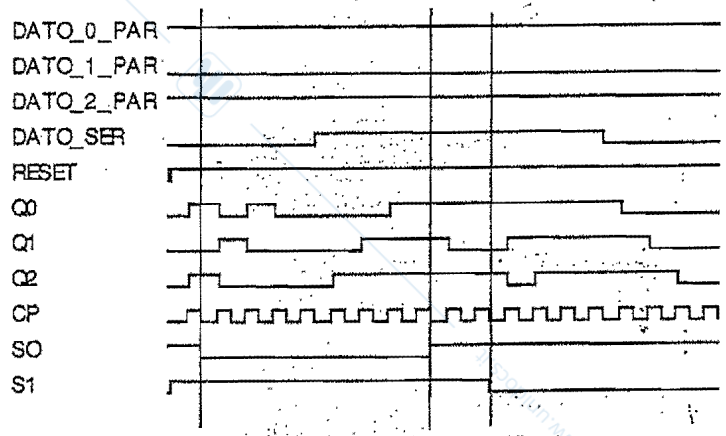


Fig. 15.1b: Registro multifunzione: tempificazione

- nel caso di scorrimento verso destra, la prima cella (Q_0) riceve il dato seriale di ingresso; mentre nel caso di scorrimento verso sinistra è l'ultima cella (Q_2) a ricevere il dato seriale;
- la cella centrale (Q_1) è connessa sia alla cella precedente che alla cella successiva.

Le tre linee verticali nel diagramma di tempificazione individuano rispettivamente la fase di shift verso destra, mantenimento e shift verso sinistra.

16. Generatore di sequenza con registro a scorrimento

Tipo di circuito: rete sequenziale a sincronizzazione esterna
Riferimento: circuito commerciale 74165
Obiettivo: uso di registri a scorrimento

Testo

Un registro a scorrimento, in quanto trasformatore parallelo-serie, è un generatore di sequenze. Si progetti un generatore di sequenze di 16 bit ciclico, nel senso che ripete ciclicamente la medesima sequenza assegnata.

Progetto

Per la realizzazione del progetto è sufficiente un registro a scorrimento circolare di 16 bit, con caricamento parallelo ed uscita seriale, dotato di un segnale *SH/LD* che discrimina il caricamento (SH/LD=0) dallo shift (SH/LD=1). La sequenza viene dapprima caricata nel registro (*fase di load*) e poi presentata sull'output (*fase di shift*), ricostituendosi man mano nel registro attraverso il collegamento circolare fra l'ultimo ed il primo bit dello stesso registro.

Il clock scandisce i bit della sequenza, la sua frequenza determina la forma d'onda complessiva del segnale generato.

Descrizione del circuito (fig. 16.1)

Il circuito è ottenuto collegando in serie due registri commerciali ad 8 bit del tipo 74165, operanti sul fronte di salita del segnale di sincronismo. I registri sono montati con l'uscita seriale del primo (*serial-out*) collegata con l'ingresso seriale del secondo (*serial-in*). La struttura circolante si è ottenuta con il collegamento fra serial-out del secondo e serial-in del primo.

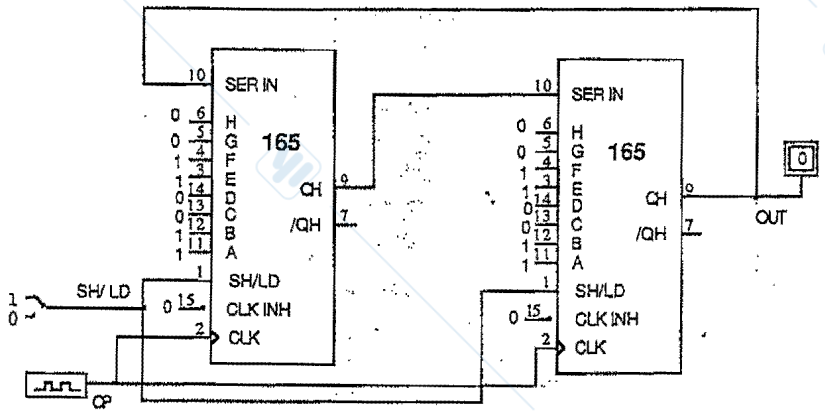


Fig. 16.1a: Generatore di sequenze

I registri vengono caricati parallelamente dalla sequenza assegnata (si è esemplificata la sequenza 0011001100110011) quando il switch SH / LD è in posizione "load" (ingresso SH/LD=0) ed opera da shift-register quando è in posizione "shift" (ingresso SH/LD=1), presentando la sequenza generata sull'uscita "OUT" del secondo registro.

Si noti in particolare che la sequenza esemplificata, in quanto costituita da 2 bit "1" seguiti periodicamente da 2 bit "0", costituisce anche una forma d'onda con una frequenza 1/4 della frequenza del clock: con questo metodo è anche possibile ottenere, dunque, la divisione in frequenza di un segnale. Dal diagramma di tempificazione si nota come la sequenza di uscita sia sincronizzata con il fronte 0→1 del clock, per cui si ha un valore nullo in uscita (OUT) per i primi due colpi di clock , unitario per i successivi due e così via.

Fig. 16.1b: Generatore di sequenze: tempificazione

Si noti che, inizializzando il registro con opportune sequenze, è possibile ottenere, oltre alla divisione in frequenza, anche uno sfasamento. Ad esempio, le due sequenze: 0011...0011 e 1100...1100 forniscono due segnali tra loro sfasati di mezzo periodo.

Capitolo sesto
Reti composte

1. Sistemi e reti composte

Un sistema è in genere realizzato ponendo assieme più reti logiche fra di loro opportunamente collegate. Il sistema nella sua interezza è una macchina sequenziale, ma non sempre è facile né opportuno affrontarne il progetto con la ricerca di un modello unico che lo rappresenti. Più spesso è invece utile trattare il sistema come composto da macchine distinte che si progettano separatamente e poi si collegano fra di loro: il sistema infatti diventa allora più controllabile, secondo l'antico detto "divide et impera", adoperato anche con successo nella progettazione del software.

Le singole macchine componenti il sistema risolvono ciascuna uno dei sottoproblemi del problema che il sistema deve affrontare; l'individuazione di tali macchine è dunque tipica di ciascun problema, del quale occorre individuare i sottoproblemi che lo compongono. Le macchine sequenziali componenti sistemi più complessi sono tipicamente:

- contatori, in tutti quei casi in cui nel problema esiste un aspetto di conteggio;
- registri a scorrimento, in tutti quei casi in cui occorre serializzare o parallelizzare un dato;
- registri a scorrimento ciclici con funzioni di contatori;
- singoli flip-flop con funzioni di controllo.

A queste macchine sequenziali fondamentali si aggiungono di volta in volta macchine specifiche.

Si pongono ora in evidenza alcuni aspetti del collegamento fra macchine.

Decomposizione in macchine componenti

Una macchina M(Q) avente $Q=\{q_1, q_2, ...\}$ come insieme degli stati si può sempre decomporre in due macchine componenti, $M_1(Q_1)$, $M_2(Q_2)$: dette

infatti P_1, P_2 due partizioni di Q e P_0 la partizione degenere (partizione nulla) di Q avente come elementi gli stati q_i, e supposto:

$$P_1 \times P_2 = P_0$$

è sufficiente che ciascun elemento di P_1 sia associato ad un elemento di Q_1 ed analogamente P_2 ad un elemento di Q_2. L'elemento del prodotto cartesiano di due partizioni è l'intersezione fra le coppie di elementi (cioè gli elementi comuni alle due) e quindi la relazione di cui sopra significa che le coppie a due a due si intersecano in un unico elemento di Q e che tutti gli elementi di Q sono generati da tali intersezioni; per fissare le idee si veda la fig.1.1, ove gli stati di M sono stati disposti alla intersezione fra righe e colonne di una matrice ideale. Le righe determinano una partizione (nell'esempio, gli elementi sono a=1,2,3; b=4,5; c=6,7), così come le colonne (A, B, C) e il prodotto cartesiano delle due è proprio la partizione P_0, ad esempio, la coppia (a, A) del prodotto è 1, (a, B) è 2 de così via. La macchina

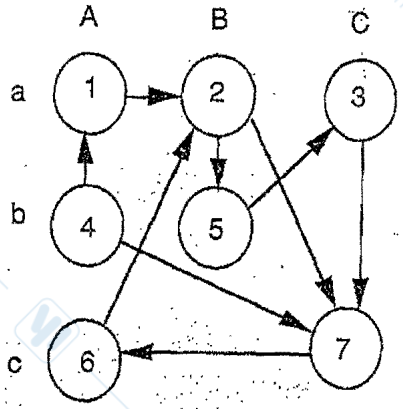


Fig.1.1: Decomposizione di una macchina

M si può quindi decomporre nelle macchine M_1 con Q_1={a, b, c} ed M_2 con Q_2={A, B, C}; la macchina composta avrà come stati il prodotto cartesiano $Q_1 \times Q_2$ che in realtà include Q ($Q \subseteq Q_1 \times Q_2$) in quanto alcune coppie del prodotto non corrispondono ad alcuno stato di Q (ad esempio la coppia B, c); le macchine M_1 e M_2 si possono progettare derivandone le tabelle di stato ed uscita da quelle di Q: lo stato seguente e l'uscita sarà in generale funzione di tutti gli stati di Q (e quindi di quelli di Q_1 e Q_2), mentre nei punti ove l'intersezione è vuota (come B, c) si hanno punti di non specificazione.

Nel caso in cui le partizioni non godano di nessuna particolare proprietà, gli stati seguenti di ciascuna delle due macchine componenti sono funzioni, oltre che degli ingressi e dei propri stati, anche degli stati dell'altra macchina. In tali circostanze, sviluppare il progetto con l'uso di M_1, M_2, non conduce in generale ad alcuna concreta semplificazione, in quanto occorrerebbe in ogni caso far riferimento alla totalità degli stati in ogni passo del progetto e le due macchine risultanti sarebbero strettamente interconnesse.

Se viceversa almeno una partizione è "chiusa", il progetto di M si può scomporre in due progetti distinti e il circuito risultante si realizza con le due macchine non così strettamente interconnesse. Una partizione è chiusa se, per ogni ingresso, l'insieme degli stati seguenti di un elemento della partizione è incluso in un unico elemento della stessa partizione. Sul grafo, ciò equivale a dire che per ogni ingresso, gli stati seguenti di una riga (colonna) appartengono tutti ad una medesima riga (colonna). Si vedano gli esempi ai paragrafi seguenti.

Supposta P_1 chiusa e P_2 no, ne risulta lo schema di fig.1.2a, detto di composizione "in serie": M_1 è indipendente da M_2, gli stati seguenti di M_2 dipendono, oltre che da quelli di M_2 stessa, anche da quelli di M_1; le uscite si realizzano in una macchina combinatoria C, in funzione degli ingressi e degli stati di M (e quindi di M_1 e M_2).

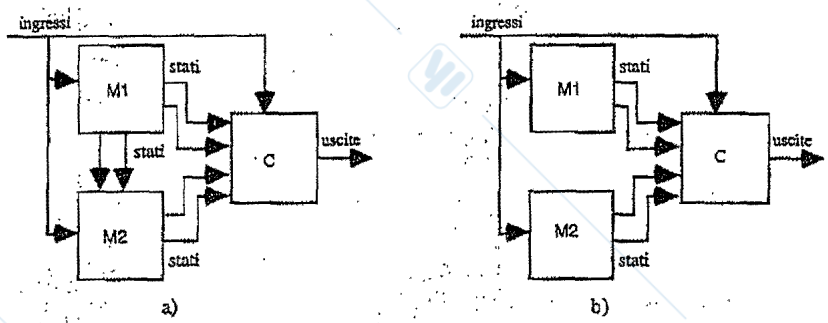


Fig.1.2: Decomposizione di macchine: a): schema seriale ; b): schema parallelo

Se anche P_2 è chiusa, lo schema di collegamento è "parallelo": le due macchine sono del tutto indipendenti fra loro.

La tempificazione delle macchine componenti e di quella risultante sono coerenti: se ad esempio M è sincrona, anche M_1 ed M_2 lo sono e possono essere sincronizzate in parallelo dallo stesso impulso (vedi esempi ai paragrafi successivi).

Un progetto che si basi sulla teoria di cui sopra è detto *progetto per decomposizione*; invero, la teoria di cui sopra si attua a verifica di una idea intuitiva: riconosciuto nel problema un sottoproblema che sarebbe risolto da una apposita macchina (eventualmente precostituita), si applica la teoria di cui sopra per perfezionare il progetto. E' quanto si vedrà in molti degli esempi che seguono.

Uso di clock a più fasi

Se una rete è sincronizzata da un impulso sincrono con il clock c_1, è necessario che gli ingressi siano stabili all'avvento dell'impulso. Se allora questi sono uscite di un'altra rete, quest'ultima non può essere sincronizzata anche essa al tempo c_1, pena l'insorgere di alee. Questa può invece essere sincronizzata da impulsi sincroni con un'altra fase del clock, c_2. Si veda in proposito l'esempio dei §§ 5 e 7.

Collegamento con i segnali via-fine

Una rete che debba sviluppare un algoritmo su comando di un'altra rete, riceve di solito da quest'ultima un segnale *via*, a seguito del quale inizia ad operare; al termine dell'operazione, la rete comunica con un altro segnale, *fine*, il fatto che il risultato è disponibile per la rete che lo ha richiesto. I segnali *via* e *fine* sono ingressi delle reti in esame ed operano con la tempificazione di queste: a livelli, impulsivi, sul fronte. Si veda l'esempio del § 5.

2. Riconoscitore di codice 8421 con contatore

Tipo di circuito: rete composta
Riferimento: RL, XI - 4
Obiettivo: progettazione di sistemi

Testo

Costruire, adoperando un contatore per il conteggio dei 4 bit, una rete nella quale entrano serialmente i bit di un codice decimale 8-4-2-1 a partire dal bit meno significativo e dalla quale esce un segnale che individua se i quattro bit costituiscono o meno una delle 10 parole-codice previste. La rete sia inizializzata da un segnale di reset (le specifiche del circuito coincidono con quelle del § V-6).

Tabella e grafo di stato

La tabella e il grafo coincidono con quelli del § V-6, che sono qui riportati (fig.2.1).

stati \ x	1	0
S_0	$S_1/0$	$S_1/0$
S_1	$S_2/0$	$S_3/0$
S_2	$S_4/0$	$S_4/0$
S_3	$S_4/0$	$S_5/0$
S_4	$S_0/0$	$S_0/1$
S_5	$S_0/1$	$S_0/1$

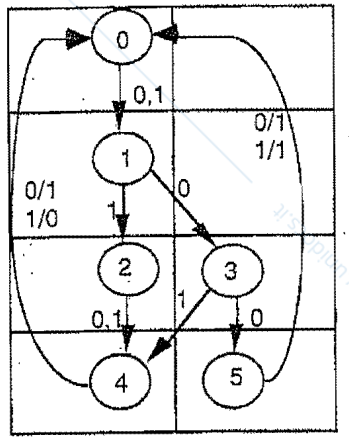


Fig. 2.1: Riconoscitore 8421: tabella e grafo

Progetto per decomposizione

Nel disporre i nodi del grafo di stato, si è avuto cura di schierarli all'incrocio fra 4 righe (a, b, c, d) e due colonne (A, B). Ciò corrisponde allo aver individuato due partizioni, l'una rappresentata dalle righe e l'altra dalle colonne: le righe partizionano gli stati in P_1=(a, b, c, d) = (0; 1; 2,3; 4,5), le colonne in P_2 = (A, B)=(0,1,2,4; 3,5); il prodotto cartesiano delle due partizioni è la partizione nulla, cioè l'insieme degli stati di M non ripartito: $P_1 \times P_2 = P_0$.

Ripartiti gli stati come sopra, la macchina M può essere realizzata con 2 macchine componenti, l'una, M_1, con gli stati corrispondenti alla partizione P_1=(a , b, c, d) (cioè alle righe dello schieramento), l'altra, M_2, corrispondente a P_2 = (A, B) (alle colonne). Ogni stato di M è corrispondente ad un elemento di $P_1 \times P_2$, cioè è alla intersezione di una riga ed una colonna.

Delle due partizioni, P_1 è chiusa; sul grafo di fig.2.1, si può infatti verificare che per ogni ingresso, gli stati seguenti di una riga appartengono tutti ad una medesima riga: ad *a* per entrambi gli ingressi segue *b*, a *b* segue *c* e così via, come nella tabella e nel grafo di fig. 2.2.

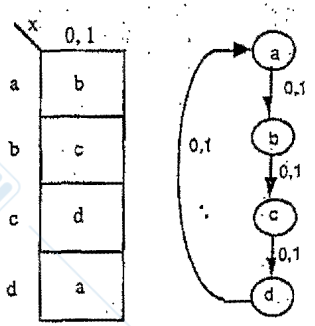


Fig. 2.2: Partizione sulle righe del grafo di fig.2.1

La partizione sulle colonne invece dà luogo alla tabella e al grafo di figura 2.3. Essa non è chiusa, in quanto, ad esempio, dalla colonna (stato) A, per l'ingresso 0 si va a B se si parte dalla riga b, si resta in A altrimenti: gli stati seguenti della macchina M_2 dipendono anche dagli stati di M_1.

x	1	0
aA	bA	bA
bA	cA	cB
cA	dA	dA
cB	dA	dB
dA	aA	aA
dB	aA	aA

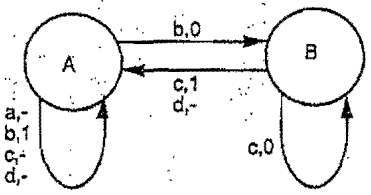


Fig. 2.3: Partizione sulle colonne del grafo di fig.2.1

In fig. 2.4, la tabella di fig. 2.3 è trasformata nella tabella di stato di M_2, ponendone gli stati di M_1 come ingressi unitamente con l'ingresso di M, x. Sono anche indicate le uscite della macchina originaria M, funzioni ovviamente degli stati di M_1, M_2 e degli ingressi; si notino i punti di indifferenza all'incrocio della riga B con le colonne a, b.

La macchina M è dunque realizzabile con M_1 ed M_2 collegate fra loro come nello schema di fig. 2.5, detto seriale in quanto M_2 appare posta in serie ad M_1. Le uscite sono computate in una rete combinatoria C che opera sugli ingressi e sugli stati di M (e quindi di M_1 ed M_2).

stati di M_1, x	a,1	a,0	b,1	b,0	c,1	c,0	d,1	d,0
A	a,1	a,0	b,1	b,0	c,1	c,0	d,1	d,0
B	-	-	-	-	A/0	B/0	A/1	A/1

Fig. 2.4: Tabella di stato di M_2

Nel caso specifico, solo M_2 è funzione dell'ingresso x, mentre il clock opera in parallelo su M_1, M_2 e su C (per le eventuali uscite impulsive). Si noti in proposito che la definizione di "seriale" attribuita alla macchina non significa che le due componenti operano in serie nel tempo.

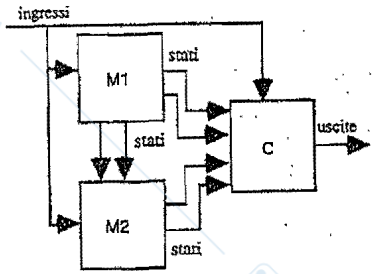


Fig. 2.5: Schema di collegamento in serie

Progetto delle macchine componenti

Macchina M_1

E' un contatore modulo-4, senza altri ingressi oltre il clock e il reset.

Macchina M_2

E' sufficiente un'unica variabile di stato, y, che si realizza in un flip-flop RS. Le funzioni di posizionamento sono allora:

$$\text{SET} = \bar{x} \cdot b \qquad \text{RESET} = \bar{x} + d$$

e si possono ricavare dalla tabella di stato della macchina oppure più semplicemente notando che (cfr. il grafo originario) y è settato solo per x=0 nello stato b di M_1 ed è resettato sempre per x=1 e nello stato *d* di M_1.

Macchina combinatoria C

L'uscita z a livelli è dedotta dalla tabella della macchina M, esprimendone gli stati in funzione di quelli di M_1, M_2:

$$z = d\cdot\bar{x} + d\cdot y$$

Essa è infatti alta solo nello stato d, se è x=0 (il bit più significativo della sequenza è 0) oppure, sempre nello stato d, ma con M_2 nello stato B (la sequenza dei primi 3 bit è -00).

Descrizione del circuito (fig.2.6)

Confrontare il circuito con quello del § V-6 del quale questo è una variante:
- x, c, reset, z, z' hanno i medesimi significati e sono realizzati alla medesima maniera.
- Il "cuore" del circuito è costituito dalle macchine della decomposizione M_1, M_2, C.

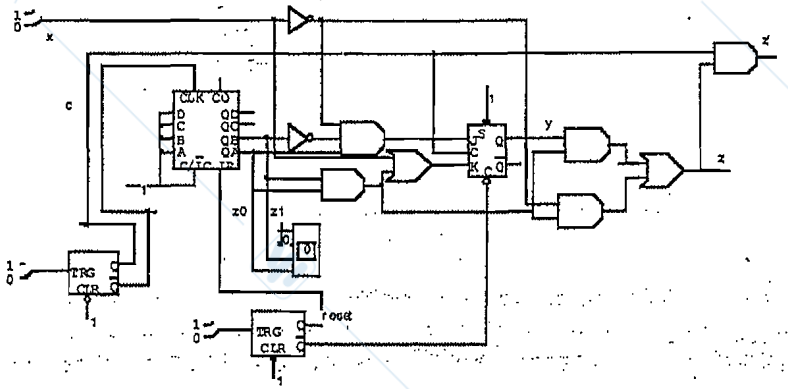


Fig. 2.6 a: Riconoscitore di 8421 con contatore

Macchina M_1

- il contatore modulo-4 è realizzato con un contatore modulo-16 commerciale del quale si adoperano solo le due uscite meno significative;
- esso conta con la sequenza 0-1-2-3-0-1-2-3 ....., codificata sulle due variabili z_1, z_0;
- il reset 1-attivo del contatore è sull'ingresso CLR e lo pone nello stato 0;

- gli ingressi di load del contatore non sono adoperati.

Macchina M_2

- per lo stato y si è adoperato un flip-flop JK edge triggered sul fronte di discesa, usato come RS abilitato;
- il flip-flop è resettato con un impulso 0-attivo sul clear;
- vista la codifica degli stati di M_1, le funzioni di posizionamento si trasformano in:

$$\text{SET} = \bar{x}\cdot\bar{z}_1\cdot z_0 \qquad \text{RESET} = \bar{x}\cdot y + y\cdot z_0$$

Macchina combinatoria C

- tenendo conto della codifica degli stati di M_1, M_2, si ha:

$$z = z_1 z_0.\bar{x} + d\cdot y$$

- l'uscita impulsiva z' è semplicemente:

$$z' = z\cdot c$$

Tempificazione

Valgono le stesse considerazioni del citato circuito del § V-6 cui si rimanda. Si noti soltanto che, poiché il contatore adoperato è attivo sul fronte di salita e tutta la rete deve reagire sincronicamente sul secondo fronte di c, l'ingresso del contatore è eccitato con $\bar{c}$.

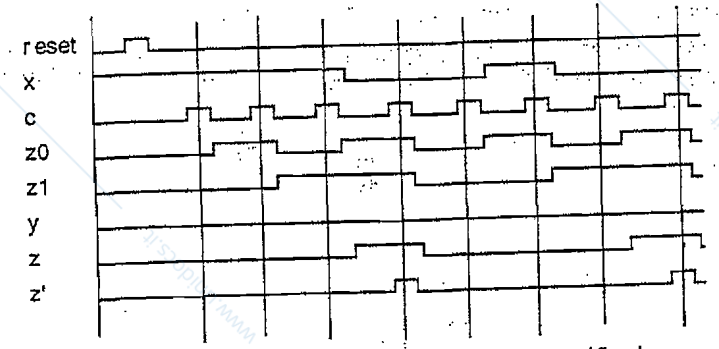


Fig. 2.6b: Riconoscitore di codice 8421 con contatore: tempificazione

3. Decomposizione parallela.

Tipo di circuito: rete composta
Riferimento: RL, XI - 5
Obiettivo: progettazione di sistemi

Testo

Costruire una rete che implementi la tabella di figura 3.1.

Progetto

La macchina M è descritta dalla tabella data. Ad essa è stato associato il diagramma di stato (fig. 3.1).

x	0	1
S_0	S_1	S_2
S_1	S_5	S_3
S_2	S_4	S_0
S_3	S_2	S_1
S_4	S_3	S_5
S_5	S_0 /1	S_4

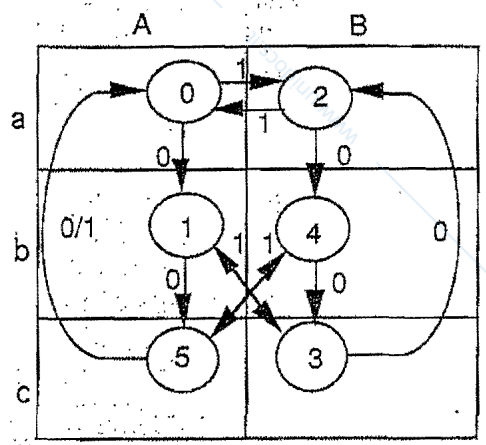


Fig. 3.1: Decomposizione parallela: tabella e grafo

Nel disporre i nodi del grafo di stato, si è avuto cura di schierarli all'incrocio fra 3 righe (a, b, c) e 2 colonne (A, B). Ciò corrisponde all'aver individuato due partizioni, l'una rappresentata dalle righe e l'altra dalle colonne: le righe partizionano gli stati in P_1= (a, b, c) = (0,2; 1,4; 3,5), le colonne in P_2=(A, B)=(0,1,5; 2,4,3); il prodotto cartesiano delle due partizioni è la partizione nulla: $P_1 \times P_2 = P_0$. La macchina M può essere realizzata con le macchine componenti M_1 ed M_2 corrispondenti a P_1 e P_2 rispettivamente.

Le due macchine sono rappresentate dai grafi e dalle tabelle che seguono. Da essi si evince anche che entrambe le partizioni sono chiuse: l'evoluzione fra a, b, c è indipendente da quella fra A, B e viceversa.

x	0	1
a	b	a
b	c	c
c	a	b

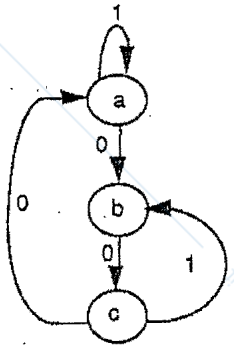


Fig. 3.2: Macchina M_1

x	0	1
A	A	B
B	B	A

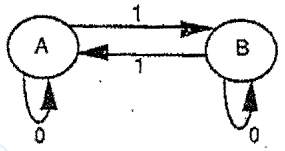


Fig. 3.3: Macchina M_2

La macchina M è, dunque, realizzabile con le macchine M_1 ed M_2 collegate fra loro come nello schema parallelo di figura 3.4.

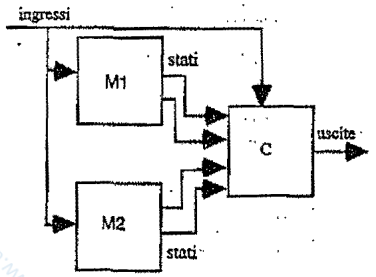


Fig. 3.4: Schema di decomposizione parallela

Progetto delle macchine componenti

Macchina M_1

- variabili di stato: y_2, y_3
- codifica degli stati: a = (00), b=(01), c= (10)
- flip-flop di tipo D, sincronizzato da x
- variabili di posizionamento:

$$d_2 = y_3 \qquad d_3 = y_2 \cdot x + \bar{y}_2 \cdot \bar{y}_3 \cdot \bar{x}$$

Macchina M_2

E' un flip-flop T, sincronizzato da x; si pone $A = \bar{y}_1$, $B = y_1$.

Macchina combinatoria C

L'uscita z impulsiva è dedotta dalla tabella della macchina M, esprimendone gli stati in funzione di quelli di M_1, M_2:

$$z = S_5 \cdot \bar{x} = \bar{A} \cdot c \cdot \bar{x} = \bar{y}_1 \cdot y_2 \cdot \bar{y}_3 \cdot \bar{x}; \qquad z' = z \cdot c$$

ed, essendo non specificata la combinazione delle variabili y_2 y_3, si ha infine:

$$z = \bar{y}_1 \cdot y_2 \cdot \bar{x}; \qquad z' = z \cdot c$$

Descrizione del circuito (fig. 3.5)

- un display esadecimale evidenzia lo stato complessivo; considerando le assegnazioni effettuate, si ha la seguente corrispondenza fra lo stato di M e il codice espresso dal display:

	$y_1y_2y_3$	codice
S_0	000	0
S_1	001	1
S_2	100	4
S_3	110	6
S_4	101	5
S_5	010	2

- reset = 0 pone la rete nello stato 0.

Il circuito è composto dalle macchine:

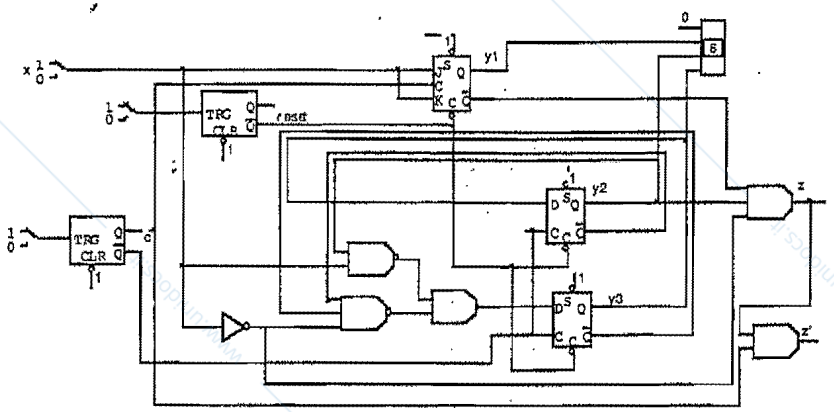


Fig. 3.5a: Rete progettata con decomposizione parallela

- M_1, realizzata con flip-flop edge-trigged sul fronte di salita, attivati dal clock cn ($\bar{c}$) sincrono con c sul secondo fronte;
- M_2, realizzata con un flip-flop JK edge-trigged sul fronte di discesa (secondo fronte di c);
- C (macchina combinatoria): che ha l'uscita z' sincrona con l'impulso c.

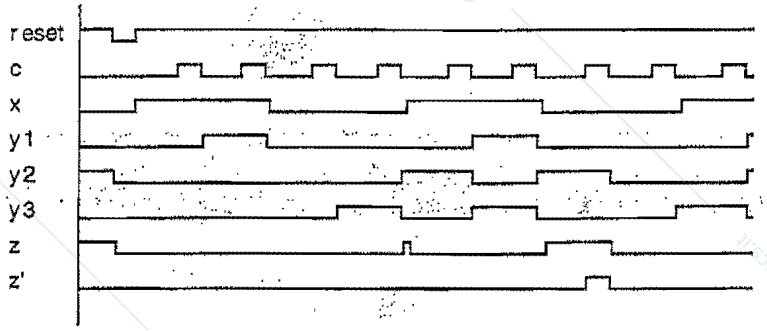


Fig. 3.5b: Rete progettata con decomposizione parallela: tempificazione

Tempificazione

La tempificazione è quella classica delle reti sincrone a sincronizzazione esterna: l'uscita impulsiva è costruita mediante la AND con l'impulso, mentre

gli stati evolvono con il secondo fronte di quest'ultimo. Poiché il flip flop JK è sul fronte di discesa e i flip flop D su quello di salita, essi sono attivati dalle due uscite opposte del monostabile.

4. Controllo di parità sequenziale

Tipo di circuito: rete composta
Riferimento: RL, XI - 5
Obiettivo: progettazione di sistemi

Testo

Costruire una macchina M per il controllo di parità di un byte. Il byte è sincronizzato con un clock. Studiare la tempificazione dell'uscita.

Diagramma di stato

La macchina è descritta dal diagramma di stato di fig.4.1, disegnato in modo da mettere in evidenza le due macchine componenti del problema: un contatore mod-8 (sulle righe) ed un flip-flop trigger (sulle colonne). In figura sono mostrate due versioni; si analizza dapprima la versione a).

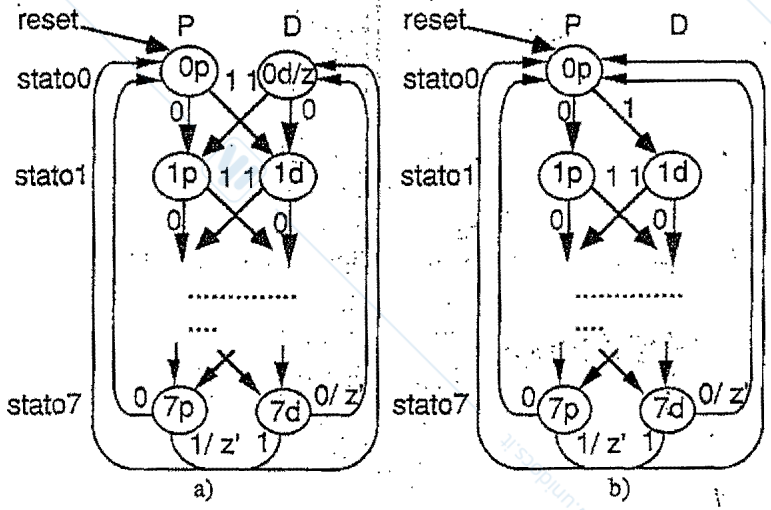


Fig. 4.1: Diagramma di stato della rete di parità seriale

Gli stati rappresentano l'intervallo fra i bit e tengono memoria se la sequenza pervenuta fino ad allora sia pari o dispari; lo stato *ip* indica che la sequenza è pari fino al bit i-esimo, id che è dispari; gli stati 0p, 0d sono quelli che indicano la parità della sequenza precedente e quindi 0d indica che il byte precedente è dispari. Un segnale di reset conduce la macchina nello stato 0p prima di ogni sequenza di bit.

Sono previste (e saranno in seguito discusse) due uscite: l'una, z, di Moore, a livelli, che identifica lo stato 0d. L'altra, z', di Mealy, impulsiva, piazzata sulle transazioni verso 0d.

Progetto

Sulla base del diagramma di fig.4.1a le due partizioni che generano le macchine sono entrambe chiuse e pertanto lo schema risultante è quello di figura. 4.2.

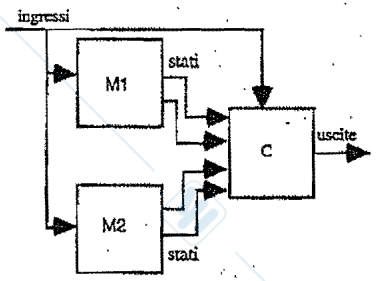


Fig. 4.2: Controllore di parità: schema generale

Detto quindi x il bit in ingresso, c il clock e z, z' le uscite a livelli ed impulsiva, si ha:

$$z = stato0 \cdot D\,; \qquad z' = \left(stato7 \cdot D \cdot \overline{x} + stato7 \cdot P \cdot x\right) \cdot c$$

Nella soluzione b), non è prevista l'uscita a livelli (che nella soluzione a comunque ha problemi di tempificazione, vedi in seguito), ed allora gli stati 0p, 1p risultano equivalenti e sostituiti dal solo 0p; la rete è allora immediatamente pronta per l'esame di un byte dopo l'arrivo di quello precedente, senza necessità del segnale di reset. In tal caso, però, la partizione relativa alle colonne non è chiusa: il flip-flop infatti non è indipendente dal contatore, ma deve essere resettato in corrispondenza di *stato7*.

Descrizione del circuito (fig. 4.3)

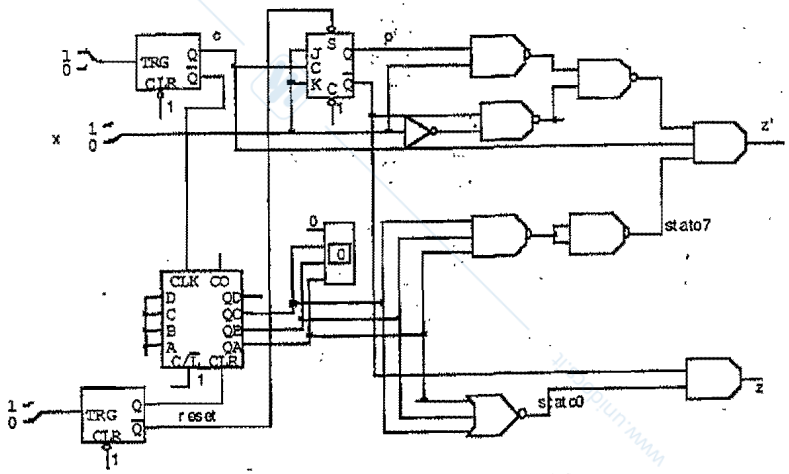


Fig. 4.3a: Controllore di parità sequenziale

La macchina implementata è quella della decomposizione parallela del grafo di fig. 4.1a. Si noti che:
- un display esadecimale evidenzia lo stato complessivo;
- il segnale *reset* pone la rete nello stato 0p, azzerando il contatore e ponendo a 1 (stato P, pari) il flip-flop.

Le tre macchine componenti sono le macchine:
- M_1 *(contatore mod-8)*, realizzata con un contatore mod-16, del quale si adoperano solo le tre uscite z_0, z_1, z_2, resettato dal segnale *reset* e attivo sul fronte di salita.

- M_2 *(flip-flop)*, realizzata con un flip-flop JK edge triggered sul fronte di discesa, posto in set (stato pari) dal segnale *reset*.

- *C (macchina combinatoria)*, che ha:
  l'uscita $z = stato0 \cdot D$ con *stato0* calcolato con una NOR;
  l'uscita z' calcolata mediante:

$$z' = (stato7 \cdot c) \cdot D \cdot \bar{x} + (stato7 \cdot c) + P \cdot x \cdot z$$

con *stato7* calcolato con una AND (NAND invertita).

Tempificazione

La tempificazione è quella classica delle reti sincrone a sincronizzazione esterna: l'uscita impulsiva è costruita mediante la AND con l'impulso mentre gli stati evolvono con il secondo fronte di quest'ultimo. Poiché il flip flop JK è sul fronte di discesa e il contatore su quello di salita, essi sono attivati dalle due uscite opposte del monostabile.

Si noti la criticità del segnale a livelli z: nella transizione 0p→1d vi è una variazione simultanea delle due variabili stato0 e D, chiaramente visibile nella simulazione di fig.4.3b. Per tale motivo è preferibile l'uscita impulsiva e la soluzione b.

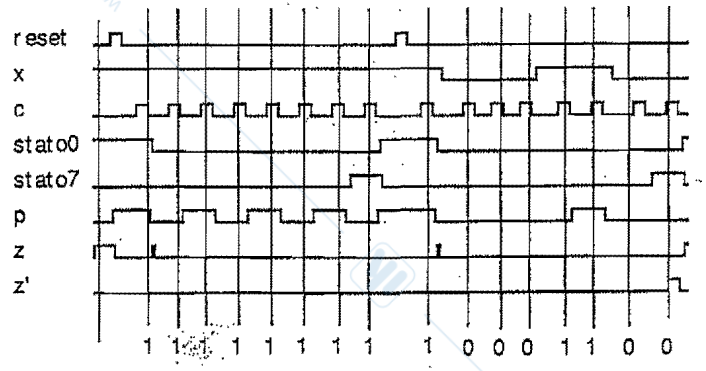


Fig. 4.3b: Controllore di parità sequenziale : tempificazione
(sono mostrate le sequenze 1111 1111 (pari) e 1000 1100 dispari)

Circuito con variante (parità sequenziale con decoder fig. 4.4)

In alternativa, si può adoperare un decoder per il calcolo di *stato0* e *stato7*. Nel circuito mostrato è adoperato un decoder commerciale (74154) a 4 ingressi (e 16 uscite), adoperandone solo 3 ingressi e 2 uscite. Il decoder fornisce uscite 0-attive e quindi si è operato come segue:
- l'uscita del decoder è negata per ottenere *stato7*;
- l'uscita z è ottenuta con una NOR dei negati di *stato0* e D.

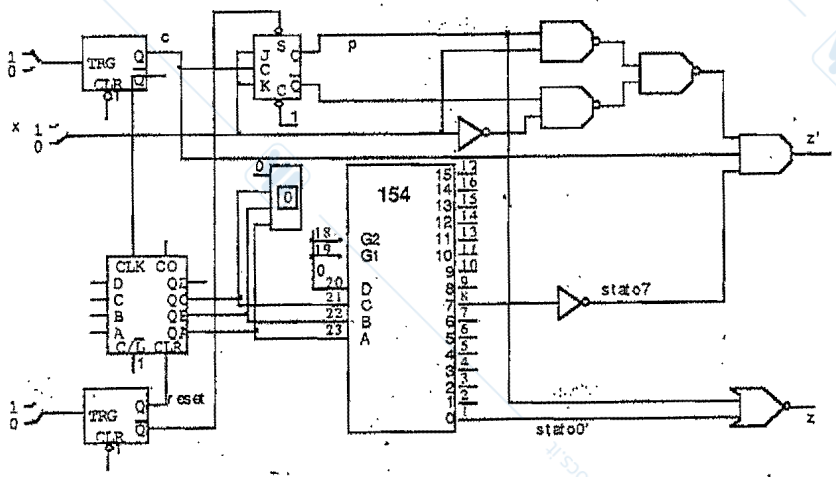


Fig. 4.4: Controllore di parità con decoder

5. Adder seriale

Tipo di circuito: rete composta
Riferimento: MEI, V - 3.
Obiettivo: progettazione di sistemi

Testo

Sviluppare il progetto di un addizionatore binario di interi positivi che operi serialmente su dati costituiti da due stringhe di n bit (in particolare si consideri n=8).

Analisi e completamento delle specifiche

La macchina nel suo complesso è costituita da:
- due registri-addendi A, B;
- un registro-somma S;
- un addizionatore seriale di n bit propriamente detto, ADD;
- un sistema di controllo della tempificazione complessiva, CONTR.

Oltre alla somma S=A+B, ADD deve fornire, al termine della esecuzione dell'addizione, un segnale binario R (variabile booleana) che indichi se l'addizione abbia provocato overflow:

$$R = A+B \geq 2^n$$

I registri-addendi sono registri a scorrimento, con i bit meno significativi che alimentano ADD, che a sua volta alimenta il registro-somma (fig. 5.1).

La macchina ADD è una rete sequenziale a sincronizzazione esterna; supponiamo che sia sincronizzata da un segnale di sincronismo C_2. I registri-

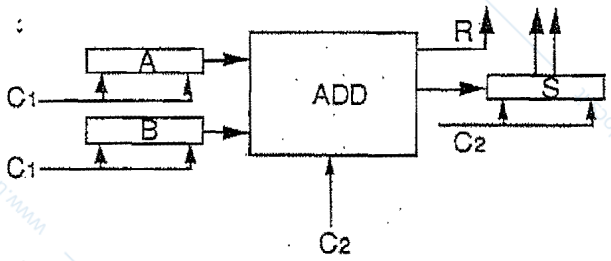


Fig. 5.1: Schema generale dell'addizionatore

addendi ne costituiscono gli ingressi a livelli e devono quindi variare nello spazio temporale fra due C_2 consecutivi; ipotizziamo che varino in sincronismo con un clock C_1 di fase diversa da C_2. Quindi, gli shift register addendi sono sincronizzati da C_1, mentre ADD è sincronizzato da C_2. Il registro-somma, dovendo acquisire l'uscita a livelli di ADD, può operare in sincronismo con ADD e quindi sul clock C_2.

Le fasi generali dell'operazione consistono in:
- caricamento dei registri-addendi: supponiamo che avvenga in parallelo (discuteremo in chiusura di questa ipotesi);
- esecuzione della addizione e graduale formazione della somma nel registro-somma;
- inoltro della somma effettuata verso le altre apparecchiature.

Tali fasi sono gestite dal sistema di controllo CONTR.

La macchina ADD

L'addizionatore di interi ADD consta di due componenti:
- un adder sequenziale di 1 bit, FA (Full adder),
- un contatore modulo-n per l'individuazione della fine dell'addizione.

Le due reti sono opportunamente collegate fra di loro: si tratta nel complesso di una macchina realizzata per decomposizione funzionale in serie o in parallelo (si veda il seguito).

L'addizionatore FA è una macchina a due stati interni (coincidenti con il riporto), due ingressi a, b (i bit da sommare), l'uscita s (il bit-somma) ed

opera secondo il diagramma di stato di fig. 5.2. Se il flip-flop di stato è di tipo D, la parte combinatoria della rete, che diremo BIT-ADD, è un full-adder combinatorio.

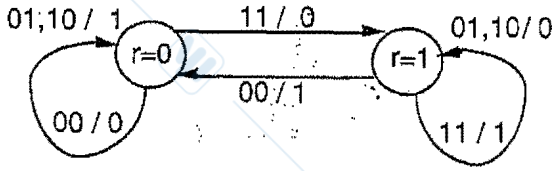


Fig. 5.2: Full adder seriale

La macchina completa (FA + contatore) è rappresentata da uno dei due grafi di fig. 5.3 (cfr. anche §4, ove c'è un caso analogo). Lo "stato 0" è lo stato iniziale del contatore ed è anche lo stato che la macchina raggiunge dopo aver sommato l'ultimo bit. Il segnale reset conduce la macchina complessiva nello stato iniziale del contatore, con riporto r=0.

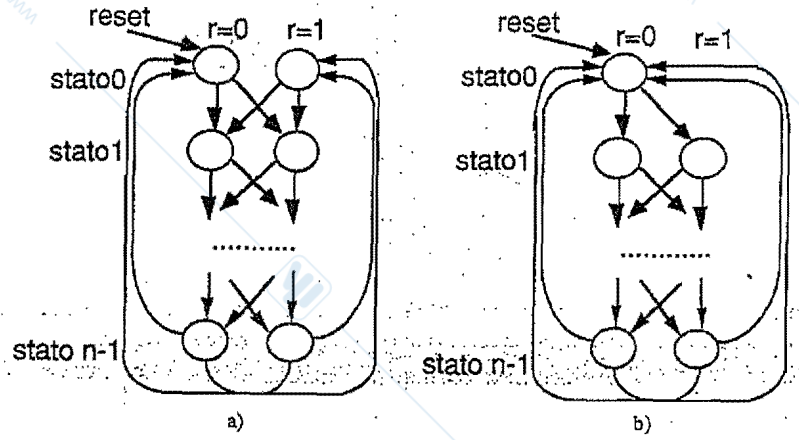


Fig. 5.3: Addizionatore ad n bit

In fig.5.3a), allorché il contatore è nello stato 0, lo stato interno di FA rappresenta anche il bit-overflow, in quanto riporto dopo l'n-esimo bit; prima di ogni sequenza da addizionare, è necessario il segnale di reset, che pone inizialmente r=0. In fig. 5.3b), invece, non esiste un segnale a livello che individui l'overlow, ma questo può soltanto essere segnalato da una uscita impulsiva nel passaggio da uno dei due "stati n-1" allo "stato 0"; per contro,

non è necessario il segnale reset se non la prima volta, in quanto la macchina, appena terminata una addizione, è subito pronta per realizzarne un'altra.

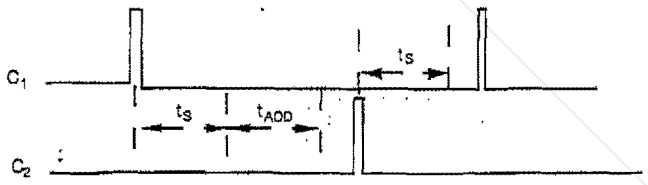


Fig. 5.4: Tempificazione fra addizionatore e registri

Il confronto fra le due soluzioni è legato alla tempificazione. In fig. 5.4 è mostrata la tempificazione della apparecchiatura completa fra due bit da addizionare (cfr. anche fig. 5.1): in un istante sincrono con C_1 è innescato lo shift nei registri-addendi, che si completa in un tempo t_s; sono quindi disponibili gli addendi dell'addizione e la parte combinatoria dell'addizionatore binario impiega un tempo t_{ADD} per rendere disponibile somma e riporto uscente. Dopo un intervallo di sicurezza, quindi, in sincronismo con C_2, la somma è inviata al registro-somma e il riporto uscente è memorizzato nello stato di FA. Quindi, il registro-somma è anch'esso sincronizzato con C_2, e assorbe compiutamente la somma dopo un ulteriore tempo t_s.

Stante questa tempificazione, il confronto fra le due soluzioni di fig. 5.3 lascia preferire la prima, in quanto con la seconda il bit-overflow impulsivo, che è sincrono con l'ottavo impulso C_2, richiederebbe poi un ulteriore elemento di memoria per mantenersi disponibile fino al tempo in cui è disponibile anche la somma completa nel registro S.

Il sistema di controllo CONTR

Distingueremo l'operazione complessiva nelle 3 fasi di cui all'analisi delle specifiche:

a) Caricamento dei dati nei registri-addendi: in questa fase vengono anche resettati il flip-flop riporto ed il contatore, in modo che tutto sia pronto per l'esecuzione della somma.

b) Esecuzione dell'addizione: una volta che i registri-addendi siano stati caricati, l'addizionatore deve partire, nel senso che devono essere resi attivi ADD e i registri a scorrimento. Una rete di controllo, allora, può generare, dal clock generale di macchina, una sequenza di 8 clock da inviare alle macchine in questione.

c) Inoltro del risultato: consiste semplicemente nell'inviare verso le apparecchiature-utenti la somma accumulatasi in S e il bit-overflow; è sufficiente

allo scopo un apposito strobe, generato quando il dato nel registro-somma è stabile, quindi all'ottavo C_1.

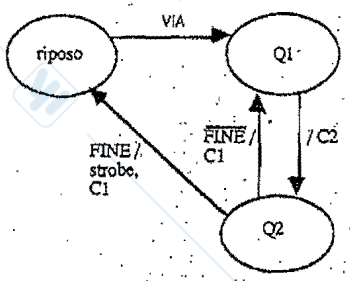


Fig. 5.5: Grafo di transizione della rete di controllo

La rete di controllo opera secondo il grafo di fig.5.5, costruito nella ipotesi che tutte le transizioni siano sincrone con un unico clock generale di macchina e che da questo la rete derivi la sequenza di 8 coppie distinte C_2 e C_1. Allorché i dati siano pronti nei registri-addendi, viene inviato un segnale di VIA, a seguito del quale l'addizionatore è attivato; al termine del conteggio, il contatore, genera un segnale FINE, che riconduce in riposo la macchina. In sincronismo con la fine viene anche generato il segnale *strobe* per l'inoltro del risultato verso le apparecchiature utenti della somma.

Si noti che il primo clock attivo deve essere C_2, in quanto al termine del caricamento dei registri-addendi sono già disponibili all'uscita di BIT-ADD somme e riporto. Con l'ottavo C_2, è avviato verso il risultato l'ultimo bit, che è disponibile al successivo C_1: in sincronismo con questo va dunque generato lo strobe verso le apparecchiature utenti. L'ultimo C_1 è utile inoltre se i registri-addendi sono di tipo circolante; in questo modo, con l'ottavo C_1 il registro ripresenterà l'addendo nella sua forma iniziale.

Invece della generazione in loco delle due fasi di clock, si può anche usare, ove disponibile, un clock bifase generale. La rete di controllo si limita allora a due soli stati, *riposo* e *adder attivo*. Quest'ultimo, equivalente alla coppia Q1-Q2, abilita le due fasi generali di clock ad essere applicate alle macchine in esame; in tal caso, VIA e FINE sono sincroni con C_i in modo che la prima fase attivata sia C_2, e che lo strobe venga con l'ultimo C_1.

Descrizione del circuito (fig. 5.6)

Il circuito è realizzato con alcuni componenti commerciali della famiglia TTL 74xxx :
- gli shift-register 74165 "parallel-in serial-out" come registri-addendi,

- lo shift-register 74164 serial-in parallel-out" come registro-somma,
- il contatore 84160.

Approfittando del fatto che i 165 sono edge-triggered sul fronte di discesa mentre il 164 e il 160 su quello di salita, si è adoperato un unico segnale di clock, C, del quale il fronte di salita fa le funzioni di C_2, quello di discesa di C_1: il flip-flop D della rete ADD, coerentemente con le esigenze di tempificazione, è edge-triggered sul fronte di salita. La durata del clock alto deve essere dunque commisurata a t_s+t_{ADD} (cfr. fig. 5.4).

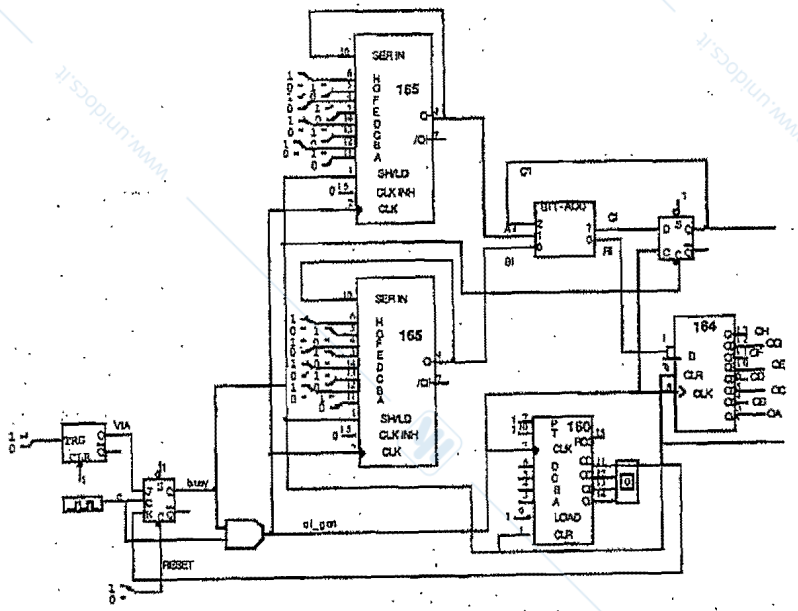


Fig. 5.6a): Addizionatore seriale: circuito

La parte combinatoria dell'addizionatore, BIT-ADD, è realizzata con un PLA che calcola somma e riporto:

$$R_i = A_iB_i + A_iC_i' + B_iC_i'$$
$$C_i = A_i \oplus B_i \oplus C_i'$$

In luogo del contatore mod-8, è stato adoperato quello commerciale, che è modulo-16. Dopo l'ultimo bit, allora, il contatore va nello stato 8 (e non

direttamente a 0), mentre torna a 0 con il reset. Ciò rende disponibile un segnale, C_8, che indica utilmente la fine del conteggio.

La rete CONTR è realizzata con il flip-flop *busy*, posto in set da VIA e in reset da C_8, sempre in sincronismo con un fronte di discesa (C_1): a tale scopo si adopera un flip-flop JK edge-triggered sul fronte di discesa del clock principale. Il segnale VIA, simulato con un monostabile, è un impulso lungo un po' di più del periodo del clock, in modo da contenere almeno un fronte di discesa di questo (nel circuito simulato, il periodo è 20 u.t., VIA dura 25 u.t.). Il segnale *busy* così generato assume le seguenti funzioni:

- genera il treno degli 8 clock *cl_gen*, ponendosi in AND con il clock c;
- funge da reset o clear per il flip-flop D, per il 164 e per il contatore (*C=SH/LD=CLR=busy=0*);
- abilita i 165 a funzionare da shift-register (*SH/LD=busy=1*);
- consente di effettuare il load del 165 (*SH/LD=busy=0*);
- funge da strobe per le macchine-utenti della somma (fronte 1→0);
- garantisce che le macchine-utenti una volta abilitate ricevono il primo impulso di clock di durata corretta (il segnale di busy è attivo sul fronte 1→0 per cui l'impulso successivo è il primo impulso utile).

Viene inoltre utilizzato un segnale di RESET per portare la rete CONTR nello stato iniziale di riposo. Tale ingresso è posto nel clear del flip-flop.

Ai fini della simulazione, i registri-addendi sono caricati al valore fissato da appositi switch. E' esemplificato il caso della addizione:

0101 0001 + 0101 1011 = 1010 1100
(81) + (91) = (172)

Tempificazione

In fig. 5.6b) è mostrata la tempificazione per l'addizione di cui sopra. Si noti:

- VIA che lancia l'operazione,
- busy che si alza sulla variazione 1→0 di c dentro VIA,
- cl_gen, treno di 8 clock estratti da c,
- leggendo ai fronti 0→1 di cl_gen si ritrovano i bit A_i, B_i a partire da quelli meno significativi: 11, 01, 00 ...
- anche i corrispondenti riporti entranti C'_i (0, 1, 1, ...) si leggono in corrispondenza dei fronti 0→1,
- la somma (0011...a partire dalla cifra meno significativa) è trasferita da R_i a QA ancora sul fronte 0→1 ed è stabile sul fronte 1→0,
- subito dopo un fronte 0→1, R_i assume valori impropri, in quanto corrispondenti al nuovo riporto e ai vecchi bit-addendi ma al successivo fronte 1→0 si presentano i nuovi addendi e al prossimo 0→1, R_i è corretto (si veda ad esempio il valore improprio 1 assunto fra il primo ed il secondo fronte 0→1).

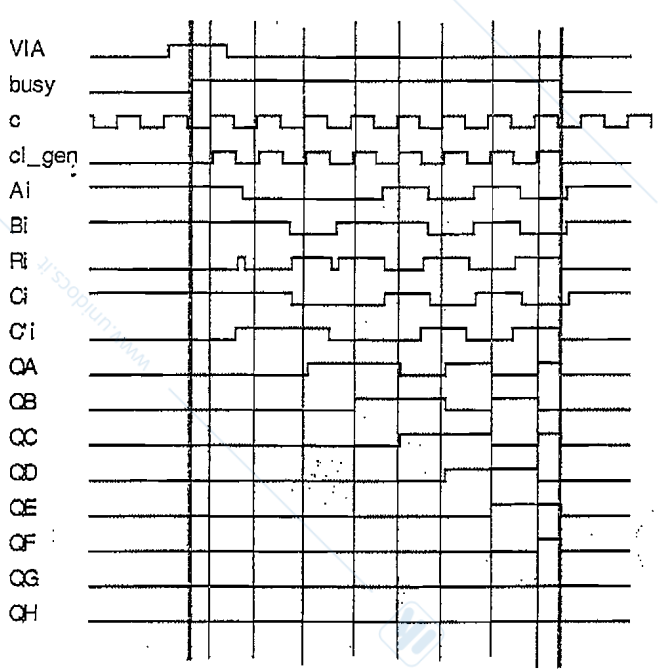


Fig. 5.6.b): Addizionatore seriale: tempificazione

6. Cronometro

Tipo di circuito: rete composta
Riferimenti: §§ I-5, V-8.1
Obiettivo: progettazione di sistemi

Testo

Progettare un cronometro digitale in grado di misurare e visualizzare unità, decimi e centesimi di secondo su di un display a sette segmenti, utilizzando un oscillatore operante alla frequenza di 1 kHz. Il cronometro sia dotato di due pulsanti, uno per avviarlo e fermarlo, l'altro per visualizzare, su un unico visualizzatore, il tempo cronometrato in uno o più istanti intermedi e allo "stop" del cronometro.

Specifiche di progetto

La macchina ha tre segnali di ingresso a livelli:
- *START/STOP*, per attivare o disattivare il cronometro;
- *PARTIAL*, per ottenere sul display il tempo parziale senza arrestare il cronometro;
- *CLEAR*, per riportare il cronometro nello stato iniziale (tempo 0);
- *cl-gen*, generato da un oscillatore.

In uscita la macchina deve fornire:
- le 4 cifre (unità, decine decimi e centesimi di secondo), ciascuna su di un display e quindi per ciascuna di esse i 7 segnali a livello necessari per i illuminare i visualizzatori (cfr.§ I-5);
- un segnale di "ripple", che nasce nel caso in cui il tempo trascorso sia di 1h, da utilizzare eventualmente per costruire un cronometro che preveda anche il conteggio delle ore.

Impostazione del progetto

La macchina consta essenzialmente di un contatore, di un registro per memorizzare il conteggio quando richiesto (segnali PARTIAL e STOP) e di un trascodificatore per alimentare il visualizzatore a 7 segmenti.

In primo luogo occorre derivare dal clock un impulso periodico che conti i centesimi di secondo, quindi un segnale a frequenza di 100 Hz; supponendo che l'oscillatore oscilli a frequenza di 1 kHz, occorre un contatore che divida la frequenza del clock per 10 (se la frequenza fosse di 10 kHz, occorrerebbe un divisore per 100 e così via). Se si considerano, ad esempio, oscillatori commerciali a cristallo di quarzo si ha che questi hanno frequenze di risonanza da alcuni chilohertz a diversi megahertz.

Il segnale a 100 Hz va poi in ingresso ad un contatore di tempo, che conta centesimi, decimi, unità e decine di secondi. Da questo contatore viene caricato il registro di visualizzazione, all'arrivo di uno dei due segnali PARTIAL o STOP. L'uscita del registro è trascodificata ed inviata ai visualizzatori a segmenti.

Il sistema è dunque complessivamente costituito da:
- macchina M_1: divisore di frequenza;
- macchina M_2: contatore di tempo;
- macchina M_3: registro di memorizzazione;
- macchina M_4: trascodificatori e visualizzatori.

Scelta di progetto

Per il progetto di M_1 e M_2 si utilizza come componente un contatore modulo-16 con le seguenti caratteristiche:
- impulso di conteggio CLK: il contatore opera sul suo fronte di salita;
- bit di ingresso a livelli C, D, B, A, in ordine dal più al meno significativo, che ne definiscono il valore da caricare;
- segnale di ingresso a livelli $C/\overline{L}$ che ne abilita il conteggio ($C/\overline{L}=1$) o il caricamento ($C/\overline{L}=0$), sempre sul fronte di salita di CLK;
- segnale a livelli di azzeramento CLEAR;
- variabili di stato del conteggio QA, QB, QC, QD in ordine dalla più alla meno significativa;
- segnale di uscita a livelli 0-attivo, $C0=\overline{QA\cdot QB\cdot QC\cdot QD}$, corrispondente al segnale *div* (cfr.§ V-8.1).

Progetto delle macchine componenti

Macchina M_1 (contatore divisore di frequenza)

La macchina riceve il segnale di conteggio dell'oscillatore *cl_gene* a 1 kHz e deve generare il segnale con il periodo di un centesimo di secondo *cl_cen*: è dunque un contatore modulo 10. Il contatore è ottenuto da un contatore modulo-16, con la tecnica del posizionamento iniziale e ciclico a 6 (cfr. § V-11), motivata dalla sua maggiore semplicità circuitale: il contatore conta 6, 7, ..., 15, 6, 7, ... Si ha dunque (cfr. fig.6.1, comunque descritta in dettaglio in seguito):

- il valore 6, fissato con A=0, B=1, C=1, D=0, deve essere precaricato all'inizio: al segnale START/STOP si assegna la funzione di precaricamento (START/STOP=0) oppure di abilitazione al conteggio (START/STOP=1);
- per C0=1 (il contatore è al massimo valore del conteggio, div=1) il contatore deve essere ancora di nuovo caricato a 6;
- in entrambi i casi, il segnale ha effetto con il primo fronte di salita di CLK e si ha dunque:

$$C/\overline{L}=C0\cdot START/STOP$$

- il segnale CLEAR non è adoperato (START/STOP ne svolge le funzioni): CLEAR=0;
- il contatore a valle (M_2) opera sul fronte di salita per cui il fronte di C0 (segnale 0-attivo) opera come segnale di ripple:

$$cl\_cen = C0\uparrow$$

Macchina M_2 (contatore del tempo)

La macchina, ricevendo in ingresso il segnale *cl-cen*, conta centesimi, decimi e secondi con una struttura in cascata (a propagazione, ripple) di 4 contatori:

- contatore mod-10 dei centesimi, con ingresso *cl_cen* ed in uscita i 4 bit del conteggio ed il ripple *cl_dec* per i decimi;
- contatore mod-10 dei decimi con ingresso *cl_dec* ed in uscita i 4 bit del conteggio ed il ripple *cl_sec* per le unità di secondo;
- contatore mod-10 delle unità, con ingresso *cl_sec* ed in uscita i 4 bit del conteggio ed il ripple *cl_10sec* per le decine di secondo;
- contatore mod-6 delle decine (il conteggio va fino a 59 secondi) con ingresso *cl_10sec* ed in uscita i 4 bit del conteggio ed il ripple *cl_h* per le ore da adoperare per una eventuale espansione del contatore.

Per i contatori di cui sopra è adoperata la tecnica del conteggio con azzeramento dopo l'ultimo stato (9 per quelli mod-10, 5 per quello mod-6), al fine di disporre alle uscite del codice numerico 8-4-2-1 (il contatore conta 0, 1,..., 9, 0, 1... oppure 0, 1, ..., 5, 0, 1...) e quindi di adoperare i trascodificatori standard verso i 7 segmenti, come in § I-5. Si ha dunque, considerando che le uscite corrispondenti ai primi tre stadi sono pari a 9 e quella dell'ultimo stadio è pari a 5:

$$cl\_dec = \overline{QD \cdot QA}\uparrow$$
$$cl\_sec = \overline{QD \cdot QA}\uparrow$$
$$cl\_10sec = \overline{QD \cdot QA}\uparrow$$
$$cl\_h = \overline{QC \cdot QA}\uparrow$$

I segnali sono 0-attivi e ottenuti da porte nand. Tale scelta è stata fatta per far sì che la variazione $0 \rightarrow 1$ dei segnali equivalga alla generazione del segnale di conteggio (ripple) per i contatori posti negli stadi a valle. I quattro contatori sono resettati dal segnale CLEAR applicato all'ingresso CLR; si potrebbe anche adoperare $\overline{START/STOP}$ ma si è preferita questa soluzione, che è comunque più economica in quanto non richiede l'invertitore.

Macchina M_3 (memorizzazione del tempo)

Occorre memorizzare 4 bit per ogni cifra (centesimi, decimi, unità, decine); questi sono raggruppati in due registri ad 8 bit, con caricamento sul fronte di salita di un impulso CLK. Poiché il caricamento deve avvenire sia con il segnale PARTIAL che con lo STOP, entrambi sono differenziati (attraverso un monostabile) e si ha:

$$CLK = store = \partial PARTIAL + \partial\overline{START/STOP}$$

I due registri sono azzerati con il segnale di CLEAR:

CLR=CLEAR

Macchina M_4 (trascodificatore)

Le uscite del registro vengono trascodificate mediante 4 trascodificatori dal codice 8-4-2-1 a quello a 7 segmenti (cfr. § I-5); i 7 segnali binari uscenti da ciascuno di essi sono applicati alle lampadine dei 4 visualizzatori.

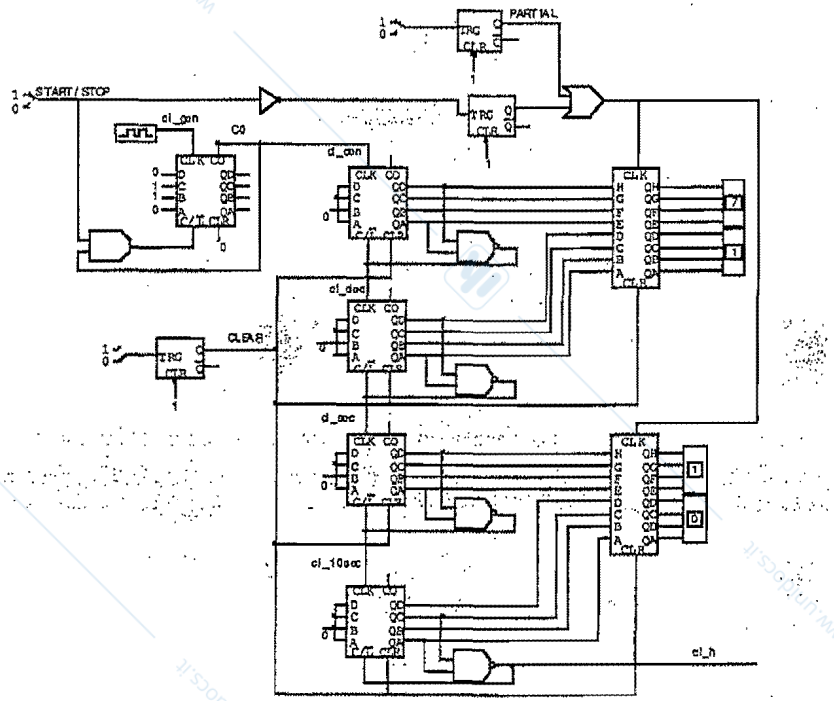


Fig. 6.1a: Cronometro

Descrizione del circuito (fig.6.1)

In figura è riportato il circuito con esclusione dei trascodificatori e dei visualizzatori; per essi si faccia riferimento al § I-5. Si noti in particolare:
- i segnali a livello START/STOP e PARTIAL sono differenziati da altrettanti monostabili, interessando di questi il fronte;
- il segnale CLEAR è anch'esso differenziato, pur dovendo agire come livello, al fine di evitare che sia lasciato inavvertitamente al valore attivo;
- il visualizzatore esadecimale sul contatore divisore è posto a puro scopo di controllo del funzionamento in fase di simulazione;
- anche i 4 visualizzatori sui registri sono posti a solo scopo di controllo della simulazione, in quanto i relativi ingressi devono invece accedere ai trascodificatori e ai visualizzatori a 7 segmenti.

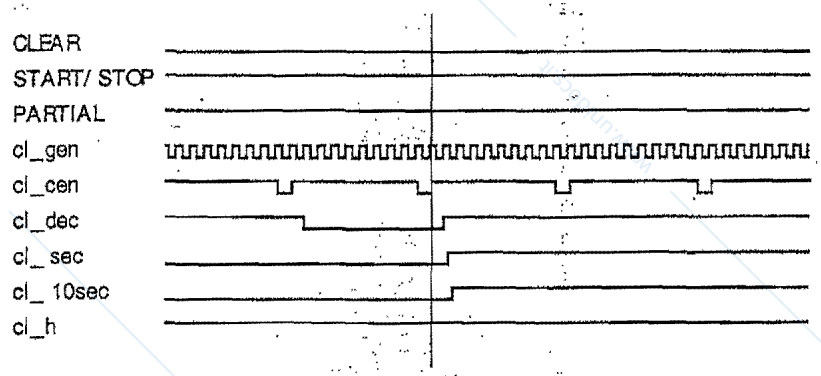


Fig. 6.1b: Diagramma di tempificazione

Tempificazione

In figura 6.1b è in particolare è riprodotto il passaggio da 09.99 a 10.00 sec (vedi linea verticale nel diagramma di tempificazione):
- all'arrivo di *cl-gen* scatta la cifra del divisore di frequenza, che muta da 9 a 0 (9→0), in quanto aveva precedentemente raggiunto il valore massimo ed era *cl-cen=0*;
- la variazione produce *cl-cen=0→1* per cui scatta anche la cifra dei centesimi, che muta 9→0 (in quanto era *cl-dec=0*);
- la variazione dei centesimi produce *cl-dec=0→1* per cui scatta anche la cifra dei decimi, che muta 9→0 (in quanto era *cl-sec=0*);
- la variazione dei decimi produce *cl-sec=0→1* per cui scatta anche la cifra delle unità, che muta 9→0 (in quanto era *cl-10sec=0*);

- la variazione delle unità produce *cl-10sec=0→1* per cui scatta anche la cifra delle decine.

Nel caso in cui la commutazione del cronometro produca il passaggio da 59.99 a 00.00 (il contatore complessivo è un contatore modulo-600) sono attivi tutti i *cl-x* ed allora scattano tutte le cifre l'una dopo l'altra e si genera il fronte di *cl-h*.

7. Ricevitore seriale

Tipo di circuito: rete composta
Riferimento: circuiti commerciali 74160, 74164, 74374
Obiettivo: progettazione di sistemi

Testo

Progettare un sistema per gestire la ricezione seriale e la successiva conversione serie-parallelo di messaggi composti da un numero variabile di byte. A tale scopo si assuma un protocollo di trasmissione per il quale il primo byte del messaggio è riconosciuto tale se preceduto da una sequenza di almeno tre byte, costituita da due o più byte *SYNC* seguiti da un byte *SOH* (*SYNC....SYNC-SOH*) e la terminazione del messaggio è identificata mediante un byte di fine (*EOT*). La macchina trasmette su un canale parallelo tutti i byte compresi fra SOH e EOT e ritorna poi in uno stato di riposo. Il byte di *START* è codificato con 80h, *SOH* con 04h ed *EOT* con 01h.

Definizione delle specifiche

L'ingresso primario della macchina è un bit seriale, rappresentato da un segnale a livelli, *bit*, che costituisce il valore del bit trasmesso ed un segnale impulsivo di sincronizzazione, cl_1, che ne determina i valori significativi. Ingresso ausiliario è un segnale, *RESET*, che pone il sistema nello stato iniziale.

L'uscita è il carattere parallelizzato su un byte, *dato*, ed un segnale di sincronismo ad esso associato, *p*, che è inviato verso le apparecchiature che devono ricevere il dato.

Impostazione del progetto

La macchina ha essenzialmente le seguenti funzionalità:
- riconoscimento dei byte significativi SYNC, SOH, EOT;

- individuazione del primo bit di una sequenza di byte di inizio (SYNC, ..., SYNC, SOH);
- riconoscimento delle sequenze di byte;
- conversione serie-parallelo dei byte.

La conversione serie-parallelo è effettuata, come di consueto, da uno shift-register (R), sincronizzato dal segnale cl_1 e da un associato contatore modulo-8 (CONT) che emette come ripple il segnale *p*.

Il riconoscimento dei byte significativi della sequenza avviene a mezzo di una rete combinatoria C che ha per ingresso gli otto bit parallelizzati (uscita parallela dello shift register) e per uscite binarie SYNC, SOH e EOT, associate ai rispettivi caratteri.

L'individuazione del bit ove inizia la sequenza, il riconoscimento della sequenza di byte e in generale i problemi di sincronizzazione sono affidati ad una apposita rete sequenziale M, caratterizzata dai seguenti segnali di ingresso:

- impulsi di sincronismo cl_2 (altra fase del clock) e *p* (ripple del contatore);
- segnali binari SYNC, SOH, EOT (provenienti dalla rete C), mutamente esclusivi;

e le seguenti uscite:

- reset del contatore CONT, *res*;
- segnale di abilitazione all'invio verso il ricevente del dato memorizzato nel registro, *strobe*.

Lo schema complessivo della macchina è quello di figura 7.1.

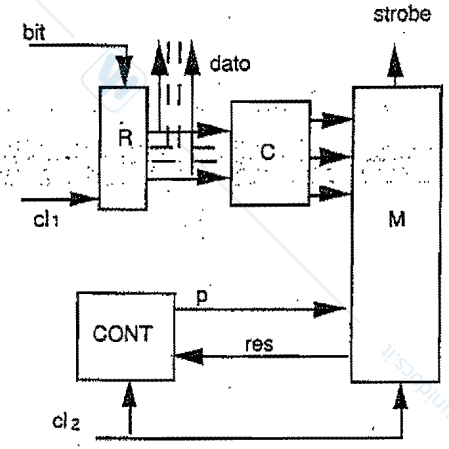


Fig.7.1: Schema generale del ricevitore

Il registro R e la macchina sequenziale M operano serialmente: M riceve in ingresso l'uscita della rete combinatoria C che opera sul dato parallelizzato da R. Le due macchine sono sincronizzate con due differenti segnali, cl_1 e cl_2, tra loro ritardati di un tempo *T*. Il segnale cl_1 determina gli istanti significativi in cui si acquisisce il bit trasmesso; mentre il segnale cl_2 è utilizzato per l'incremento del contatore CONT, che fornisce il segnale p quando il dato parallelizzato è in R, e per la sincronizzazione di M, che deve operare sul dato dopo che questo è stato acquisito da R ed elaborato da C. Per il corretto funzionamento del ricevitore, la somma del tempo di ritardo delle reti C e parte combinatoria di M deve essere inferiore al ritardo *T* fra le due fasi del clock, in modo da garantire che il segnale di posizionamento dei registri di M sia stabile all'arrivo del segnale cl_2.

La rete di controllo M (cfr. fig.7.2), è caratterizzata dai seguenti stati:

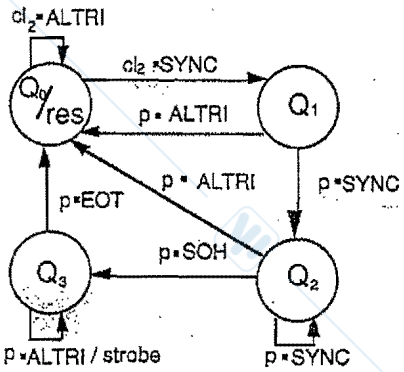


Fig. 7.2: Grafo di transizione degli stati di M

a) Q_0, stato di riposo: i bit in arrivo vengono via via immagazzinati dal registro a scorrimento e l'uscita *SYNC* della rete C viene analizzata ad ogni bit (impulso cl_2). In questo stato la macchina M ha l'uscita *res* sempre attiva bloccando il contatore al valore di conteggio zero. Quando si riconosce un byte *SYNC*, M transita allo stato Q_1: nella sequenza di bit si è riconosciuto un byte *SYNC* e, dunque, si ipotizza di aver così individuato il bit ove è iniziata la sequenza di byte. Il cambio di stato di M ha anche l'effetto di rendere l'uscita *res* non attiva per cui il contatore inizia la fase di conteggio.

b) Q_1, stato in cui è stato riconosciuto il primo *SYNC*: ai tempi dettati ora dal contatore (impulso p) si verifica se il secondo byte della sequenza sia *SYNC*; in caso affermativo si transita verso Q_2, altrimenti si torna a Q_0. In quest'ultimo caso il byte *SYNC* precedentemente ricevuto non appartiene alla sequenza di sincronizzazione del messaggio e si torna ad analizzare i bit clock per clock.

c) Q_2, stato in cui si è riconosciuta una sequenza di 2 o più SYNC: si esamina il nuovo byte, analogamente allo stato Q_1, e si ritorna da capo oppure si procede verso Q_3 se si è ricevuto il byte SOH.

d) Q_3, stato di ricezione del testo del messaggio; per ogni carattere ricevuto si invia lo strobe all'unità ricevente e quando viene riconosciuto il carattere EOT di fine testo si ritorna nello stato di riposo Q_0.

Progetto

Il progetto di massima di cui sopra viene ora discusso ed affinato. La rete di fig.7.2 è una rete autosincronizzata, ma di tipo particolare: solo nello stato Q_0 deve prendere in esame l'impulso cl_2, mentre per gli altri stati l'impulso di sincronizzazione è p, ripple del contatore ($p = div \cdot cl_2$): la rete può allora essere sincronizzata da un unico impulso (l'espressione sarà completata quando gli stati saranno stati codificati):

$$c = Q_0 \cdot cl_2 + \overline{Q_0} \cdot p \qquad (1)$$

Avendo così ridotto ad un sol impulso il segnale di sincronizzazione di M si ottengono tutti i vantaggi della rete a sincronizzazione esterna. La tabella di stato codificata è mostrata in figura 7.3.

		ingressi			
(stati)	S_2S_1	SYNC	SOH	EOT	Altri
(Q_0)	00	01	00	00	00
(Q_1)	01	10	00	00	00
(Q_2)	10	10	11	00	00
(Q_3)	11	11	11	00	11

Fig. 7.3: Tabella di stato

Avendo scelto flip-flop di tipo D per le variabili di stato, dalla tabella si ottengono:

- segnali di posizionamento:

$$d_2 = S_1 \cdot SYNC + S_2 \cdot SYNC + S_2 \cdot \overline{SOH} + S_1 \cdot S_2 \cdot \overline{EOT}$$

$$d_1 = \overline{S_2} \cdot \overline{S_1} \cdot SYNC + S_2 \cdot SOH + S_1 \cdot S_2 \cdot \overline{EOT}$$

- uscite:

$$res = \overline{S_1} \cdot \overline{S_2}$$

$$strobe = S_2 \cdot \overline{S_1} \cdot \overline{EOT} \cdot p \qquad (2)$$

e riprendendo il calcolo dell'impulso dalla (1):

$$c = \overline{S_1} \cdot \overline{S_2} \cdot cl_2 + S_1 \cdot p + S_2 \cdot p \qquad (3)$$

Sulla base dei codici assegnati a SYNC, SOH, EOT, i corrispondenti segnali binari sono:

$$SYNC = d_7 \cdot \overline{d_6} \cdot \overline{d_5} \cdot \overline{d_4} \cdot \overline{d_3} \cdot \overline{d_2} \cdot \overline{d_1} \cdot \overline{d_0}$$

$$SOH = \overline{d_7} \cdot \overline{d_6} \cdot \overline{d_5} \cdot \overline{d_4} \cdot \overline{d_3} \cdot d_2 \cdot \overline{d_1} \cdot \overline{d_0} \qquad (4)$$

$$EOT = \overline{d_7} \cdot \overline{d_6} \cdot \overline{d_5} \cdot \overline{d_4} \cdot \overline{d_3} \cdot \overline{d_2} \cdot \overline{d_1} \cdot d_0$$

Descrizione del circuito (figg. 7.4, 7.5, 7.6)

In figura 7.4 è riportato il registro R (circuito commerciale 74164) e la rete combinatoria C per la determinazione dei segnali SYNC, SOH e EOT secondo le (4); si noti la realizzazione delle nor a 8 ingressi con due or a 4 ed una nor.

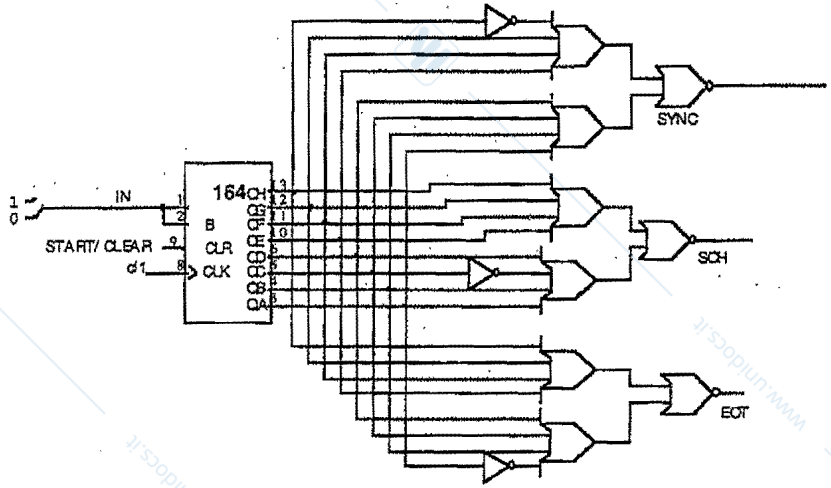


Fig. 7.4: Registro a scorrimento e rete combinatoria C

In figura 7.5 è riportato il contatore CONT (circuito commerciale 74160) e i circuiti ausiliari per determinare il reset in funzione degli stati di M ($res=\overline{S_1}\cdot\overline{S_2}$) ed il calcolo di c secondo le (3).

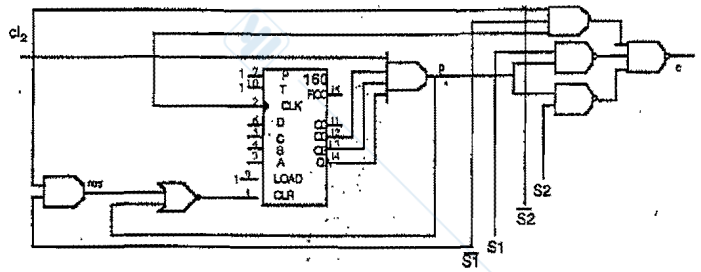


Fig. 7.5: Macchina CONT e circuiti ausiliari

La macchina complessiva è mostrata in figura 7.6, ove sono assemblati i circuiti di cui alle precedenti figure 7.4 e 7.5, la rete M ed un apposito apparato per la "bufferizzazione" del dato proveniente da R da inoltrare verso l'unità ricevente, cioè per la sua trasformazione in un dato di tipo tristate da collegare su un bus.

La bufferizzazione è realizzata con il circuito commerciale 74134; tale componente ha in ingresso, oltre al byte da bufferizzare, un segnale di abilitazione generale (*abil*) ed un segnale di caricamento del singolo dato *strobe* e mantiene neutra l'uscita sempre che è *abil=0*, caricando un nuovo dato con *strobe=1* e *abil=1*. L'apparato viene dunque abilitato all'atto del riconoscimento di *SOH* e rimane abilitato fino al riconoscimento del segnale di *EOT*, quindi nello stato Q_3 della rete M sempre che il carattere ricevuto non sia EOT:

$$abil = S_2 \cdot S_1 \cdot \overline{EOT}$$

L'uscita strobe delle (2) è quindi suddivisa in due componenti: una a livelli, *abil*, e l'altra impulsiva, *strobe*, che si semplifica in:

$$strobe=p$$

In tal modo il buffer acquisisce i dati, se in stato di abilitazione, al termine della conversione serie-parallelo grazie al segnale *p* proveniente da CONT.

La rete dispone di un segnale CLEAR che riposiziona M nello stato iniziale e rende pari a zero gli elementi del registro a scorrimento R. Si sottolinea che il CLEAR ponendo a zero gli elementi di R rende impossibile che in fase iniziale si possa rivelare un segnale di SYNC per presenza di caratteri dovuti a precedenti conversioni nel registro R.

Nelle figure 7.6b, c e d sono riportati i diagrammi di tempificazione per illustrare le differenti fasi del protocollo. La generazione dei segnali è ottenuta con dei monostabili per rendere agevole la verifica del comportamento del circuito.

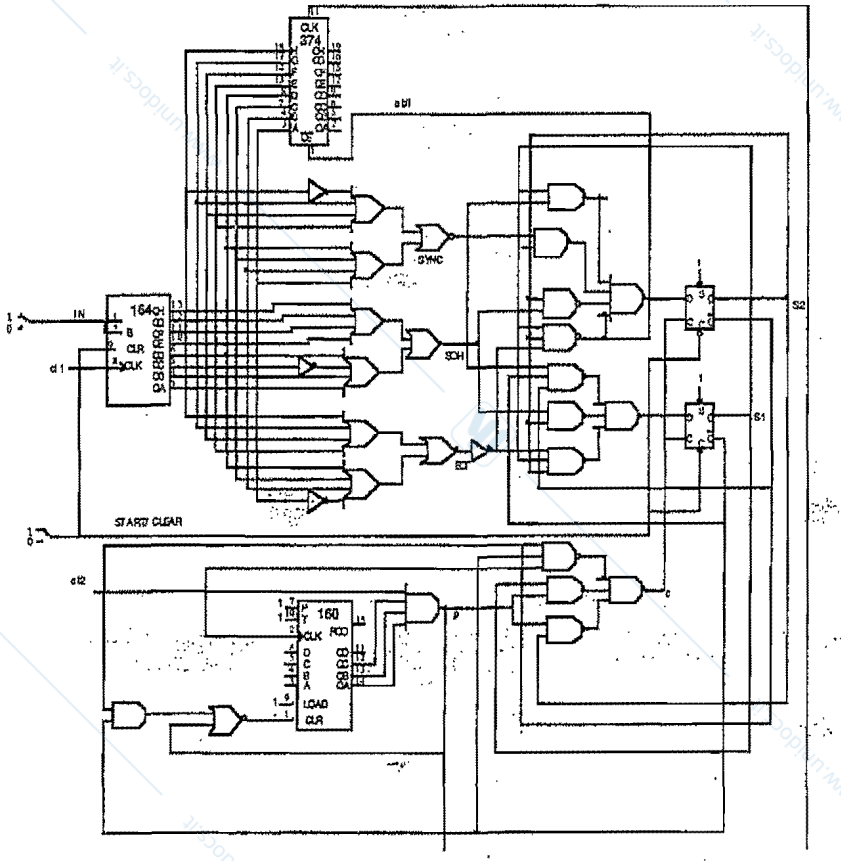


Fig. 7.6a: Circuito del riconoscitore

Tempificazione

Per il corretto funzionamento del ricevitore la somma del tempo di ritardo della rete combinatoria C (pari a 3 u.t.) e della rete combinatoria della macchina sequenziale M (pari a 2 u.t.) deve essere inferiore al ritardo del segnale cl_2 rispetto a cl_1 in modo tale da garantire che il segnale di posizionamento dei registri di M sia stabile all'arrivo del segnale cl_2.

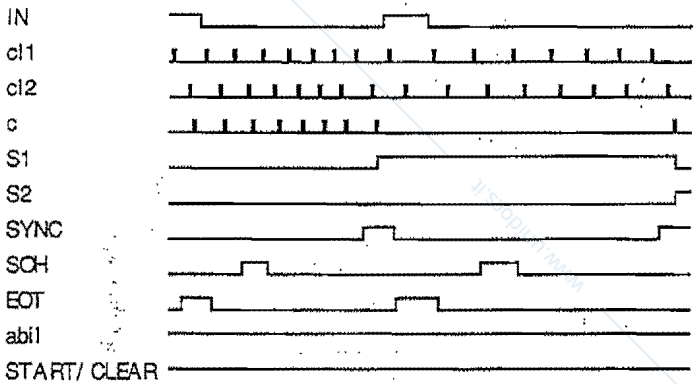


Fig. 7.6b: Diagramma di tempificazione per la ricezione SYNC

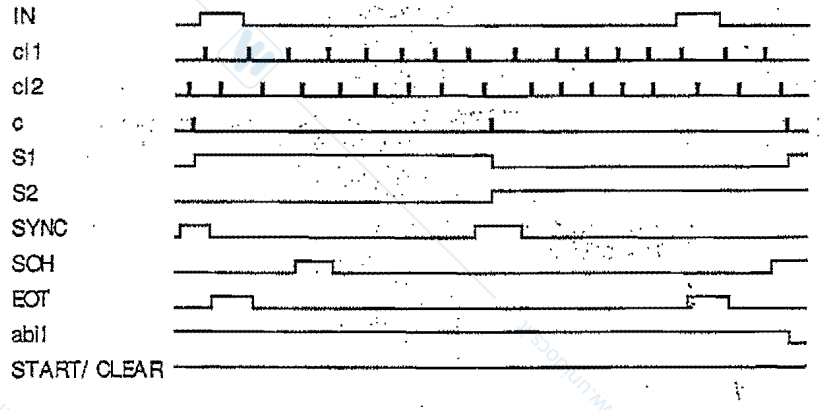


Fig. 7.6c: Diagramma di tempificazione per la ricezione SYNC-SYNC-SOH

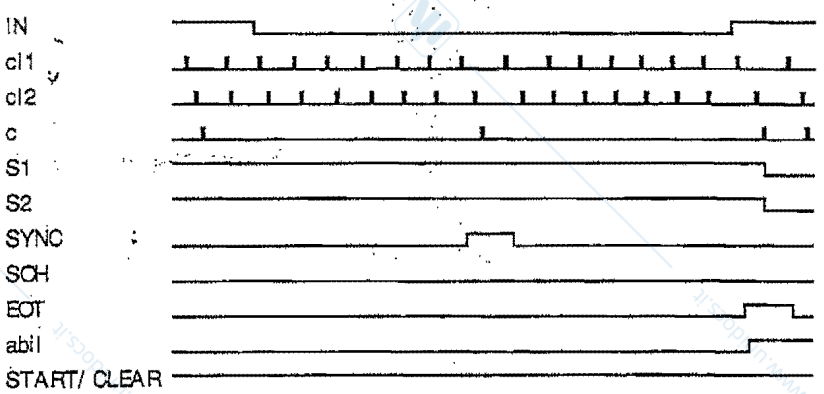


Fig. 7.6d: Diagramma di tempificazione per la ricezione EOT

8. Distributore automatico

Tipo di circuito: rete composta
Riferimento: circuito commerciale 74169
Obiettivo: progettazione di sistemi

Testo

Progettare la rete di controllo di un dispositivo che a seguito della ricezione di 150 lire fornisca la distribuzione automatica di un oggetto. A tale scopo si ritiene che il dispositivo possa ricevere in ingresso monete da 50 o 100 lire e che sia in grado di fornire eventuale resto di 200 lire, se nel sistema sono presenti monete da 50 ricevute da precedenti riscossioni o se non si è del tutto esaurita una riserva iniziale di tali monete.

Impostazione del progetto

Il dispositivo è composto di due reti: *conta-monete*, che riconosce l'avvenuta ricezione del pagamento previsto (150 o 200 lire) e *resto*, che gestisce l'eventuale fornitura del resto.

Appositi sensori riconoscono se la moneta introdotta sia 50 o 100 lire, discriminando i due eventi con:

- un segnale a livelli $100 / \overline{50}$ (con questo simbolo si intende la ricezione di 100 lire nel caso in cui sia $100 / \overline{50} = 1$ o di 50 lire per $100 / \overline{50} = 0$);

- un impulso *IMP* che rende attivo tale segnale.

La rete *conta-monete* riceve in ingresso tali segnali e ne fornisce in uscita i seguenti segnali (fig.8.1):
- *DIS:* abilitazione al distributore di oggetti:
- *RES:* richiesta di resto alla rete *resto*.

La rete *resto* tiene memoria del numero di monete da 50 contenute nel serbatoio; a tale scopo riceve anche essa in ingresso i segnali $100 / \overline{50}$ e *IMP* oltre alla richiesta di resto *RES* e fornisce in uscita:
- VUOTO: segnale di deposito vuoto, la macchina non può fornire resto (il segnale attiva anche un segnalatore per l'utente);
- DIS_RES: abilitazione alla fornitura del resto (si prevede il solo resto eventuale di 50 lire).

La rete *conta-monete* è posta in uno stato iniziale da un segnale *RESET*, mentre la rete *resto* è posta nello stato iniziate da un segnale *LOAD* che fissa il numero di monete inizialmente contenute nel serbatoio.

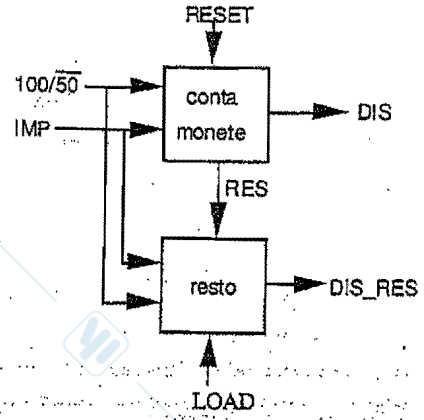


Fig.8.1: Schema generale del distributore

Progetto delle macchine componenti

Rete "conta-monete"

La rete è a sincronizzazione esterna, sincronizzata da IMP. Il grafo di stato è mostrato in figura 8.2a, ove si ha che:

- il segnale $100 / \overline{50}$ è indicato per semplicità 100 se $100 / \overline{50} = 1$ o con 50 se $100 / \overline{50} = 0$;
- le uscite DIS, RES (indicate nel grafo in questo ordine) sono supposte per il momento impulsive (si discuterà in seguito della loro sincronizzazione).

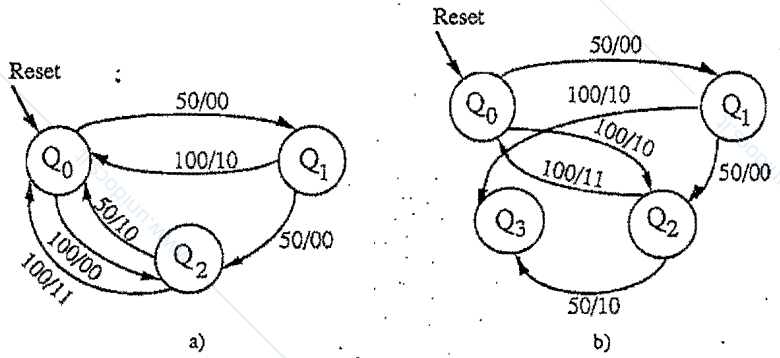


Fig. 8.2: Diagramma della rete "conta-monete"

La rete in realtà conta il numero di monete da 50 introdotte e, per il segnale di ingresso 50, è un contatore modulo-3. La rete può essere utilmente trasformata in un contatore modulo-4, di più semplice realizzazione, facendo in modo che il RESET venga comunque generato dopo ogni distribuzione, (fig.8.2b). Si tratta in particolare di un contatore ad incremento variabile: con 50 lire si incrementa di una unità, con 100 di 2 (in effetti conta il valore introdotto in unità da lire 50).

Questa tecnica si può facilmente generalizzare a sistemi in cui si debba verificare il raggiungimento di una certa soglia attraverso la somma di contributi parziali. Il progetto si può, ad esempio, generalizzare al caso che il costo del prodotto sia maggiore ed a quello che si possano introdurre monete di valore diverso: si può sempre considerare un contatore che conti in unità da 50 lire e che abbia incremento di x / 50 unità se la moneta introdotta vale x lire.

Dal grafo di fig. 8.2b si ricava la tabella di stato e quindi, dette y_1, y_0 nell'ordine le variabili di stato, si assume la codifica:

$$Q_0=(00); \qquad Q_1=(01); \qquad Q_2=(10); \qquad Q_3=(11);$$

e supposti di tipo T i flip-flop si ha (per semplicità di notazione si è posto $L = 100 / \overline{50}$):

$$t_0 = \overline{L} \qquad t_1 = L + y_0$$

e per le uscite:

$$DIS = (Q_2 + Q_1 \cdot L) \cdot IMP = (y_1 \cdot \overline{y_0} + \overline{y_1} \cdot y_0 \cdot L) \cdot IMP$$
$$RES = Q_2 \cdot L = y_1 \cdot \overline{y_0} \cdot L$$

Si noti che DIS è impulsivo e agisce sul dispositivo di distribuzione, mentre RES è a livelli, in quanto ingresso di un'altra rete (*resto*) anch'essa a sincronizzazione esterna sullo stesso impulso IMP e che pertanto prende in esame il valore di RES quando questo è stabile.

La rete "resto"

La rete *resto* è basata su un contatore che assume IMP come impulso di conteggio e conta le 50 lire nel serbatoio per il resto. Il contatore viene caricato inizialmente con il numero massimo di monete introdotte nel serbatoio e si comporta poi come di seguito schematizzato:

RES	L	contatore
0	0	entra 50 e non dà resto: si incrementa
0	1	entra 100 e non dà resto: resta fermo (è non abilitato)
1	0	l'ultima moneta è da 50, RES è certamente 0
1	1	entra 100 e dà resto: si decrementa

Si possono dunque trarre i segnali di controllo del contatore:

$$ABIL = \overline{L} + RES \qquad up / \overline{down} = \overline{RES}$$

Peraltro, il contatore deve bloccarsi a crescere se è stato raggiunto il massimo U_{max} del conteggio (corrispondente alla massima capacità del serbatoio) e a decrescere se il serbatoio è vuoto (il contatore è ad U_{min}) e si ha dunque:

$$ABIL = (\overline{L} + RES) \cdot (\overline{U_{max}} \cdot up + \overline{U_{min}} \cdot \overline{up}) = (\overline{L} + RES) \cdot (\overline{U_{max}} \cdot \overline{RES} + \overline{U_{min}} \cdot RES) =$$
$$= \overline{L} \cdot \overline{U_{max}} \cdot \overline{RES} + \overline{U_{min}} \cdot RES = \overline{L} \cdot \overline{U_{max}} + \overline{U_{min}} \cdot RES$$

L'ultima semplificazione riportata nella formula si ricava dalla tabella per la quale è impossibile che ci siano in ingresso 50 lire e si debba fornire resto ($\overline{L} \cdot RES$ è don't care).

L'operazione di caricamento iniziale del contatore è ottenuto con un'operazione di precaricamento abilitate dal segnale di LOAD.

Le uscite VUOTO e DIS-RES sono date da:

$$VUOTO = U_{min} \qquad DIS\_RES = RES \cdot \overline{U_{max}} \cdot IMP$$

Descrizione del circuito (fig. 8.3)

Si è considerata una riserva massima di 16 monete per cui è stato utilizzato un contatore modulo 16 del tipo 74169 per la realizzazione della macchina *resto*. Il contatore ha in ingresso i seguenti segnale:
- P (0-attivo): abilitazione generale del contatore;
- T (0-attivo): abilitazione all'uscita di conteggio massima ($RCO = T \cdot div$);
- CLK: conteggio;
- LOAD: caricamento di configurazione iniziale. Per LOAD=0 il contatore è caricato sul fronte di salita di CLK (per LOAD=1 il contatore invece conta sul fronte di CLK);
- u/d: abilitazione al conteggio a crescere o decrescere;

e in uscita fornisce:
- le uscite di conteggio QD, QC, QB e QA;
- RCO (0-attiva) pari a U_{max} per il conteggio a crescere e a U_{min} per il conteggio a decrescere.

Il contatore può essere pilotato:
- negando il segnale di *ABIL* e ponendolo in P per tener conto del fatto che il segnale di abilitazione è 0-attivo;
- ponendo in u/d il segnale di up/down.

La rete dispone di due differenti segnali di inizializzazione:
- LOAD, generato all'atto della configurazione iniziale del sistema, per rendere il valore del conteggio del contatore della macchina *resto* pari al numero di monete contenute nel serbatoio;
- RESET, generato per ogni richiesta di distribuzione di un oggetto, per portare la macchina *conta monete* D nello stato iniziale 00.

In fase di caricamento iniziale, oltre al segnale a livelli LOAD viene generato un impulso LOAD_RIS che agisce su CLK secondo le specifiche del contatore. LOAD-RIS è generato attraverso un monostabile che ritarda e differenzia il fronte di LOAD.

Il segnale RESET è generato da un'altra sezione (non progettata) della macchina, all'inizio del procedimento di immissione delle monete (ad esem-

pio, a seguito del fatto che l'utente selezioni l'oggetto da distribuire) ed è simulato con un switch e un monostabile differenziatore.

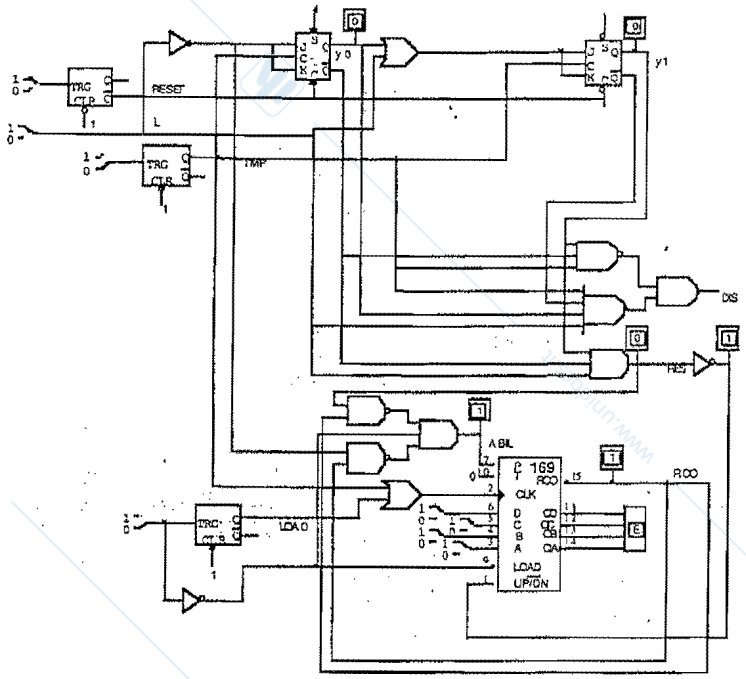


Fig. 8.3a: Distributore

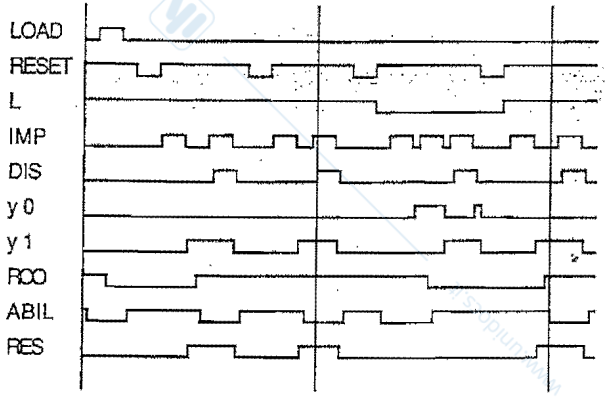


Fig. 8.3b: Distributore: tempificazione

9. Convertitore di caratteri

Tipo di circuito: rete composta
Riferimento: RL, VI-12
Obiettivo: progetto di sistemi; trasformazione da programma a circuito

Testo

Progettare una macchina sequenziale che implementi la struttura di controllo dell'algoritmo, riportato in figura 9.1a, per la conversione di stringhe S di n caratteri (con s_i generico elemento della stinga) in un numero decimale (P). Si supponga che i caratteri ricevuti possano essere o una cifra oppure il carattere "." (per individuare la componente decimale) e che la stinga contenga al più un solo carattere punto.

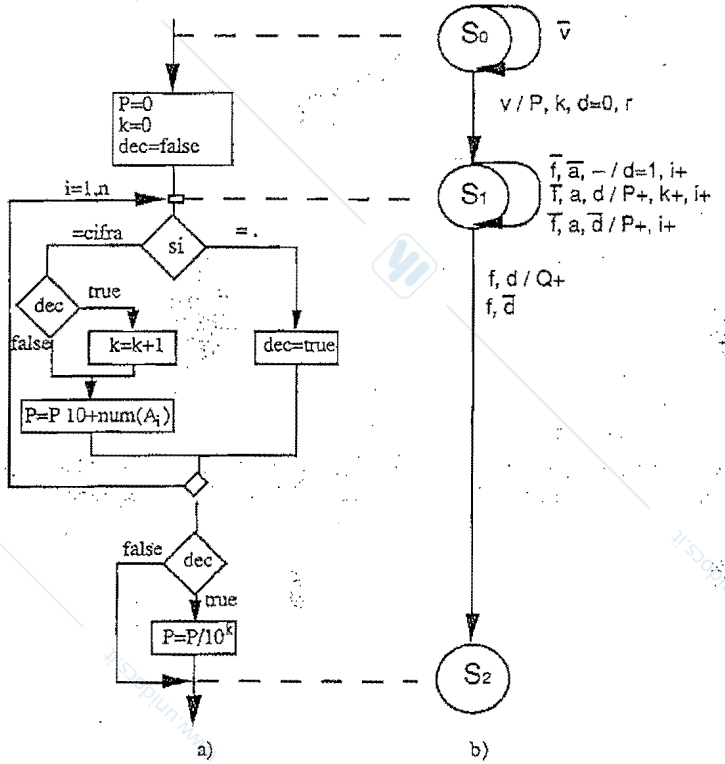


Fig. 9.1: Convertitore di caratteri a): Algoritmo; b): Grafo di stato

Analisi delle specifiche

Per l'implementazione del ciclo "*for*" si suppone di adoperare un contatore C che conta dunque il numero di passi dell'algoritmo, pari in ogni caso ad n, ed il numero di elementi della stringa S. Il contatore è asservito alla macchina da progettare, A, che provvede sia ad inviare il segnale di incremento al conteggio (*i+*) che il segnale di reset (*r*) all'atto della fase di iniziale.

La macchina A riceve in ingresso:
- *v*: segnale di via per l'attivazione della fase di calcolo;
- *a*: segnale indicante che il carattere s_i è una cifra (a) o un punto ($\bar{a}$);
- *d* (*dec*): segnale che indica se si stia analizzando la parte intera (d) o la parte frazionaria del numero ($\bar{d}$).
- *f*: segnale di fine conteggio (*div*) ricevuto dal contatore per individuare la terminazione del ciclo for

e fornisce in uscita: i segnali:
- *r*: per il reset del contatore;
- *k*, *P*: per inizializzare P=0 e k=0;
- *d=0*: per porre *d=false*;
- *d=1*: per porre *d=true*;
- *k+*, *i+*: per incrementare i conteggi di k ed i;
- *P+*: per il calcolo $P = P \cdot 10 + num(s_i)$;
- *Q+*: per il calcolo $P = P / 10^k$.

Progetto

Il diagramma di flusso può essere direttamente trasformato nel diagramma di stato della macchina sequenziale sincrona, assumendo come:
- stati i nodi del grafo di flusso;
- ingressi quelli definiti alla lista di cui sopra;
- uscite ancora quelle definite alla lista precedente, intendendo queste come segnali di abilitazione all'esecuzione delle operazioni da parte di apposite apparecchiature.

Da ogni nodo escono tanti archi quanti sono i percorsi del grafo di flusso e su ciascun arco gli ingressi sono caratterizzati dalle condizioni logiche che determinano il percorso e le uscite delle operazioni da effettuare lungo il percorso.

Le diverse operazioni da eseguire su ogni percorso sono nel caso specifico indipendenti, per cui possono essere abilitate simultaneamente. Nell'ipotesi che tutte le operazioni avvengano in un tempo massimo *T*, la rete può essere sincronizzata da un clock di periodo *T* ed il grafo di flusso si trasforma nel diagramma di stato di una macchina sincrona a sincronizzazione esterna di figura 9.1b, sincronizzata da un segnale di tempificazione periodico *CP*. Se la macchina è dedicata unicamente all'operazione di conversione lo stato S_3 risulta equivalente allo stato S_0.

Ne deriva, dunque, la tabella di transizione degli stati di figura 9.2.

		vf 00	01	11	10
stato	S_0	S_0	S_1	S_1	S_0
	S_1	S_1	S_1	S_0	S_0

Fig. 9.2: Tabella di stato della rete di controllo

Codificando l'unica variabile di stato con un flip-flop JK di nome A usato come set-reset sincronizzato si ha:

$$J_A = v \qquad K_A = f$$

La variabile *d* può essere realizzata con un flip-flop di nome D che viene posizionato in accordo con le uscite *d=0* e *d=1* e letto come d o $\bar{d}$:

$$J_D = A \cdot \bar{a} \cdot \bar{f} \qquad K_D = \bar{A} \cdot v = \bar{A}$$

L'ultima semplificazione riportata nella formula deriva dal fatto che K_D può essere utilmente posto alto anche per $\bar{A} \cdot \bar{v}$. Il flip-flop A assume il significato logico di macchina occupata per l'esecuzione dell'algoritmo di conversione.

Per quanto attiene alle altre uscite si ha:

$r = P = k = \bar{A}$ (analogamente a K_D);

$P^+ = A \cdot \bar{f} \cdot a \qquad k^+ = A \cdot \bar{f} \cdot a \cdot D \qquad i^+ = A$

$Q^+ = A \cdot f$

Si noti che la struttura di controllo è composta dai due flip-flop D ed A che costituiscono due reti composte in serie in quanto il segnali di posizionamento di D sono funzioni di A, ma non viceversa e le uscite sono funzioni degli stati di entrambi.

Progetto per decomposizione

Allo stesso risultato si può pervenire se si considera che la rete di controllo complessiva effettui la discriminazione tra parte intera e frazionaria mediante un proprio stato interno e non attraverso la variabile di ingresso ed uscita (*d*). Ne deriva il diagramma complessivo della rete di figura 9.3, nel quale lo stato S_1 identifica le cifre della parte intera, mentre S_2 quelle della parte frazionaria . La transizione $S_1 \rightarrow S_2$ avviene per l'avvenuta ricezione del carattere punto. Applicando la decomposizione funzionale si vede che la macchina associata alla partizione delle colonne è realizzata mediante il flip-flop D e quella sulle righe mediante il flip-flop A. L'equazioni di posizionamento dei flip-flop e delle uscite coincidono con quelle di cui sopra.

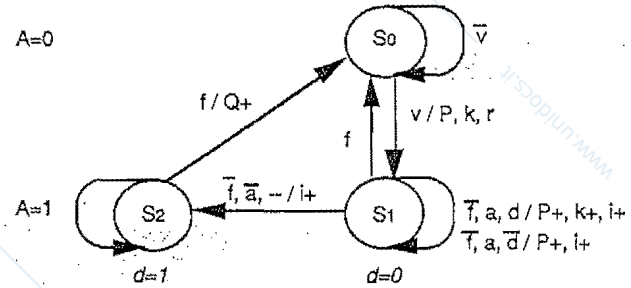


Fig. 9.3: Grafo di stato complessivo della struttura di controllo dell'algoritmo di conversione

Descrizione del circuito (fig. 9.4)

In figura è riportato il circuito per la conversione di una stringa S di 15 caratteri (n=15). I segnali a livelli *v* ed *a* in ingresso alla rete sono stati simulati mediante un switch e il segnale *f* a livelli coincide con l'uscita *div* (0-attiva) di un contatore modulo 16. Il segnale di abilitazione *cp* è stato simulato con un monostabile per rendere più agevole l'analisi della tempificazione del circuito.

Il dispositivo si può adoperare anche per la conversioni di stringhe di *l* elementi con *l*<*15*; ma in tal caso il contatore, invece di essere resettato in fase iniziale, va opportunamente precaricato con un valore pari a (*16-l-1*) per ottenere un contatore in modulo diminuito (cfr. V-10.2). Nel caso di una stringa di lunghezza 16 è necessario convertire la variazione di livello 1→0 del segnale di *div* (segnale di *ripple*), che individua l'avvenuto conteggio di 16 impulsi, mediante un flip-flop operante a variazione di fronte (ad esempio T edge-triggered sul fronte di discesa) per ottenere un segnale di conteggio

massimo *f* a livelli. Tale tecnica equivale ad aumentare il modulo del contatore.

La macchina dispone di un segnale di reset (*RES*) che posiziona il flip-flop A nello stato S_0 (A=0). Dal diagramma temporale si nota che al termine della conversione la macchina ritorna in S_0, riportando il contatore allo stato di conteggio 0. I segnali k+ e P+ sono a livelli per cui l'uscita è significativa in presenza del clock, mente il segnale Q+ è attivo sulla variazione 0→1.

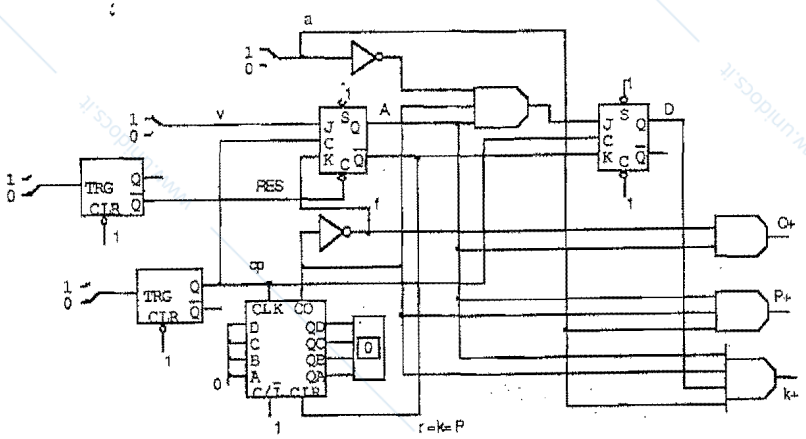


Fig. 9.4a: Convertitore di caratteri

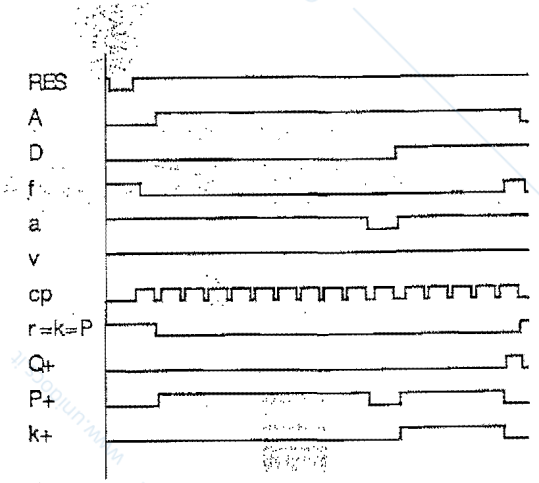


Fig. 9.4b: Convertitore di caratteri: tempificazione

10. Riconoscitore di sequenze con registri a scorrimento

Tipo di circuito: rete sequenziale sincrona
Riferimento: RL, X-6; circuito commerciale 74164
Obiettivo: uso di registri a scorrimento

Testo

Una rete sequenziale a sincronizzazione esterna (ove i tempi sono scanditi dal segnale di sincronismc) nella quale le uscite al tempo t dipendano soltanto dagli ingressi ai tempi $t\text{-}1$, $t\text{-}2$, ..., $t\text{-}i$ è sempre realizzabile con un registro a scorrimento, che memorizza gli ingressi e contiene i celle (ciascuna cella ha in generale k stati se k sono i valori possibili degli stati di ingresso), e da una rete combinatoria che li elabora. Un riconoscitore di sequenza binaria di n bit è un caso particolare di quanto sopra: per realizzarlo occorre un registro a scorrimento di n bit.

Progettare un riconoscitore delle 4 sequenze di 12 bit: 100000-01-10.

Scelta di progetto

Si sceglie, dunque, un registro a scorrimento di 12 bit, le cui celle costituenti saranno dette $Q_0, ..., Q_{11}$ e si progetta una rete combinatoria che calcoli le quattro uscite associate ciascuna al riconoscimento di una sequenza:

$$u_0 = Q_0 \cdot \overline{Q_1} \cdot \overline{Q_2} \cdot \overline{Q_3} \cdot \overline{Q_4} \cdot \overline{Q_5} \cdot \overline{Q_6} \cdot \overline{Q_7} \cdot Q_8 \cdot \overline{Q_9} \cdot Q_{10} \cdot \overline{Q_{11}}$$
$$u_1 = Q_0 \cdot \overline{Q_1} \cdot \overline{Q_2} \cdot \overline{Q_3} \cdot \overline{Q_4} \cdot \overline{Q_5} \cdot \overline{Q_6} \cdot \overline{Q_7} \cdot Q_8 \cdot Q_9 \cdot Q_{10} \cdot \overline{Q_{11}}$$
$$u_2 = Q_0 \cdot \overline{Q_1} \cdot \overline{Q_2} \cdot \overline{Q_3} \cdot \overline{Q_4} \cdot \overline{Q_5} \cdot Q_6 \cdot \overline{Q_7} \cdot Q_8 \cdot \overline{Q_9} \cdot Q_{10} \cdot \overline{Q_{11}}$$
$$u_3 = Q_0 \cdot \overline{Q_1} \cdot \overline{Q_2} \cdot \overline{Q_3} \cdot \overline{Q_4} \cdot \overline{Q_5} \cdot Q_6 \cdot \overline{Q_7} \cdot Q_8 \cdot Q_9 \cdot Q_{10} \cdot \overline{Q_{11}}$$

Le formule identificano le quattro sequenze generate dalla clausola con 10 variabili specificate ($Q_0 \cdot \overline{Q_1} \cdot \overline{Q_2} \cdot \overline{Q_3} \cdot \overline{Q_4} \cdot \overline{Q_5} \cdot \overline{Q_7} \cdot Q_8 \cdot Q_{10} \cdot \overline{Q_{11}}$) e due no, indicate con il trattino nel testo dell'esercizio.

Rispetto alla soluzione di progetto ex-novo del riconoscitore, questa presenterà complessivamente un maggior numero di porte elementari, ma è spesso più conveniente economicamente per la disponibilità sul mercato dei componenti impiegati nel progetto.

Se si suppone che, dopo il riconoscimento di una sequenza, una nuova sequenza sia costituita da altri 12 bit nessuno dei quali coincidente con quelli della sequenza riconosciuta (le due sequenze non sono parzialmente sovrapposte), allora all'atto del riconoscimento è necessario resettare il registro a scorrimento. Tale operazione permette di riportare il riconoscitore nello stato iniziale ed è necessaria per evitare che venga riconosciuta valida una sequenza di bit composta in parte da bit di una sequenza vecchia ed in parte da bit di una nuova: ad esempio, la sequenza di 22 bit 100000-01-100000-01-10 conterrebbe due volte una delle sequenze date, (con il reset dopo i primi 12 la seconda sequenza non è riconosciuta).

Il segnale di clear del registro a scorrimento può dunque essere attivato da un segnale di reset esterno (RESET) o da un segnale di avvenuto riconoscimento di una stringa (RIC_SEQ) per cui si ha:

$$RIC\_SEQ = u_3 + u_2 + u_1 + u_0$$
$$CLEAR = RESET + RIC\_SEQ$$

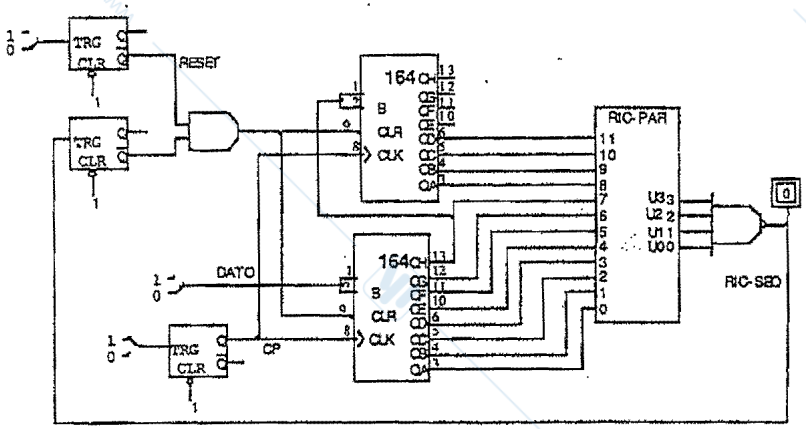


Fig. 10.1a: Riconoscitore di sequenza con registri a scorrimento

Descrizione del circuito (fig. 10.1)

Non disponendo di componenti commerciali a 12 bit, si utilizzano 2 registri a scorrimento ad 8 bit dei quali si decodificano soltanto i primi 12. Poiché il problema richiede un ingresso seriale dei dati ed una uscita parallela, si usano due registri 74164, connettendo il serial-out del primo con il serial-in del secondo. L'abilitazione e il segnale di reset sono forniti in parallelo ai due registri.

Si noti che il collegamento fra i due registri non fa altro che realizzare un unico shift-register di lunghezza doppia: nonostante le apparenze, non si tratta di due macchine in serie, ma di un'unica macchina avente come stato il

prodotto cartesiano degli stati delle due macchine; il collegamento fra le due è parallelo.

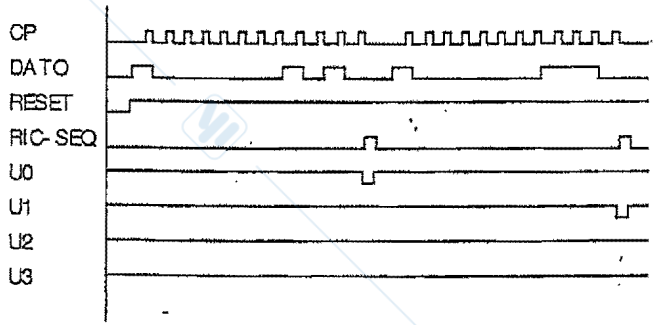


Fig. 10.1b: Riconoscitore di sequenza con registri a scorrimento

La rete combinatoria per le funzioni u_0, u_1, u_2 e u_3 è implementata mediante un PLA con uscita 0-attiva. e si ha dunque:

$$RIC\_SEQ = \overline{u_3 \cdot u_2 \cdot u_1 \cdot u_0}$$

Il reset CLR del registro a scorrimento è asincrono (a livelli) e 0-attivo; esso quindi non si può direttamente ottenere dal segnale RIC-SEQ, poiché il segnale in ingresso deve essere limitato nel tempo in modo da resettare il registro, ma da consentire la successiva memorizzazione di nuovi dati. Si pone dunque:

$$CLR = \overline{\partial RIC\_SEQ} \cdot \overline{\partial switch}$$

ove i due segnali di clear sono impulsi 0-attivi (ottenuti dalle uscite false dei monostabili) e generati il primo sul fronte di salita di RIC_SEQ, il secondo dall'azionamento di un switch.

Tempificazione

Si fa presente che per garantire il corretto funzionamento del circuito è necessario che l'intervallo di tempo che intercorre tra due successivi impulsi di abilitazione allo scorrimento (*cp*) sia maggiore della durata dell'impulso di reset (*CLR*). Se così non fosse si avrebbe la possibilità di perdere un carattere poiché se l'uscita di reset è attiva non si ha l'acquisizione di un nuovo carattere.