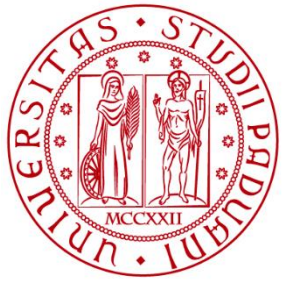
DEI
DIPARTIMENTO DI
INGEGNERIA DELL'INFORMAZIONE



UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Sistemi Digitali

Sintesi dei Circuiti Sequenziali

Marta Bagatin, marta.bagatin@unipd.it

Corso di Laurea in Ingegneria dell'Informazione

Anno accademico 2022-2023

Scopo della lezione

- Identificare i **passi per la progettazione** (= sintesi) **di un circuito sequenziale sincrono**
- Vedere un esempio di progettazione di circuito sequenziale (riconoscitore di sequenza)

Procedura per la sintesi di circuiti sequenziali

1) Identificare/ formulare le **specifiche di progetto** del circuito
2) Ottenere un **diagramma di transizione degli stati** a partire dalle specifiche
3) Determinare la **tabella degli stati** a partire dal diagramma degli stati (identificare possibili stati equivalenti, minimizzare numero di stati)
4) Assegnare una **codifica binaria** a ciascuno stato
5) Selezionare il tipo di flip-flop da usare (es.D positive-edge-triggered)
6) Derivare le **equazioni degli ingressi dei flip-flop** in base agli stati futuri nella tabella delle transizioni di stato
7) Derivare le **equazioni delle uscite del circuito**
8) **Minimizzare** le equazioni degli ingressi dei flip-flop e delle uscite
9) Disegnare il circuito e verificarne la correttezza con un **diagramma temporale** e le opportune transizioni di stato

Analogo a circuiti combinatori!

Trovare il diagramma di transizione degli stati

Formulazione del diagramma degli stati

- Trovare il **diagramma delle transizioni di stato è la parte più difficile (ma anche creativa!)** nel processo di sintesi di un circuito sequenziale
  - Per tradurre correttamente una descrizione a parole delle specifiche del circuito in un diagramma degli stati è necessario un passo di astrazione
  - Non esiste un algoritmo, occorre pratica e attenzione (e inventiva)
- Idea intuitiva del concetto di **stato**:
  - riassume tutto quello che è successo al sistema nel passato che è rilevante per determinare il comportamento futuro
  - dipende dalla storia degli ingressi forniti al sistema
  - serve al sistema per **ricordarsi della sua storia**

Stato come astrazione

- Nel momento in cui si sta formulando il diagramma di transizione di stato può essere molto utile **scrivere in modo astratto la descrizione di ciascuno stato**, cioè **descrivere l'astrazione rappresentata dallo stato**

Esempi: Stato come astrazione della sequenza ricevuta in ingresso

- Es. 1: Un sistema si trova nello stato S_1 se ha ricevuto al bit di ingresso X il valore ‘1’ per tre volte consecutive (cioè per 3 fronti di salita del clock consecutivi)
  - Si trova in S_1: se ha ricevuto al bit X in ingresso ‘00111’ o ‘10101111’
  - Non si trova in S_1 se ha ricevuto ‘00011’ o ‘011100’

- Es. 2: Un sistema si trova nello stato S_2 se ha ricevuto agli ingressi X_1 X_2, nell’ordine, le combinazioni 00, 01, 11, 10, con un numero qualsiasi di ripetizioni consecutive di ciascuna combinazione e con ’10’ come ultima combinazione
  - Si trova in S_2: se ha ricevuto: 00,00,01,01,01,11,10,10 o 00,01,11,11,11,10
  - Non si trova in S_2 se ha ricevuto: 00,11,10,10 o 00,00,01,01,11,11

Quanti stati utilizzare?

- Nella formulazione del diagramma degli stati, si aggiungono nuovi stati, mano a mano che si esaminano nuove transizioni
- Potenzialmente, ogni transizione può portare a un nuovo stato, ma è importate **riutilizzare gli stati esistenti, se possibile**, per limitare più possibile il numero di stati del sistema
  - Es. 1 - Descrizione astratta dello stato S_1: il sistema ha ricevuto al bit di ingresso X il valore '1' per le ultime 3 volte consecutive. Supponiamo che il sistema sia arrivato in S_1 ricevendo la sequenza 00111 e riceva '1' in ingresso (=> 001111). Possiamo riutilizzare S_1 come stato futuro? Sì!
- Il nostro scopo è realizzare il sistema con il **minimo numero di stati possibile**: quanto maggiore è la comprensione del significato degli stati in astratto, tanto più facile sarà **individuare il numero corretto di stati per rappresentare il sistema**

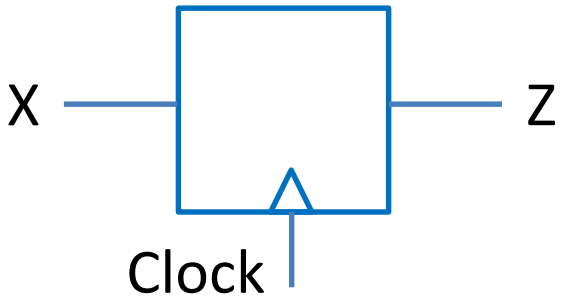
Reset

- Diversamente dai sistemi combinatori, **all'accensione un sistema sequenziale si trova in uno stato indefinito** (non è noto lo stato in cui si trovano i flip-flop)
- Un circuito deve trovarsi in uno **stato iniziale noto**, prima di poter operare correttamente e fornire in uscita valori significativi. Il circuito arriva in questo stato iniziale tramite un **segnale di reset**
  - Di solito è automaticamente applicato all'accensione del circuito
  - Può anche essere previsto un meccanismo per attivare il reset, nel caso il sistema si trovi in uno stato di errore, sconosciuto o indefinito

Esempio 4.3: Riconoscitore di sequenza

- Un riconoscitore di sequenza è un circuito in grado di rilevare se una certa sequenza si è presentata in ingresso, indipendentemente dagli ingressi che si sono succeduti prima di tale sequenza

- **Ingresso (1 bit): X, Uscita (1 bit): Z, riconosce la sequenza 1101**
  - Il circuito è inizializzato a 0 (reset all'accensione)
  - Z = 1 se si è presentata la sequenza 110 e X = 1
  - Z = 0 altrimenti

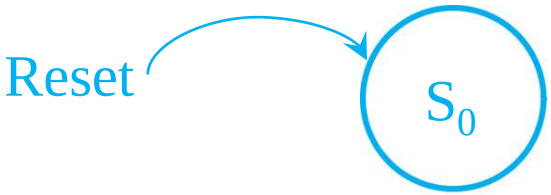


Esempio 4.3: Riconoscitore di sequenza

- **Ingresso: X, Uscita: Z, riconosce la sequenza 1101**
  - Il circuito è inizializzato a 0 (reset all'accensione)
  - Z = 1 quando si è presentata la sequenza 110 e X = 1
  - Z = 0 altrimenti
- **Primo passo: Mealy o Moore**? Osserviamo dalla formulazione del problema che l'uscita non dipende solo dallo stato presente, ma anche dall'ingresso: serve un modello di tipo Mealy

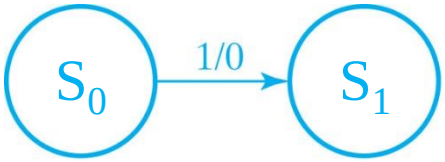
Esempio 4.3: Riconoscitore di sequenza

- **Ingresso: X, Uscita: Z, riconosce la sequenza 1101**
  - Il circuito è inizializzato a 0 (reset all'accensione)
  - Z = 1 quando si è presentata la sequenza 110 e X = 1, Z = 0 altrimenti
- Ora procediamo a **identificare gli stati**, tenendo in mente che ogni stato deve essere in grado di ricordare la storia degli ingressi (o meglio, la parte della storia che ci interessa)
- **Partiamo da uno stato arbitrario S_0, dal quale si arriva dopo il segnale di reset**: definiamo S_0 in modo astratto con **«nessun simbolo della sequenza è stato riconosciuto in ingresso»**

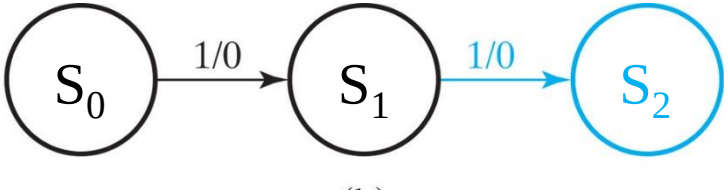


Esempio 4.3: Riconoscitore di sequenza

- **Ingresso: X, Uscita: Z, riconosce la sequenza 1101**
  - Il circuito è inizializzato a 0 (reset all'accensione)
  - Z = 1 quando si è presentata la sequenza 110 e X = 1, Z = 0 altrimenti
- Se dallo stato S_0 si riceve 1, si va in uno **stato S_1** che definiamo con **«è stato riconosciuto in ingresso il primo bit della sequenza (1)»**

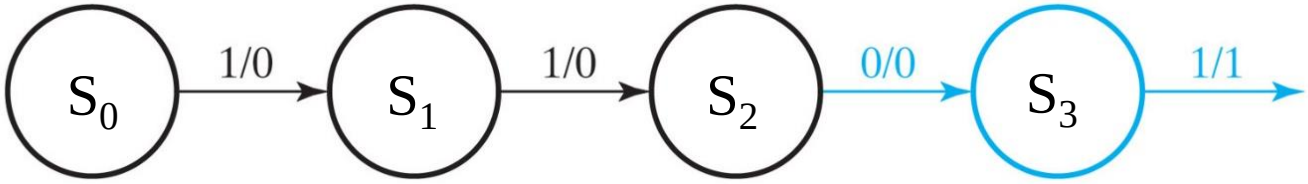


- Se da S_1 si riceve 1, si va allo stato **S_2 «sono stati riconosciuti in ingresso, i primi 2 bit della sequenza (11)»**



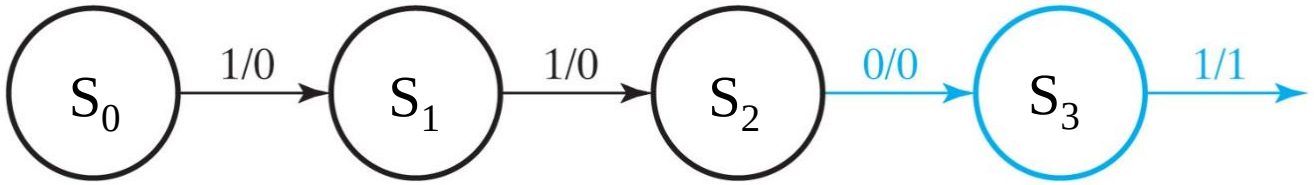
Esempio 4.3: Riconoscitore di sequenza

- **Ingresso: X, Uscita: Z, riconosce la sequenza 1101**
  - Il circuito è inizializzato a 0 (reset all'accensione)
  - Z = 1 quando si è presentata la sequenza 110 e X = 1, Z = 0 altrimenti
- Se dallo stato S_2 riceve 0, va in uno **stato S_3** che definisco con **«sono stati riconosciuti in ingresso i primi 3 bit della sequenza 110»**
- Infine se da S_3 riceve 1, l'uscita diventa 1



Esempio 4.3: Riconoscitore di sequenza

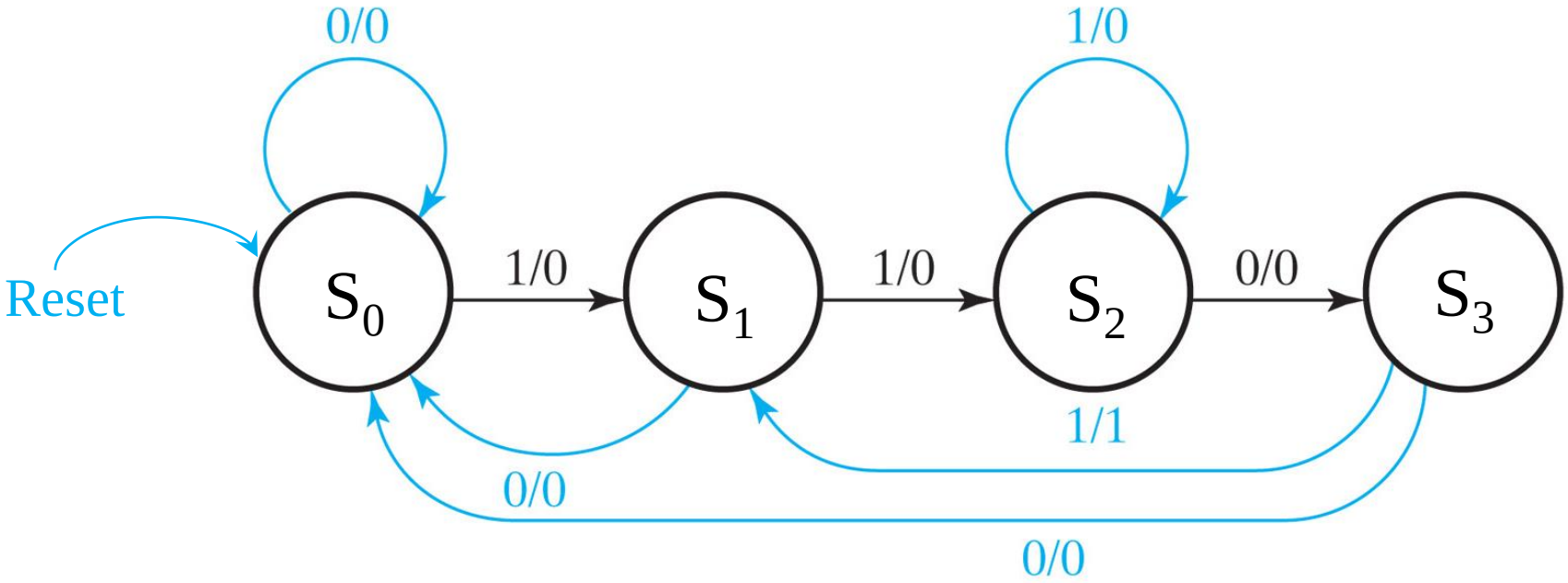
- **Ingresso: X, Uscita: Z, riconosce la sequenza 1101**
- Finora abbiamo realizzato un diagramma di stato parziale (considerate solo le transizioni con sequenza corretta)



- Ora dobbiamo completare il diagramma per le altre transizioni
  - Se dallo stato S_0 riceve 0, vi deve rimanere (non fa parte della sequenza da riconoscere)
  - Se da S_1 riceve 0, deve tornare a S_0 («nessun bit della sequenza è stato riconosciuto»)
  - Se da S_2 riceve 1, vi deve rimanere(«sono stati riconosciuti 2 bit della sequenza»)
  - Se da S_3 riceve 0, deve tornare a S_0 («nessun bit della sequenza è riconosciuto»)
  - Se da S_3 riceve 1, deve andare a S_1 («è stato riconosciuto 1 bit della sequenza»)

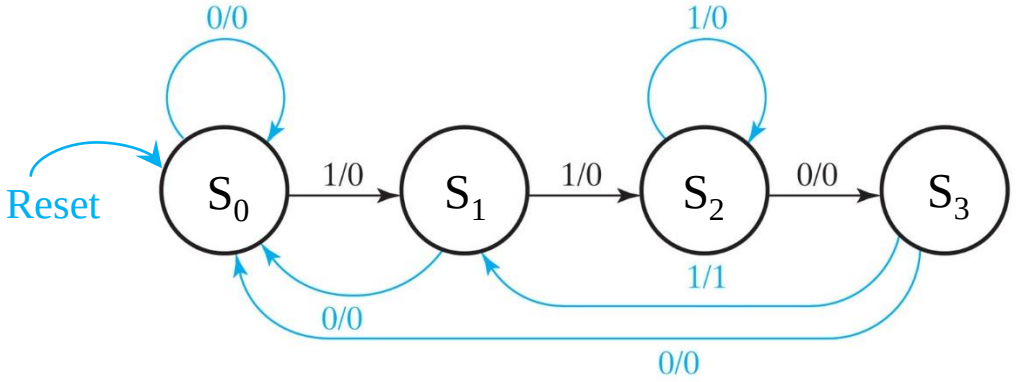
Esempio 4.3: Riconoscitore di sequenza

- **Ingresso: X, Uscita: Z, riconosce la sequenza 1101**
  - Il circuito è inizializzato a 0 (reset all'accensione)
  - Z = 1 quando si è presentata la sequenza 110 e X = 1, Z = 0 altrimenti
- Il **diagramma di stato completo** risulta:



Esempio 4.3: Riconoscitore di sequenza

- **Ingresso: X, Uscita: Z, riconosce la sequenza 1101**
- Deriviamo la tabella delle transizioni di stato e delle uscite



Present State	Next State X = 0	Next State X = 1	Output Z X = 0	Output Z X = 1
S_0	S_0	S_1	0	0
S_1	S_0	S_2	0	0
S_2	S_3	S_2	0	0
S_3	S_0	S_1	0	1

Codifica degli stati

Codifica degli stati

- Finora abbiamo indicato gli stati con dei nomi simbolici, ora assegneremo un codice binario a ciascuno stato
- Per rappresentare **m stati** servono n bit con $2^n \geq m$ ($n = \lceil \log_2 m \rceil$)
  - Si possono scegliere diverse codifiche ed esistono **diversi metodi per assegnare gli stati**, più o meno complessi (es. per ottenere una parte combinatoria a due livelli)
  - Non sempre usando meno bit per lo stato (quindi meno flip-flop) si ottimizza il costo del circuito globale (dobbiamo considerare anche il costo della parte combinatoria)
  - Attenzione: l'ottimizzazione della codifica è un problema complesso e spesso risolto con metodi euristici, non esiste un algoritmo che garantisca il risultato ottimo
- Vedremo due esempi (poi confronteremo i costi delle due scelte)
  i. Stati rappresentati con codice Gray
  ii. Stati rappresentati con codifica 1-hot

Codifica stati: 2 metodi a confronto

i. Stati rappresentati con **codice Gray**

- Rende più facile l'inserimento delle funzioni stato futuro e uscita nelle tabelle di Karnaugh
- Viene usato il numero minimo di bit per la rappresentazione degli stati, quindi minimizza il numero di flip-flop nel circuito

S_0= 00
S_1 = 01
S_2 = 11
S_3 = 10

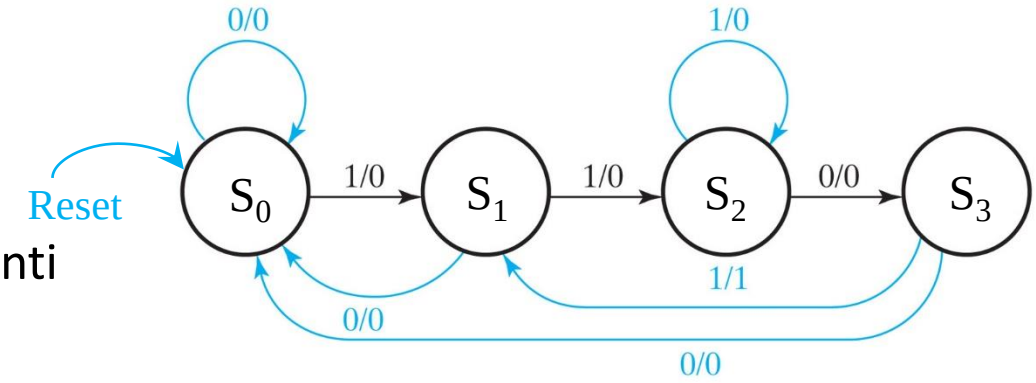
ii. Stati rappresentati con **codifica 1-hot**

- Compare un solo 1 nella codifica di ciascuno stato, quindi la logica per entrare in ciascuno stato è separata da quella per entrare negli altri
- Viene usato un numero maggiore di flip-flop, ma generalmente la logica combinatoria risulta più semplice

S_0 = 1000
S_1 = 0100
S_2 = 0010
S_3 = 0001

Esempio 4.3: Riconoscitore di sequenza

In: X, Out: Z, riconosce 1101
Z = 1 quando si è presentata la sequenza 110 e X = 1, Z = 0 altrimenti

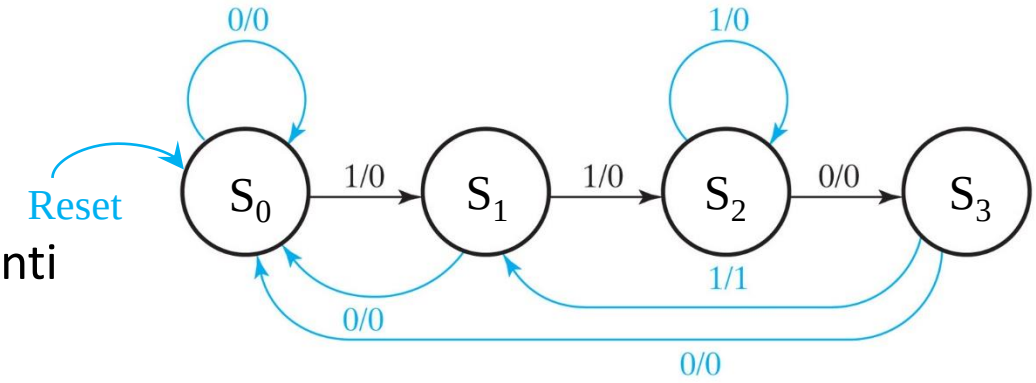


i. Stati con **codifica binaria Gray** (2 bit):

Present State	Next State		Output Z	
AB	**X = 0**	**X = 1**	**X = 0**	**X = 1**
00	00	01	0	0
01	00	11	0	0
11	10	11	0	0
10	00	01	0	1

Esempio 4.3: Riconoscitore di sequenza

In: X, Out: Z, riconosce 1101
Z = 1 quando si è presentata la sequenza 110 e X = 1, Z = 0 altrimenti



ii. Stati con **codifica 1 hot** (4 bit), 1 flip-flop per stato:

Present State	Next State		Output *Z*	
ABCD	**X = 0**	**X = 1**	**X = 0**	**X = 1**
1000	1000	0100	0	0
0100	1000	0010	0	0
0010	0001	0010	0	0
0001	1000	0100	0	1

Progettazione del circuito usando flip-flop D

Implementazione della macchina a stati

- Una volta trovato il diagramma (e/o tabella) degli stati e individuata una codifica per gli stati, la progettazione del circuito si riduce al problema di **progettare due circuiti combinatori**
  - un circuito combinatorio per determinare lo **stato futuro**
  - un circuito combinatorio per determinare l'**uscita**

Esempio 4.3: Riconoscitore di sequenza

In: X, Out: Z, riconosce 1101

Z = 1 quando si è presentata la sequenza 110 e X = 1

Z = 0 altrimenti

i. Stati con codifica binaria Gray:

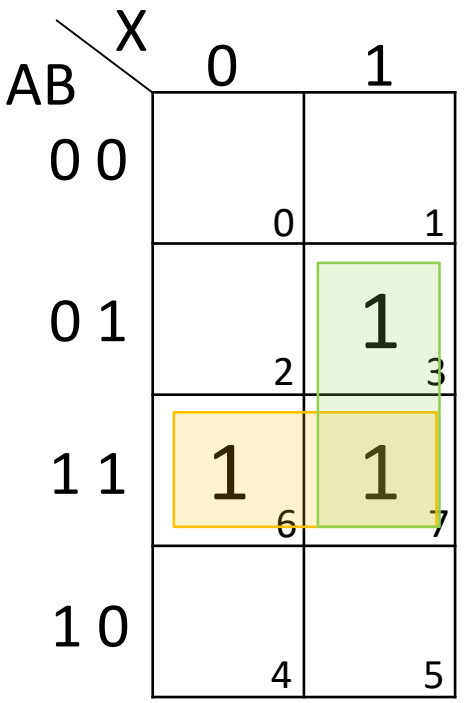
Present State	Next State		Output Z	
AB	$X = 0$	$X = 1$	$X = 0$	$X = 1$
00	00	01	0	0
01	00	11	0	0
11	10	11	0	0
10	00	01	0	1

Troviamo le **equazioni degli stati futuri** (2 bit di stato => 2 flip-flop)

- 1° bit dello stato (A):

$$A(t+1) = D_A(A, B, X) = \sum m(3, 6, 7)$$

$$\Rightarrow D_A = AB + BX$$



Esempio 4.3: Riconoscitore di sequenza

In: X, Out: Z, riconosce 1101

Z = 1 quando si è presentata la sequenza 110 e X = 1

Z = 0 altrimenti

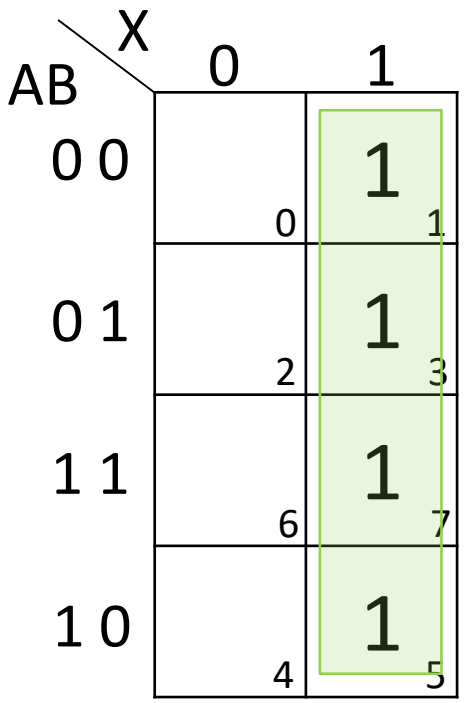
i. Stati con codifica binaria Gray:

Present State	Next State		Output Z	
AB	$X = 0$	$X = 1$	$X = 0$	$X = 1$
00	00	01	0	0
01	00	11	0	0
11	10	11	0	0
10	00	01	0	1

- 2° bit dello stato (B):

$$B(t+1) = D_B(A, B, X) = \sum m(1, 3, 5, 7)$$

$$\Rightarrow D_B = X$$



Esempio 4.3: Riconoscitore di sequenza

In: X, Out: Z, riconosce 1101

Z = 1 quando si è presentata la sequenza 110 e X = 1
Z = 0 altrimenti

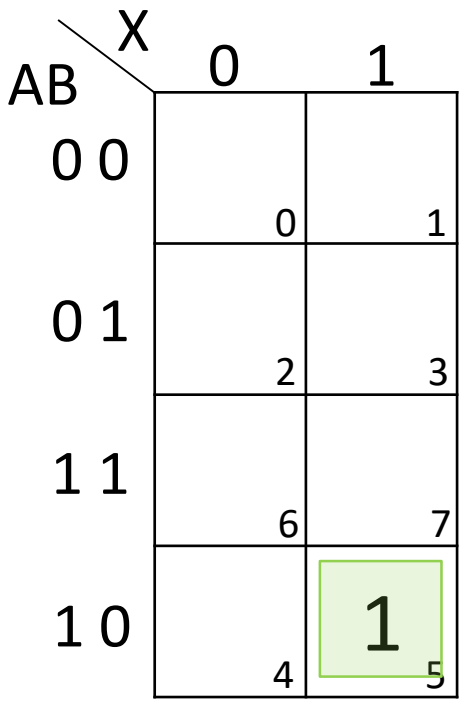
i. Stati con codifica binaria Gray:

Present State	Next State		Output Z	
AB	$X = 0$	$X = 1$	$X = 0$	$X = 1$
00	00	01	0	0
01	00	11	0	0
11	10	11	0	0
10	00	01	0	1

- Troviamo ora l'equazione dell'uscita:

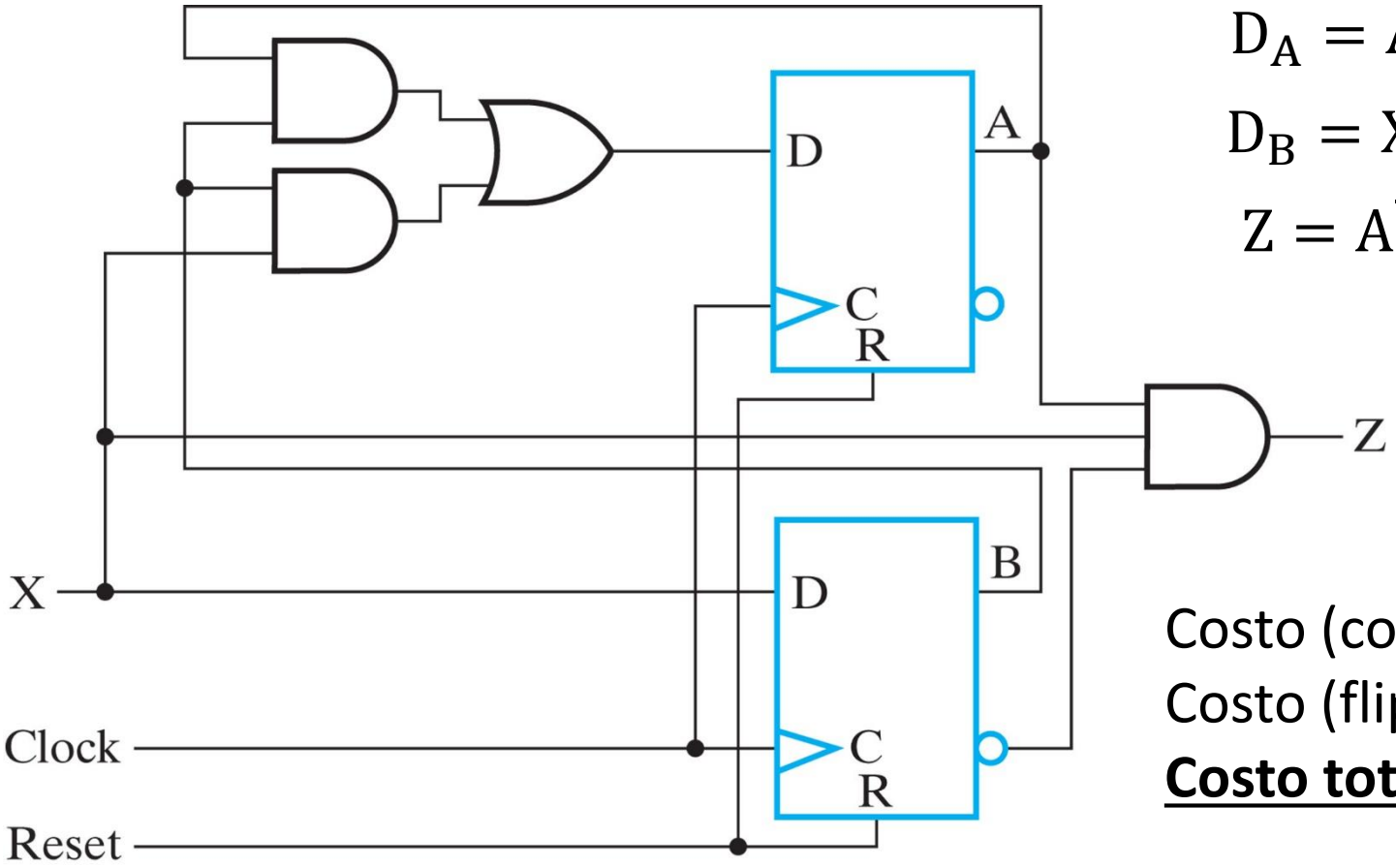
$$Z(A, B, X) = \sum m(5)$$

$$\Rightarrow Z = A\overline{B}X$$



Esempio 4.3: Riconoscitore di sequenza

Circuito finale: rappresentando gli stati con codifica binaria Gray servono 2 flip-flop per immagazzinare lo stato (registro di stato)



$$D_A = AB + BX$$

$$D_B = X$$

$$Z = A\overline{B}X$$

Costo (comb) = 9

Costo (flip-flop) = 14*2

Costo tot = 37

Esempio 4.3: Riconoscitore di sequenza

In: X, Out: Z, riconosce 1101
Z = 1 quando si è presentata la sequenza 110 e X = 1
Z = 0 altrimenti

ii. Stati con codifica 1-hot:

Present State	Next State		Output *Z*	
ABCD	X = 0	X = 1	X = 0	X = 1
1000	1000	0100	0	0
0100	1000	0010	0	0
0010	0001	0010	0	0
0001	1000	0100	0	1

Troviamo le equazioni dello stato futuro (4 bit di stato => 4 flip-flop):

$$A\,(t+1) = D_A = A\overline{X} + B\overline{X} + D\overline{X} = (A + B + D)\,\overline{X}$$

$$B\,(t+1) = D_B = AX + DX = (A + D)\,X$$

$$C\,(t+1) = D_C = BX + CX = (B + C)\,X$$

$$D\,(t+1) = D_D = C\overline{X}$$

Non si usano le MdK, le equazioni sono semplici da scrivere: basta un solo bit per identificare lo stato presente

Esempio 4.3: Riconoscitore di sequenza

In: X, Out: Z, riconosce 1101
Z = 1 quando si è presentata la sequenza 110 e X = 1
Z = 0 altrimenti

ii. Stati con codifica 1-hot:

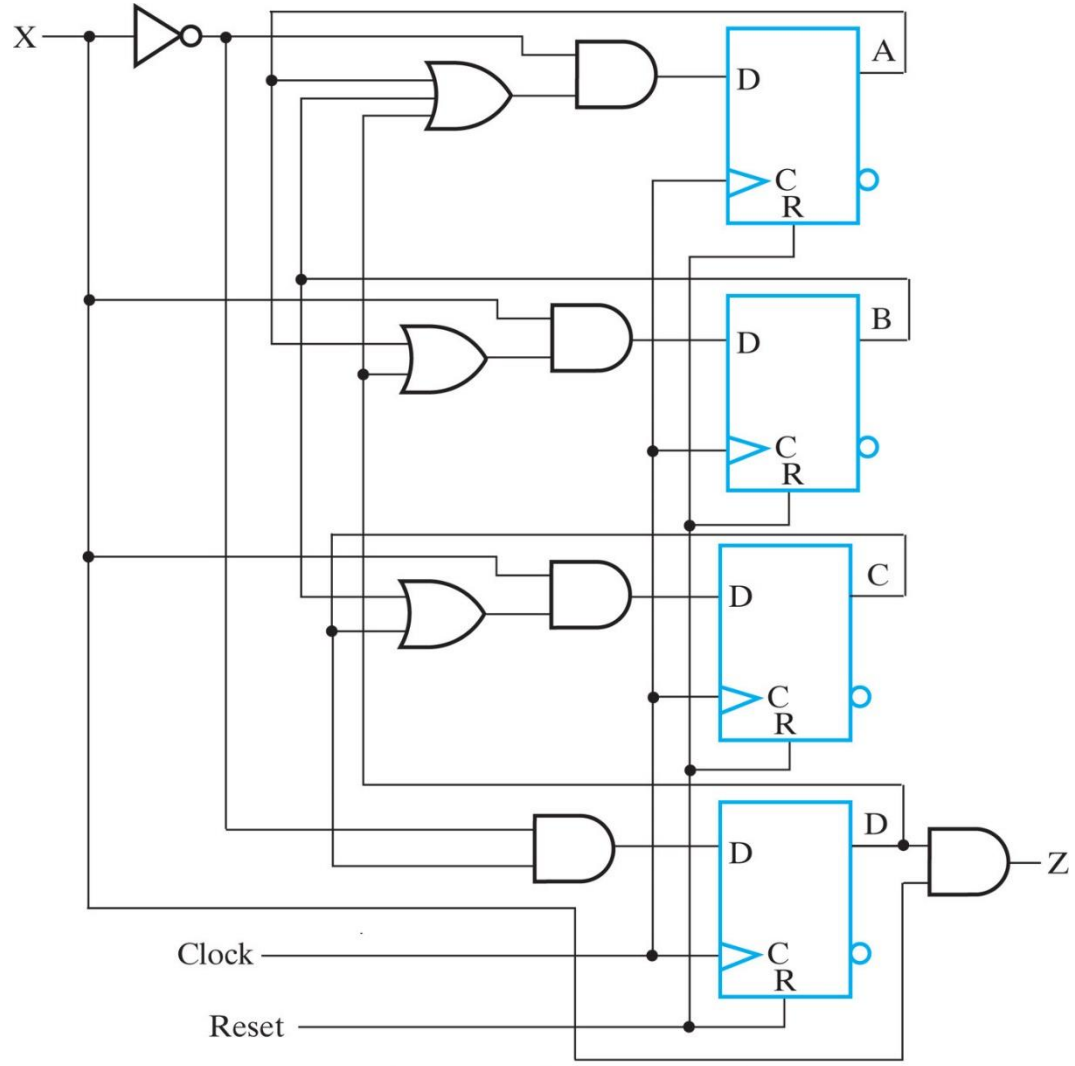
Present State	Next State		Output *Z*	
ABCD	X = 0	X = 1	X = 0	X = 1
1000	1000	0100	0	0
0100	1000	0010	0	0
0010	0001	0010	0	0
0001	1000	0100	0	1

Troviamo l'equazione dell'uscita:

$$Z = DX$$

Esempio 4.3: Riconoscitore di sequenza

Circuito finale rappresentando gli stati con codifica 1-hot servono 4 flip-flop per immagazzinare lo stato (1 flip-flop per ogni stato):

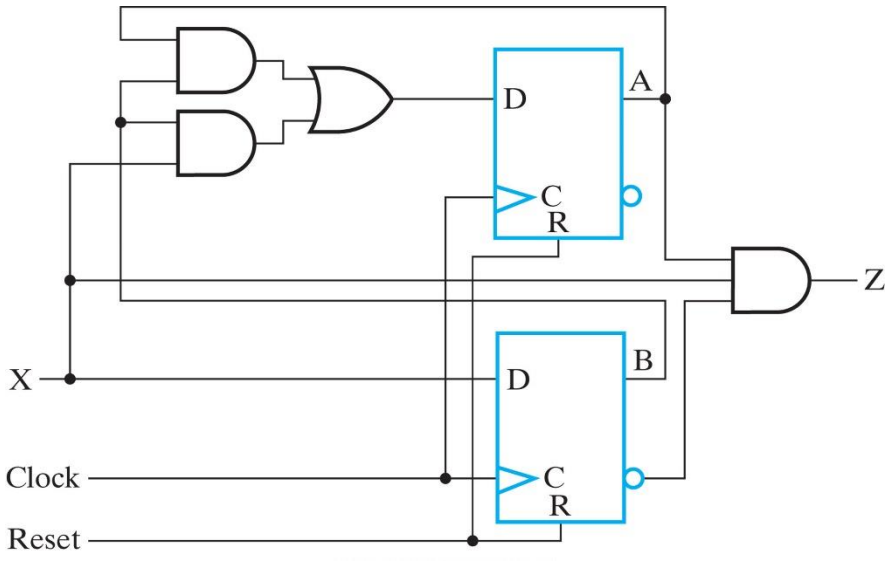


$$D_A = (A + B + D)\,\overline{X}$$
$$D_B = (A + D)\,X$$
$$D_C = (B + C)\,X$$
$$D_D = C\overline{X}$$
$$Z = DX$$

Costo (comb) = 18
Costo (flip-flop) = 14*4
Costo tot = 74

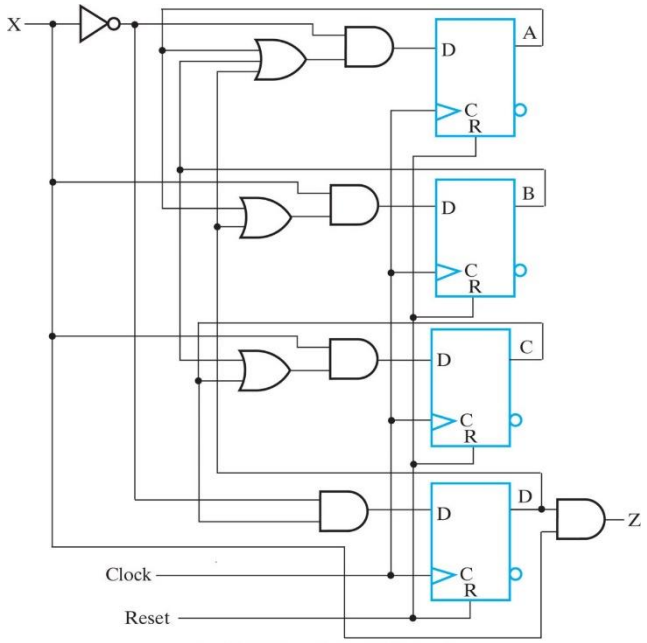
Esempio 4.3: Riconoscitore di sequenza

Circuito con codifica Gray:



Costo tot = 37

Circuito con codifica 1-hot:



Costo tot = 74

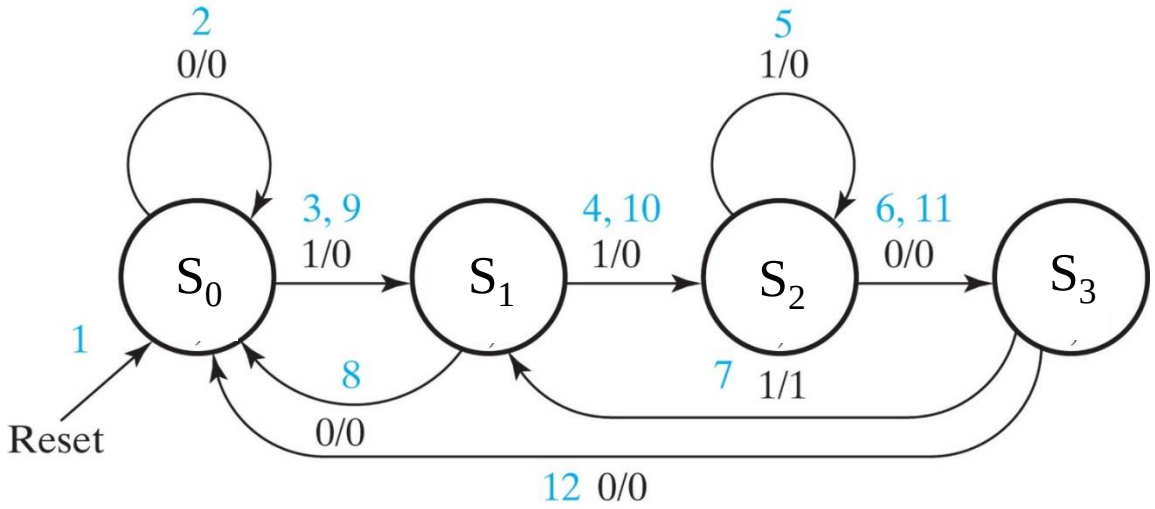
- Con la codifica Gray si risparmia circa la metà (gate input cost)
- Con la codifica 1-hot in alcuni casi si possono avere vantaggi (e.g. design più agevole, maggiore affidabilità, migliori performance del circuito)

Verifica

Verifica di un circuito sequenziale

- Lo scopo è verificare che il **circuito dia luogo al diagramma degli stati atteso** (o tabella degli stati)
  - Si applicano diverse combinazioni di input e si osserva se il circuito, agli opportuni cambiamenti del clock, produce le transizioni attese negli stati e nelle uscite
  - All'inizio del processo di verifica applicare la sequenza che porta ad un reset del circuito
- Per circuiti sufficientemente semplici, si possono applicare tutte le combinazioni di stato corrente/ingresso in modo da **verificare tutte le possibili transizioni**
- Per circuiti più complessi si applicano **alcuni vettori di test significativi**
- Nella realtà la verifica è un'attività molto complessa che si avvale di strumenti e tecniche che non affrontiamo in questo corso

Esempio 4.3: Riconoscitore di sequenza



Sequenza di ingressi da applicare per la verifica del circuito:

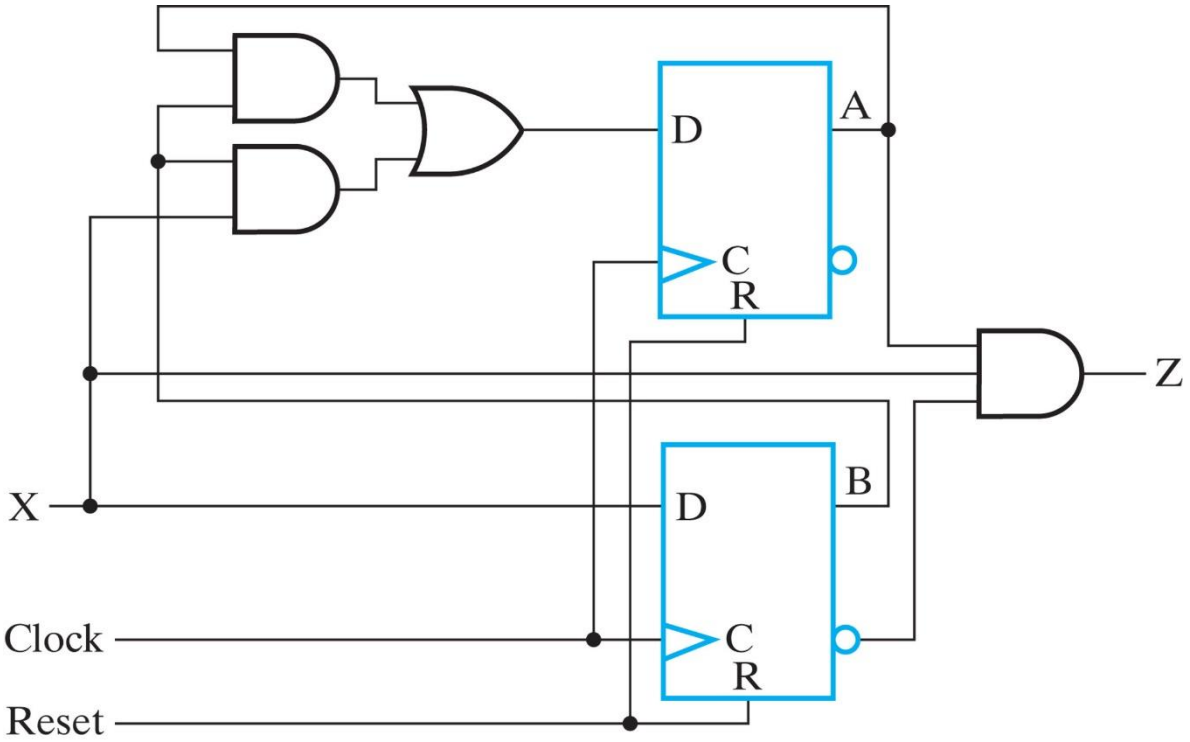
1. Reset
2. Stato S_0, ingresso 0
3. Stato S_0, ingresso 1
4. Stato S_1, ingresso 1
5. Stato S_2, ingresso 1
6. Stato S_2, ingresso 0
7. Stato S_3, ingresso 1
8. Stato S_1, ingresso 0
9. Stato S_0, ingresso 1
10. Stato S_1, ingresso 1
11. Stato S_2, ingresso 0
12. Stato S_3, ingresso 0

Disegno il percorso più corto che passa per tutte le combinazioni di stato-ingresso nel diagramma degli stati

NB: Per verificare tutte le combinazioni stato-ingresso (8 in questo esempio) ci vogliono tipicamente un numero maggiore di transizioni (bisogna ripassare più volte su alcuni stati)

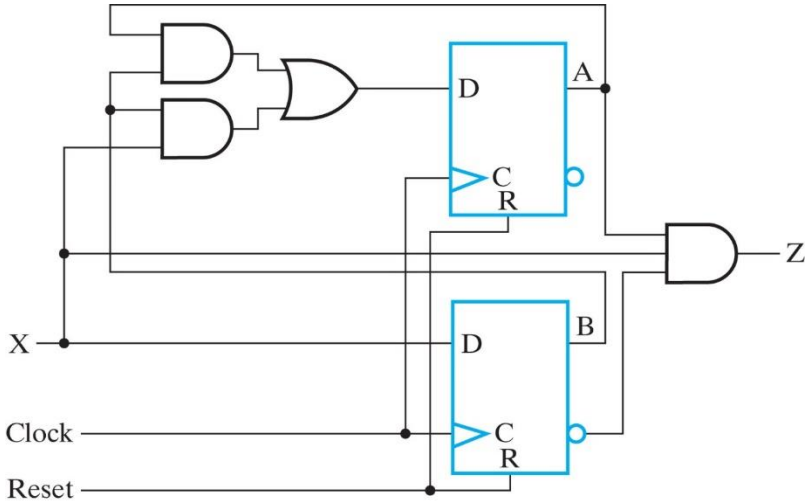
Esempio 4.3: Riconoscitore di sequenza (stati codificati con codice Gray)

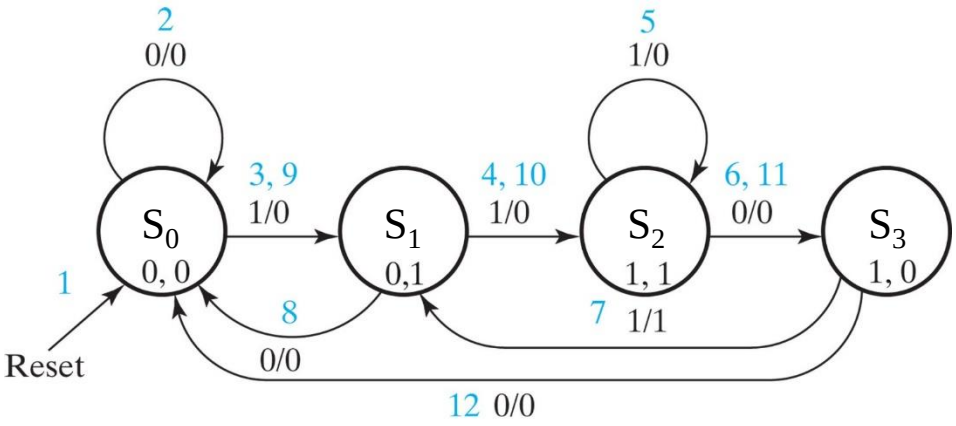
State	Bit A	Bit B
S_0	0	0
S_1	0	1
S_2	1	1
S_3	1	0

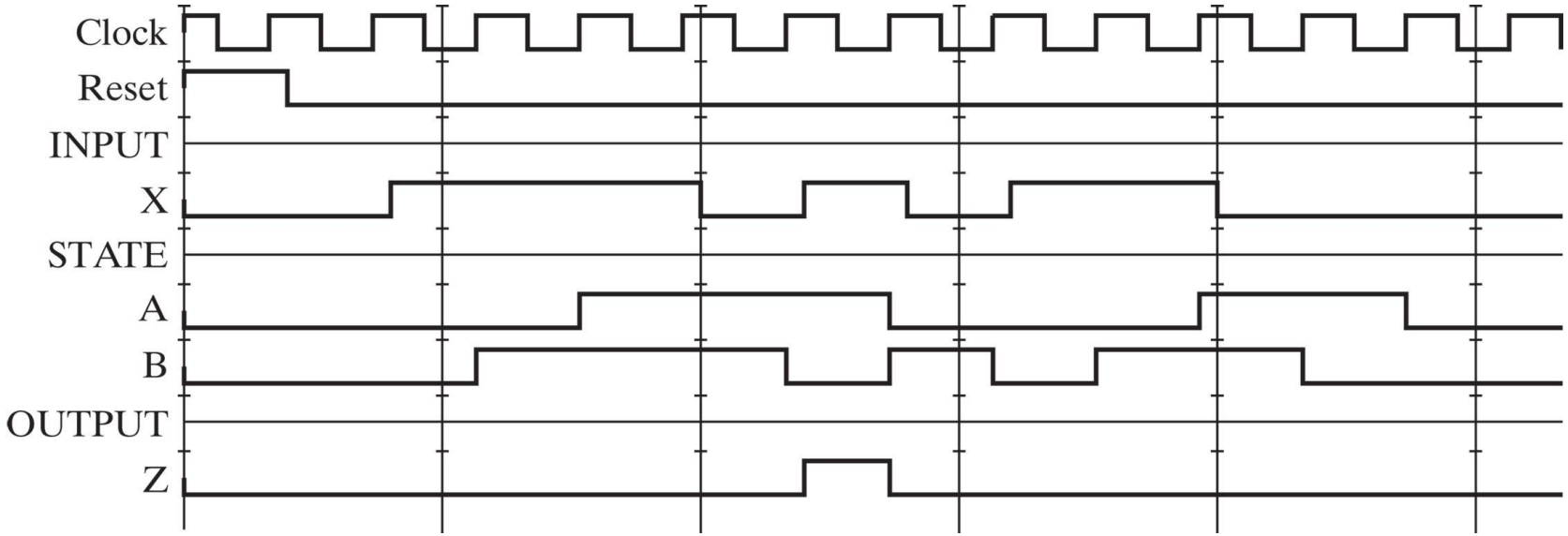


- Per verificare ciascuna transizione, applichiamo lo stato presente e il valore dell'ingresso e osserviamo lo stato futuro
  - Nel flip-flop D lo stato futuro è pari all'ingresso D prima del fronte del clock
  - Es: Stato presente S_0, Ingresso 1 -> Stato futuro S_1, Uscita 0

Esempio 4.3: Riconoscitore di sequenza (stati codificati con codice Gray)







Riepilogo

- Un sistema sequenziale sincrono consiste in un registro di stato (flip-flop che immagazzinano lo stato del sistema) e in una rete combinatoria per calcolare lo stato futuro e le uscite
- Passi per la progettazione di un circuito sequenziale sincrono:
  - Traduzione delle specifiche in un diagramma degli stati che descrive il funzionamento del sistema
  - Tabella degli stati e delle uscite: per ogni combinazione del valore di stato presente e ingressi, fornisce lo stato futuro e le uscite
  - Assegnazione di una codifica agli stati
  - Tabella di stato contenente la codifica scelta
  - Sintesi della logica combinatoria per determinare lo stato futuro
  - Sintesi della logica combinatoria per determinare l'uscita
  - Verifica del funzionamento del circuito

Disclaimer

Figures from *Logic and Computer Design Fundamentals*, Fifth Edition, GE Mano | Kime | Martin