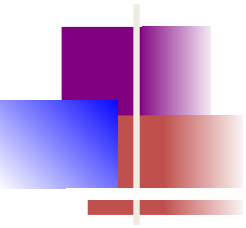
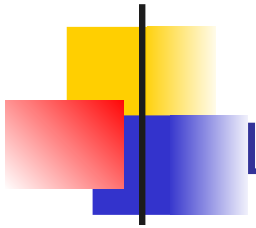


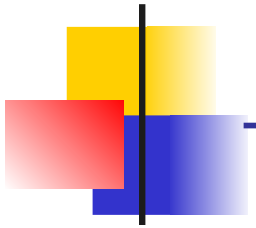
Arquitectura de Computadoras
Circuitos secuenciales

La mayor parte de los gráficos son de ***Principles of Computer Architecture*** *(1999) de Miles Murdocca y Vincent Heuring*



Lógica secuencial

- Los circuitos combinatorios no tienen memoria. Las salidas quedan totalmente determinadas por los valores presentes de las entradas
- Existe la necesidad de circuitos con memoria, cuyo comportamiento pueda depender, no sólo de los valores presentes de las entradas, sino también de los estados previos
- Un ejemplo es una máquina expendedora automática, que debe recordar cuantas monedas y de que valor han sido insertadas. El comportamiento de la máquina debe depender no sólo de la última moneda insertada sino también de cuantas y de que valor han sido insertadas previamente
- Estos circuitos son denominados Máquinas de Estado Finito (Finite State machines o FSM), debido a que tienen un número finito de estados
- Una máquina de estado finito toma las entradas y el estado actual para generar tanto las salidas como un nuevo estado

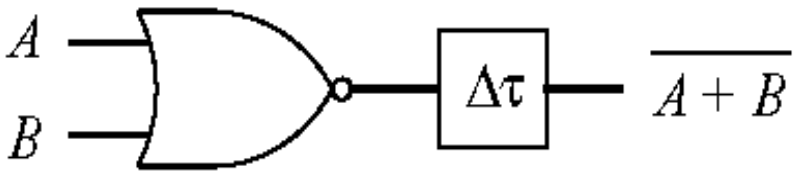


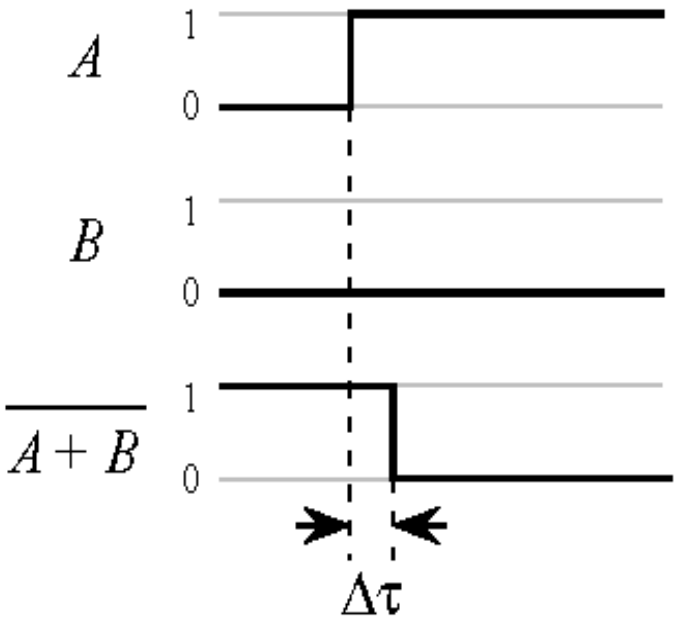
Tipos de circuitos secuenciales

- Circuitos sincrónicos
  - El comportamiento puede definirse a partir del conocimiento del estado de sus señales en instantes discretos
- Circuitos asincrónicos
  - Dependen del orden en el que cambian las entradas. El estado del circuito puede ser afectado en cualquier instante

Retardo de compuerta

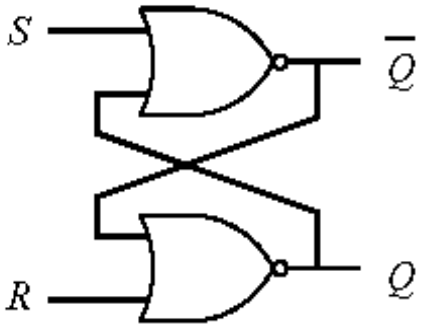
- Cualquier compuerta requiere un tiempo no nulo para calcular el valor de la salida una vez que se han estabilizado los valores de sus entradas. A los fines del análisis podemos acumular este tiempo sobre la salida de la compuerta



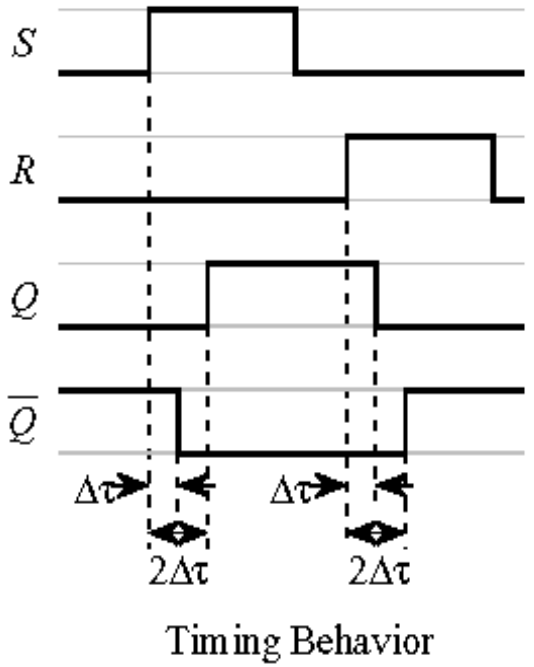


Un flip-flop SR

- El retardo de compuerta es la base de funcionamiento de un importante dispositivo de memoria: el flip-flop SR, que es un dispositivo de lógica positiva (activa alto)

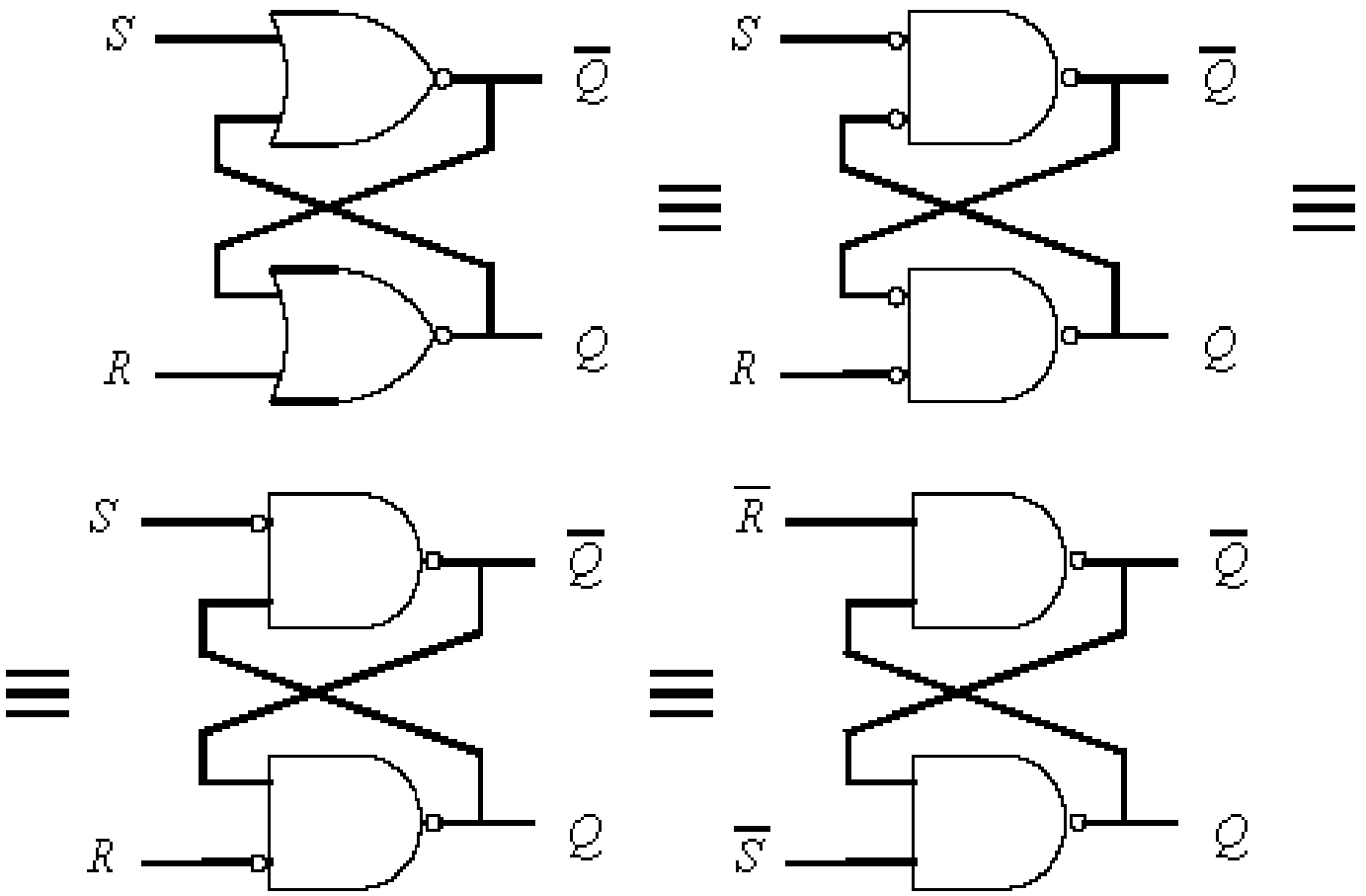


Q_t	S_t	R_t	Q_{t+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	(disallowed)
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	(disallowed)



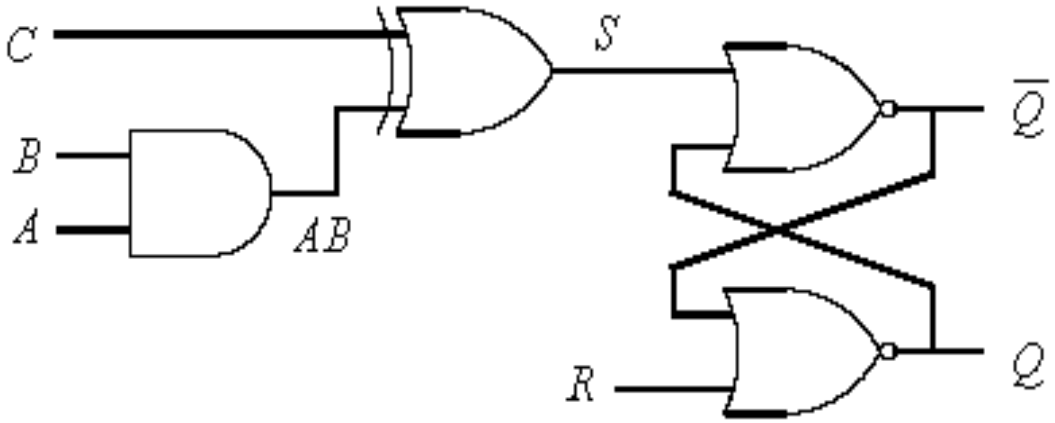
Timing Behavior

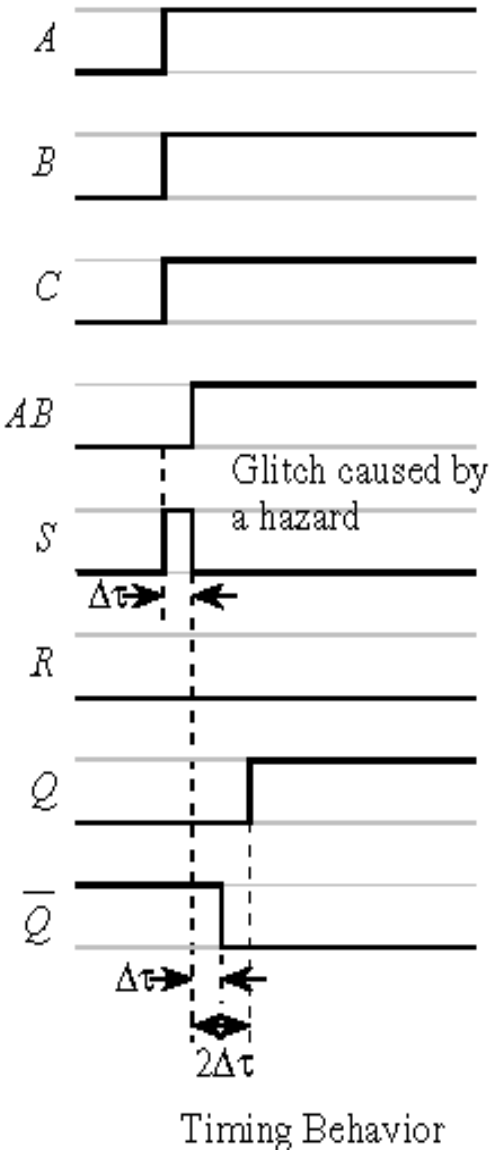
Otras implementaciones de un circuito SR



Un riesgo

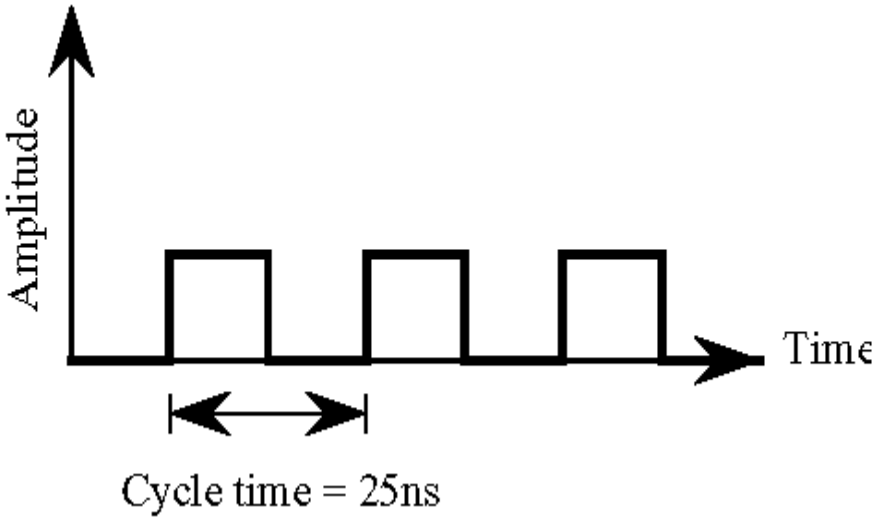
- Un riesgo es que las transiciones de estado de las entradas se produzcan en momentos indeseados (como en el ejemplo)





Relojes

- Para evitarlo es conveniente “apagar“ el flip-flop para que sólo responda a variaciones en las entradas en momentos determinados
- Para ello se suele agregar una señal de **sincronismo** o **reloj**
- En un circuito con lógica **positiva** la acción ocurre cuando el reloj está en un estado alto, o positivo. El tiempo que el reloj está bajo permite la propagación entre subcircuitos, asegurando que las entradas están estabilizadas al valor correcto la próxima vez que el reloj sube.



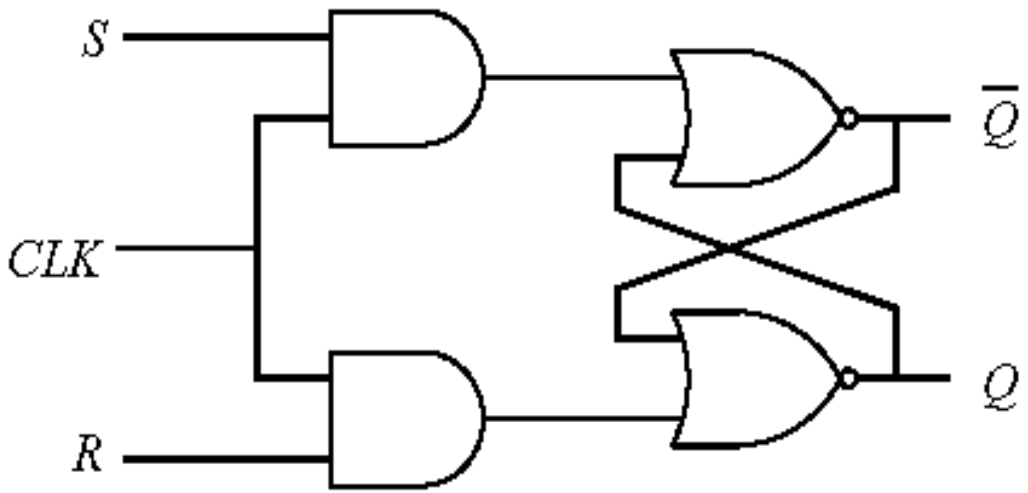
Relojes (2)

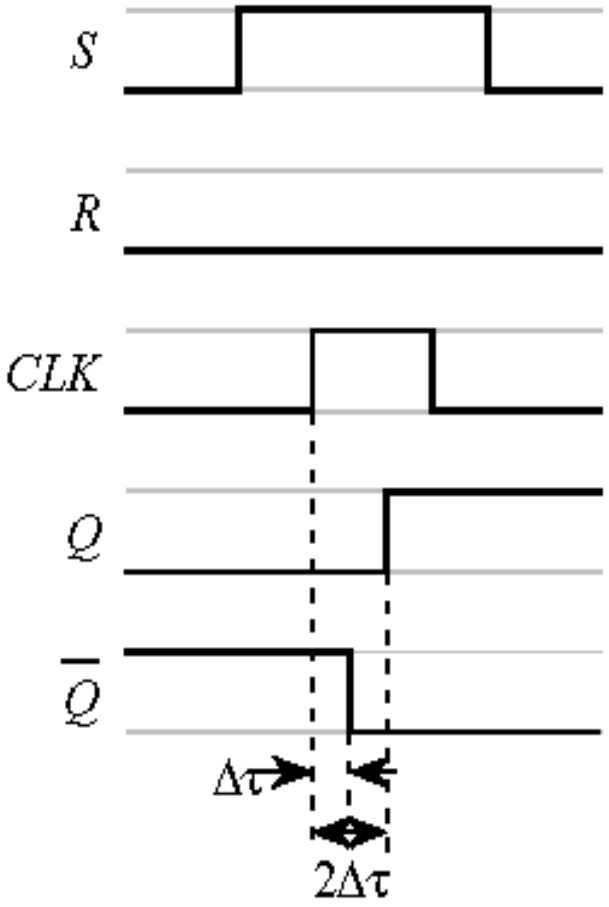
- Se define como **período** o **tiempo de ciclo** el tiempo que demora la señal del reloj para subir, caer y volver a subir
- Se define como **frecuencia** a la inversa del período (f = 1 / p)

Prefix	Abbrev.	Quantity	Prefix	Abbrev.	Quantity
milli	m	10^{-3}	Kilo	K	10^{3}
micro	μ	10^{-6}	Mega	M	10^{6}
nano	n	10^{-9}	Giga	G	10^{9}
pico	p	10^{-12}	Tera	T	10^{12}
femto	f	10^{-15}	Peta	P	10^{15}
atto	a	10^{-18}	Exa	E	10^{18}

Algunas unidades útiles

Un biestable SR sincronizado

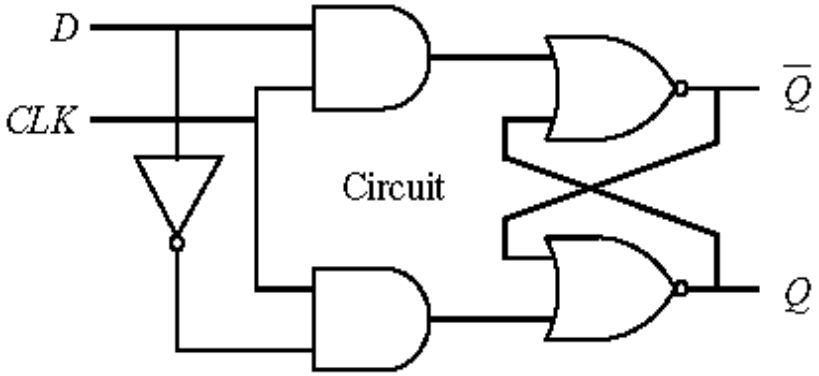


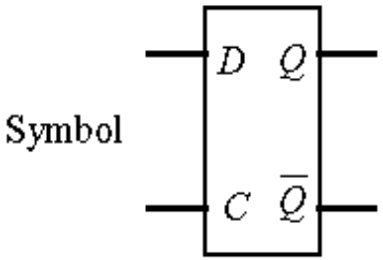


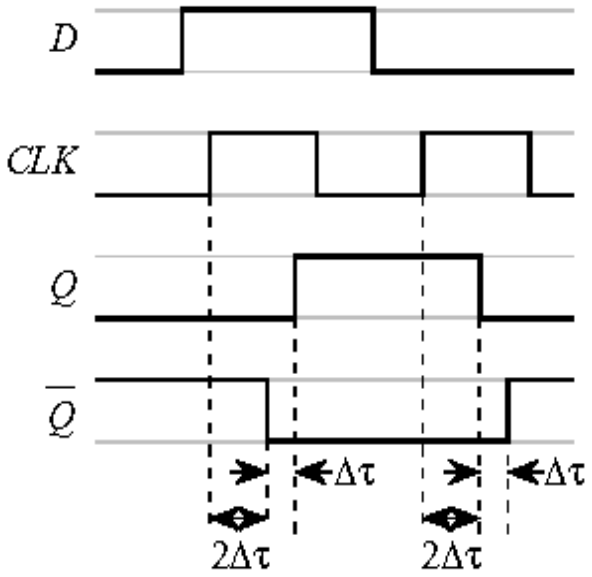
Timing Behavior

El flip-flop D

- El flip-flop SR tiene la desventaja de que para cambiar el estado hay que aplicar un 1 en una de dos entradas diferentes
- Una alternativa diferente, que permite aplicar un 0 o un 1 sobre una única entrada es el flip-flop D



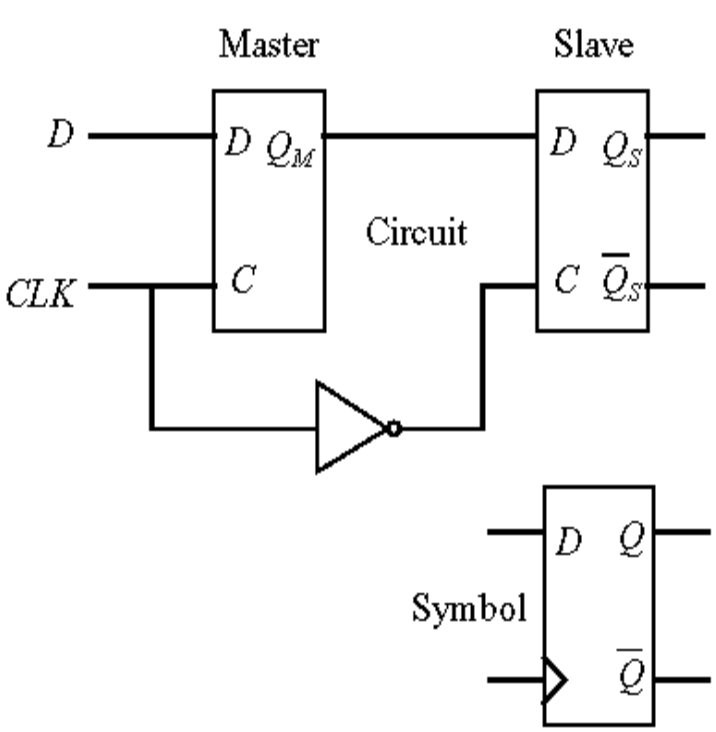


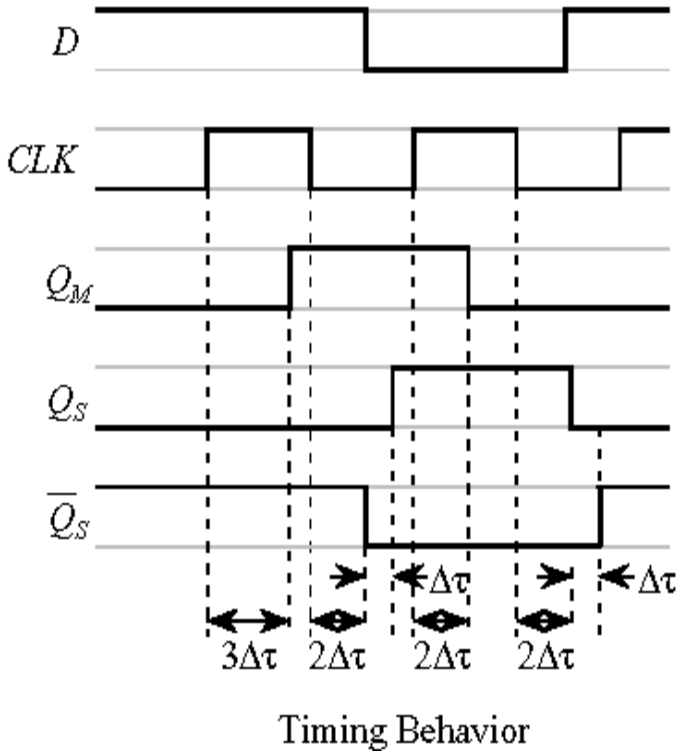


Timing Behavior

El flip-flop D maestro-esclavo

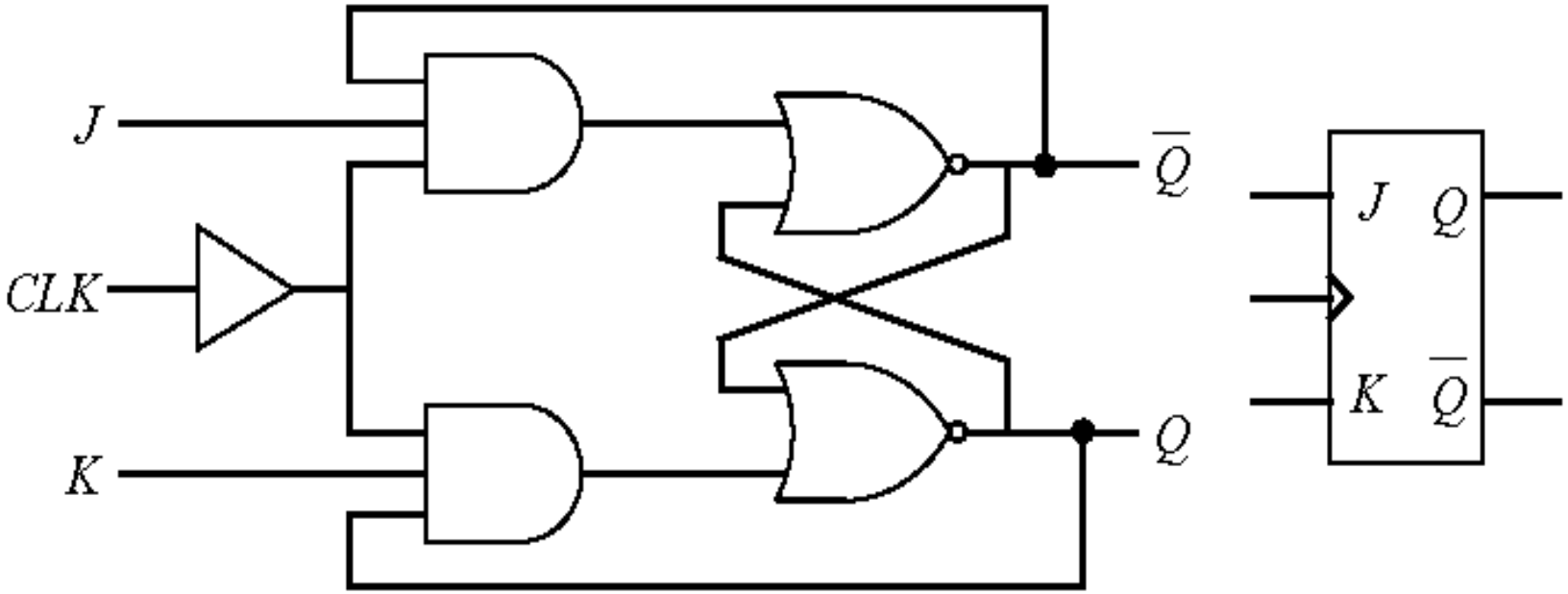
- El flip-flop D tiene el inconveniente de que puede cambiar dos veces el estado durante el ciclo de reloj
- En este caso, el estado **alto** (1) del reloj carga el dato en el maestro y el estado **bajo** (0) en el esclavo





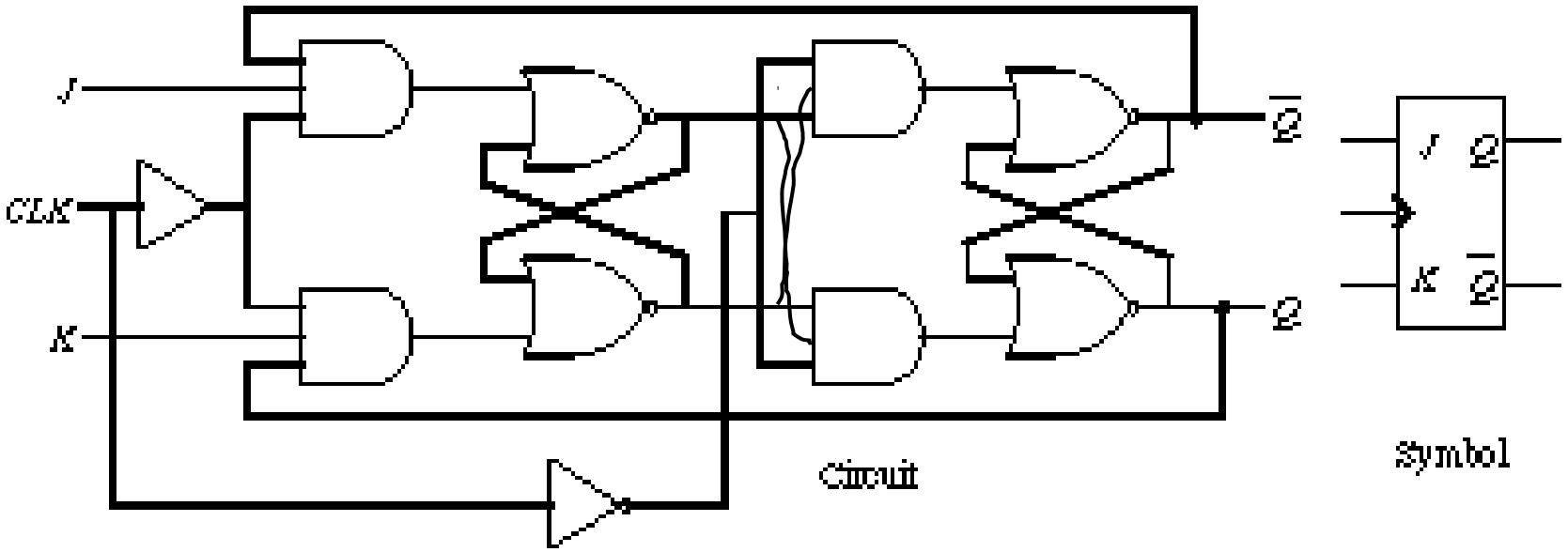
Otros flip-flops: el JK sincronizado

- El flip-flop JK elimina la indeterminación del flip-flop SR cuando ambas entradas son puestas a 1
- No obstante, aún presenta el problema de que puede cambiar de estado más de una vez cuando ambas entradas se mantienen en 1 (oscilación infinita)



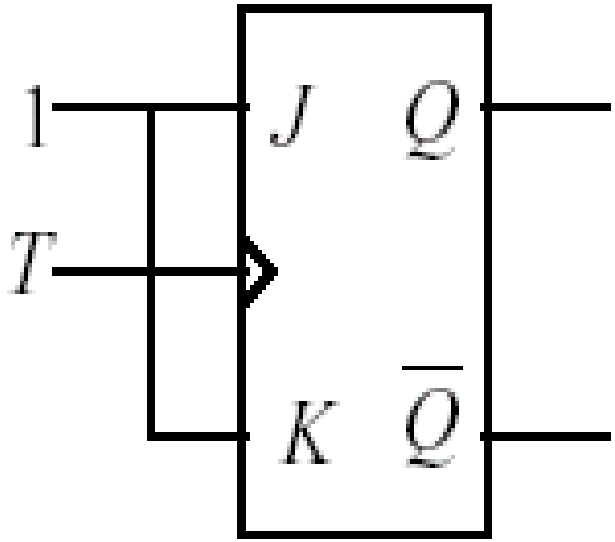
El JK maestro-esclavo

- Resuelve el problema de la oscilación infinita, pero presenta el problema de la “captura de unos”: Si se mantiene una entrada alta mientras el reloj está activo, el flip-flop puede llegar a ver el 1 como una entrada válida.

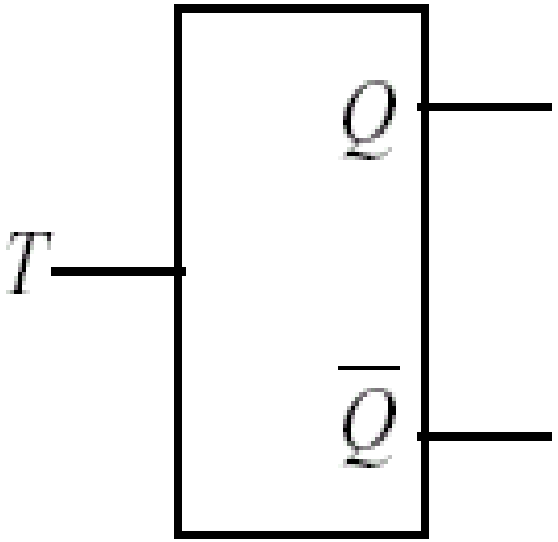


El flip-flop T (toggle)

- La presencia de un 1 en cada una de las entradas de un flip-flop JK hace que el mismo cambie de estado cada vez que es activado por la señal T (*toggle*)



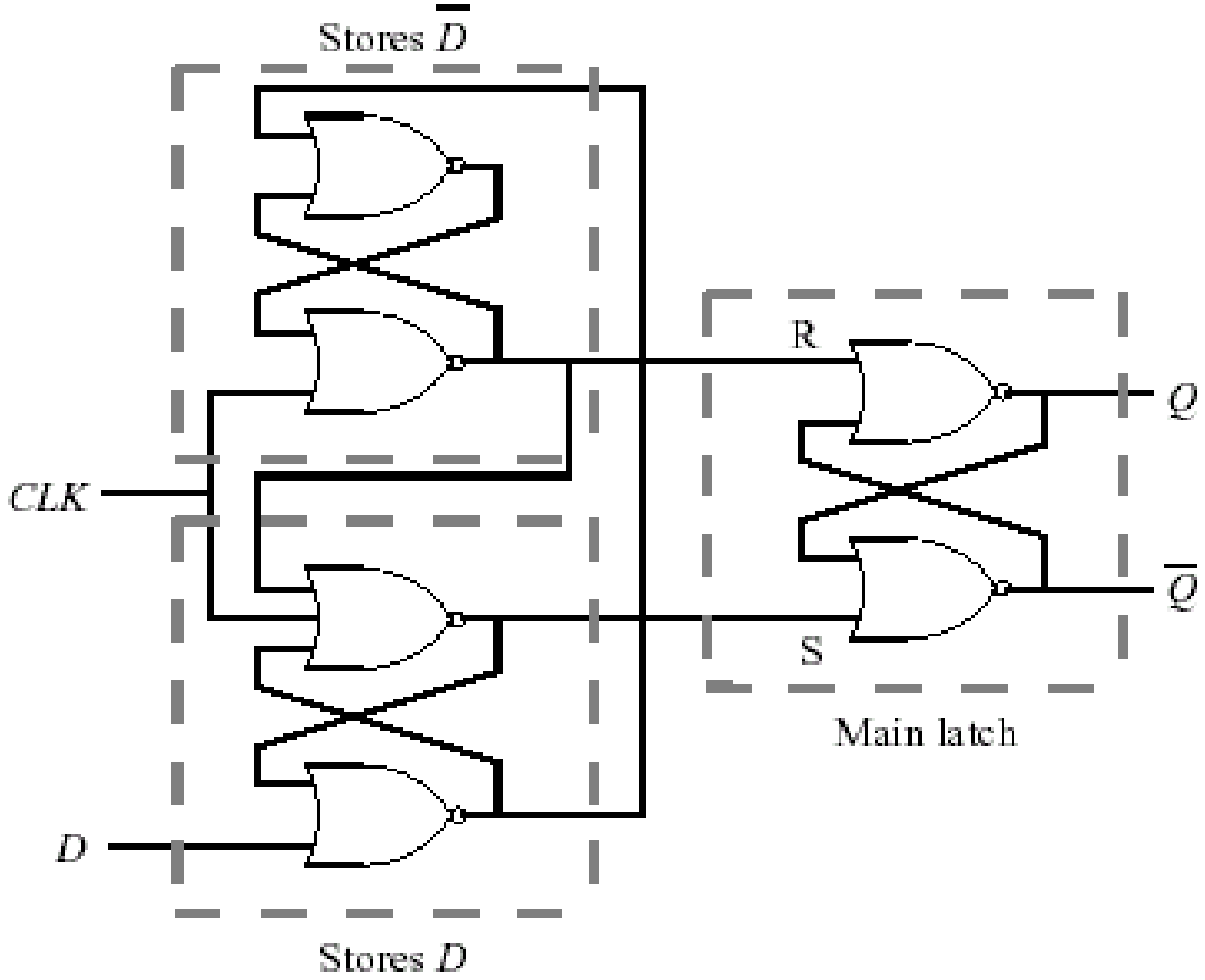
Circuit



Symbol

Activación por flanco

- La captura de 1s se resuelve construyendo flip-flops “activados por flanco”, en los que el estado de la entrada se analiza sólo durante la transición activa del reloj (alto-bajo o bajo-alto)



Ejemplo: un contador módulo 4

- Tiene una entrada de reloj y dos salidas que toman los valores 00, 01, 10, 11, 00, 01 ... con cada pulso de reloj (CLK). Se agrega una entrada de RESET para poner a 0 ambas salidas

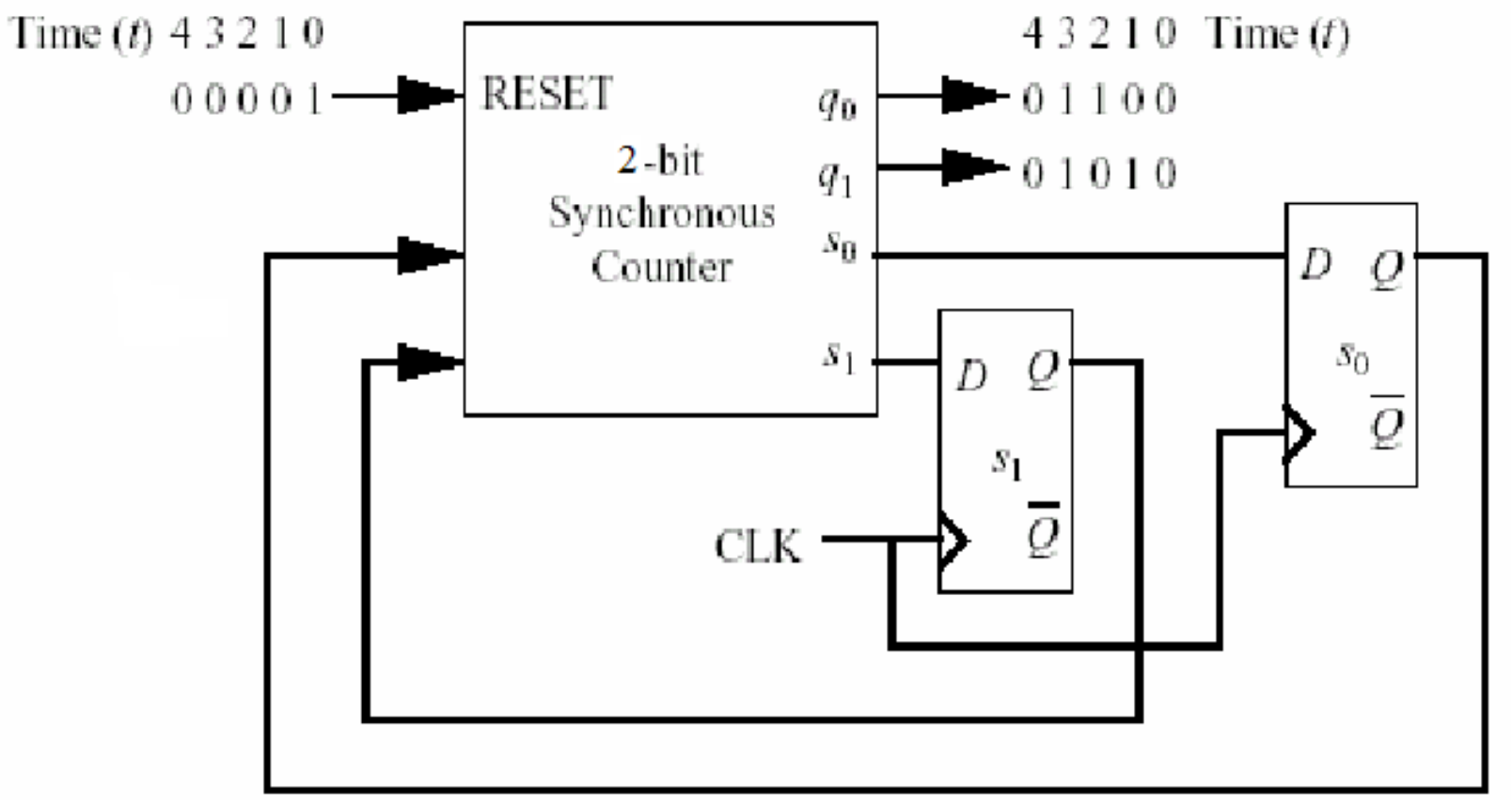


Diagrama de transiciones de estado de un contador módulo 4

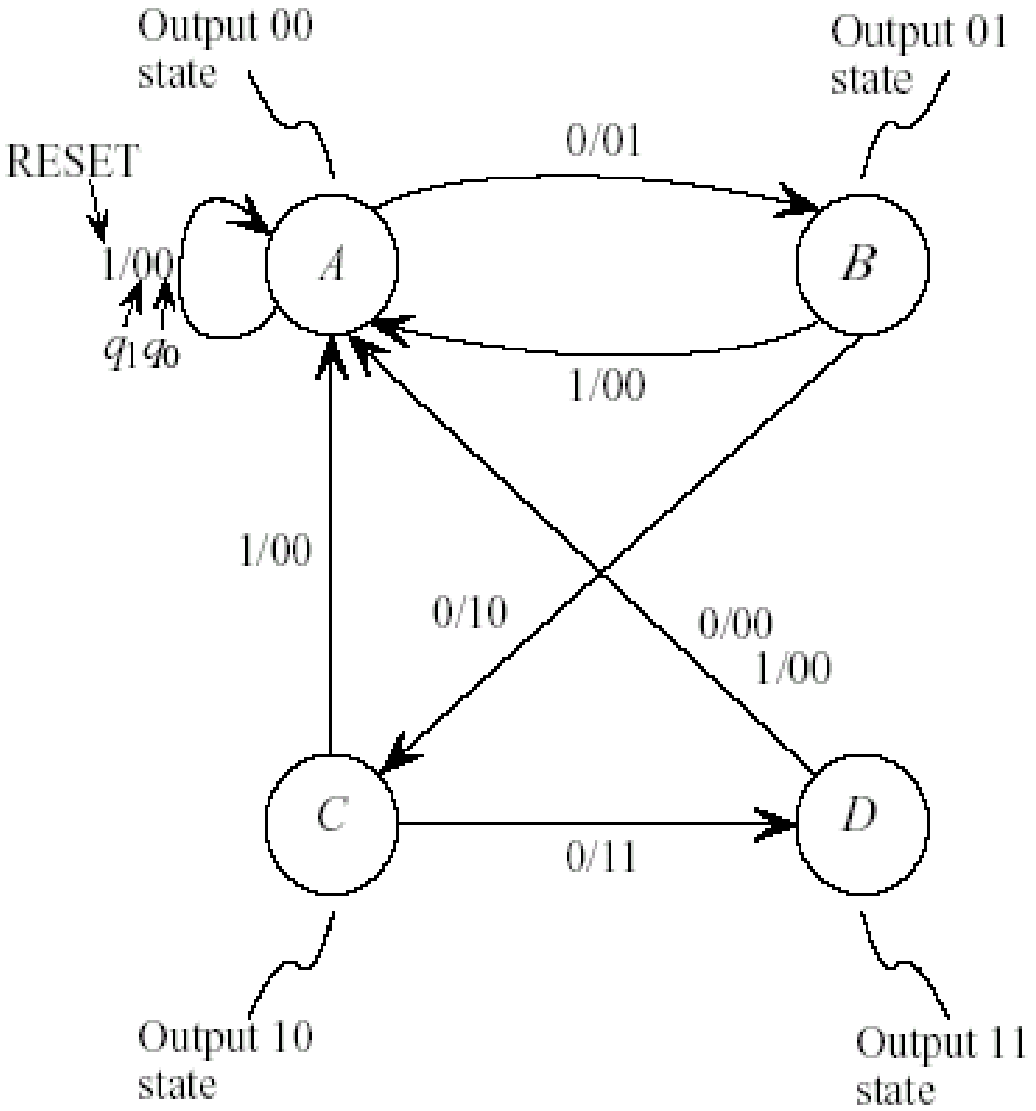


Tabla de estados de un contador módulo 4

Present state \ Input	*RESET* = 0	*RESET* = 1
A	*B*/01	*A*/00
B	*C*/10	*A*/00
C	*D*/11	*A*/00
D	*A*/00	*A*/00

Asignación de estados para un contador módulo 4

Present state (S_t) \ Input	*RESET* = 0	*RESET* = 1
A:00	01/01	00/00
B:01	10/10	00/00
C:10	11/11	00/00
D:11	00/00	00/00

Tabla de verdad de un contador módulo 4

$RESET$ $r(t)$	$s_1(t)$	$s_0(t)$	$s_1s_0(t+1)$	$q_1q_0(t+1)$
0	0	0	01	01
0	0	1	10	10
0	1	0	11	11
0	1	1	00	00
1	0	0	00	00
1	0	1	00	00
1	1	0	00	00
1	1	1	00	00

Derivación de funciones de un contador módulo 4

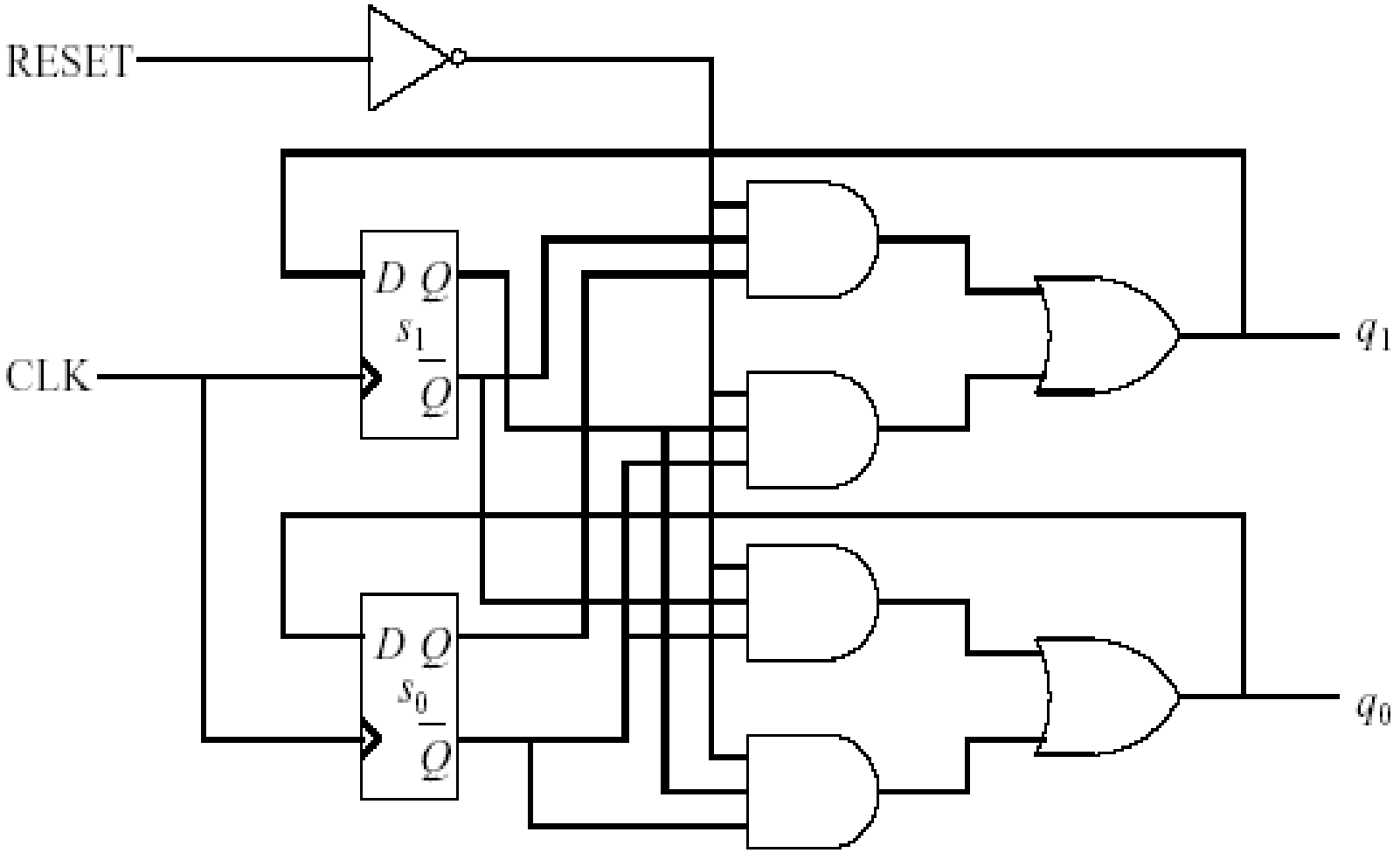
$$s_0(t+1) = \overline{r(t)}\,\overline{s_1(t)}\,\overline{s_0(t)} + \overline{r(t)}\,s_1(t)\,\overline{s_0(t)}$$

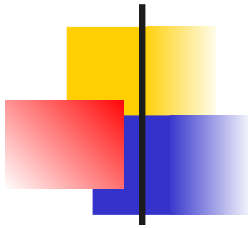
$$s_1(t+1) = \overline{r(t)}\,\overline{s_1(t)}\,s_0(t) + \overline{r(t)}\,s_1(t)\,\overline{s_0(t)}$$

$$q_0(t+1) = \overline{r(t)}\,\overline{s_1(t)}\,\overline{s_0(t)} + \overline{r(t)}\,s_1(t)\,\overline{s_0(t)}$$

$$q_1(t+1) = \overline{r(t)}\,\overline{s_1(t)}\,s_0(t) + \overline{r(t)}\,s_1(t)\,\overline{s_0(t)}$$

Circuito de un contador módulo 4





Ejemplo: un detector de secuencias

- Diseñe una máquina que produzca un 1 en la salida cada vez que exactamente dos de las últimas tres entradas han sido 1
- Por ejemplo, la secuencia de entrada 011011100 debe producir en la salida la 001111010
- Asuma que la entrada es una línea serial de 1 bit
- Utilice flip-flops D y multiplexores 8 a 1
- Comience por definir el diagrama de estados - transiciones

Diagrama de ET del detector de secuencias

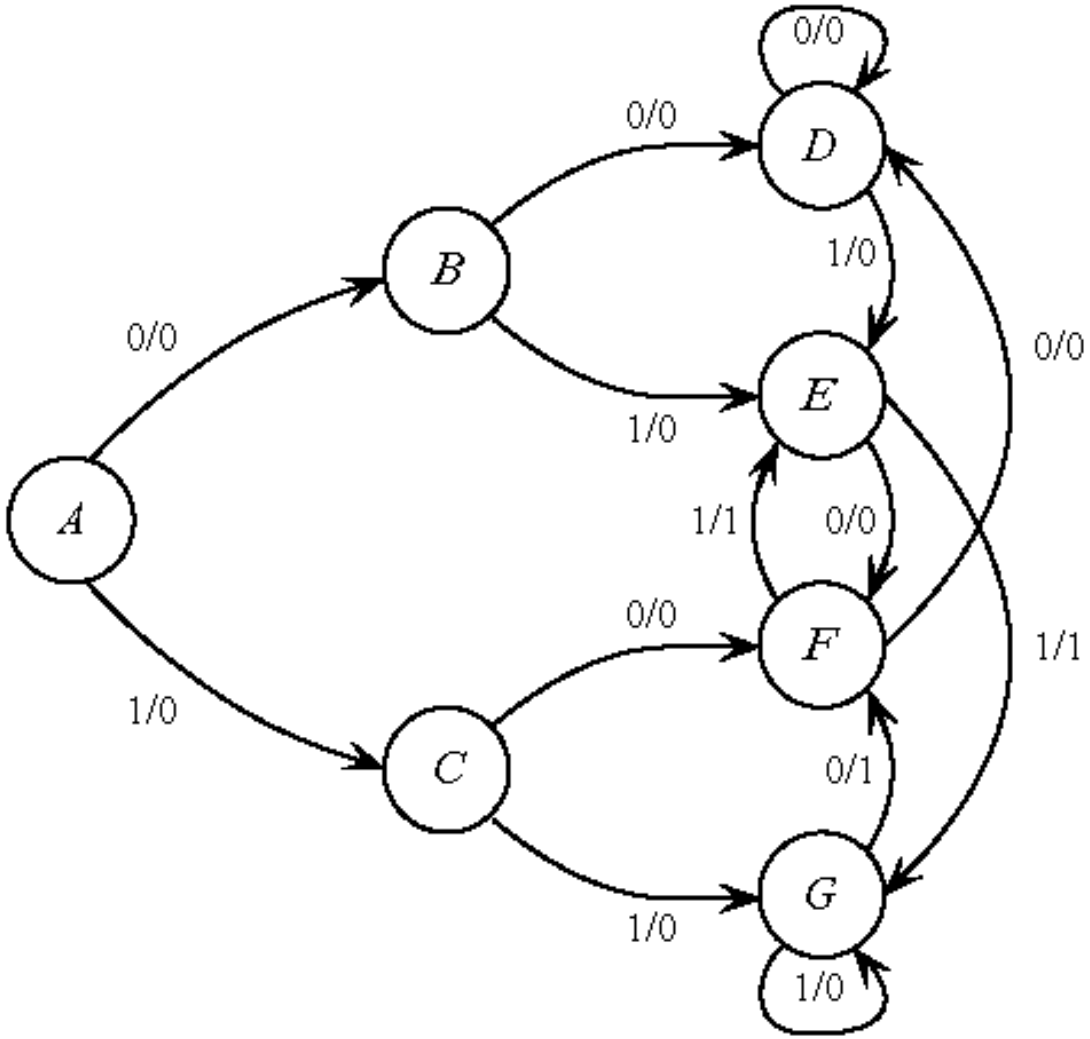
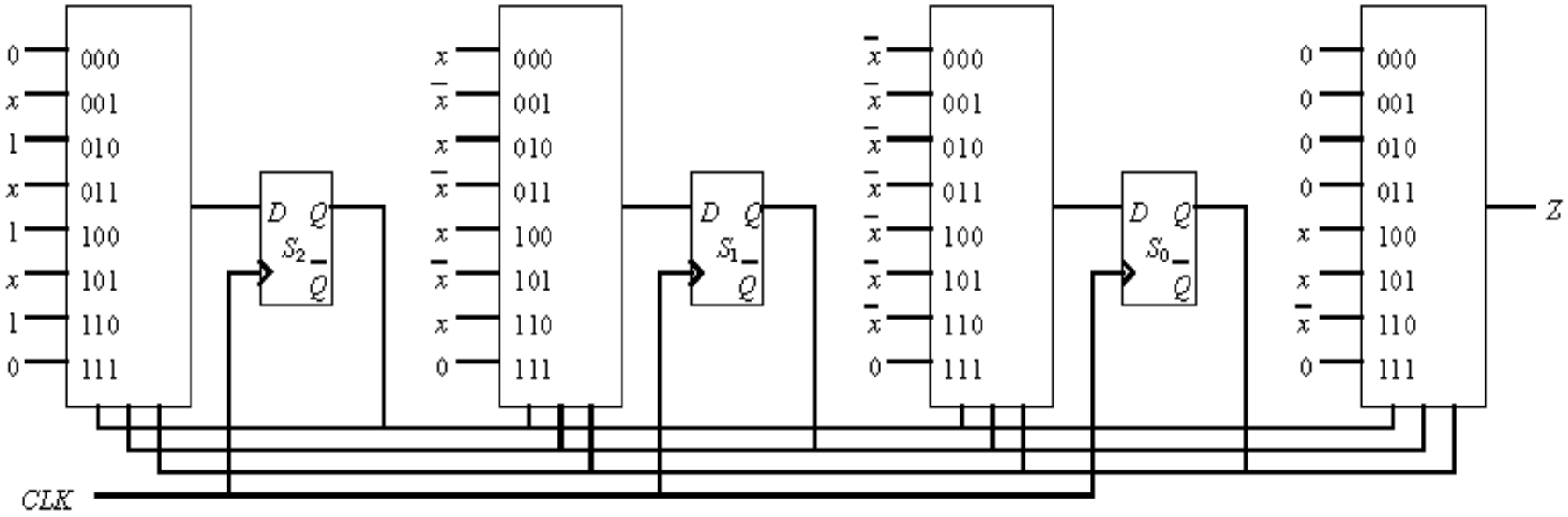
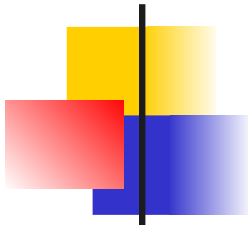


Tabla de estados del detector de secuencias

Present state \ Input	X = 0	X = 1
A	$B/0$	$C/0$
B	$D/0$	$E/0$
C	$F/0$	$G/0$
D	$D/0$	$E/0$
E	$F/0$	$G/1$
F	$D/0$	$E/1$
G	$F/1$	$G/0$

Diagrama lógico del detector de secuencias





Ejemplo: un expendedor automático

- Diseñe una máquina de estado finito para un expendedor automático que acepta monedas de 5, 10 y 25 centavos. Cuando la cantidad de dinero insertado es mayor o igual que 20 centavos, la máquina expende el ítem, da el vuelto (si corresponde), y queda a la espera de una nueva transacción
- Implemente con PLA y flip-flops D

Diagrama de ET del expendedor automático

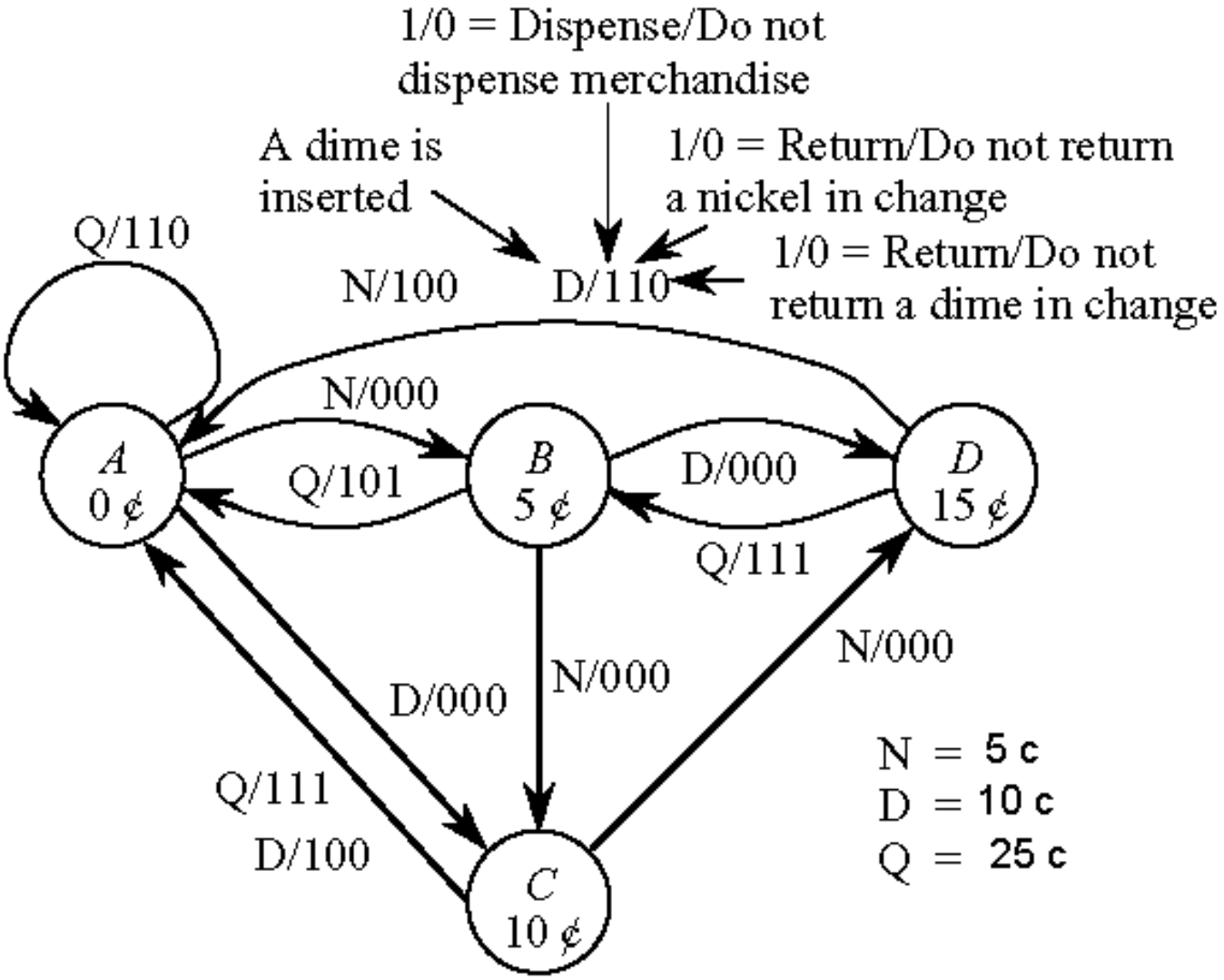


Tabla de estados y asignación de estados del expendedor automático

Input / P.S.	N 00	D 01	Q 10
A	*B*/000	*C*/000	*A*/110
B	*C*/000	*D*/000	*A*/101
C	*D*/000	*A*/100	*A*/111
D	*A*/100	*A*/110	*B*/111

(a)

Input / P.S.	N x_1x_0 00	D x_1x_0 01	Q x_1x_0 10
s_1s_0	s_1s_0 / $z_2z_1z_0$		
A:00	01/000	10/000	00/110
B:01	10/000	11/000	00/101
C:10	11/000	00/100	00/111
D:11	00/100	00/110	01/111

(b)

Diagrama lógico del expendedor automático

Base 10 equivalent	Present state S_1 S_0	Coin X_1 X_0	Next state S_1 S_0	Dime-out Z_2	Return nickel Z_1	Return dime Z_0
0	0 0	0 0	0 1	0	0	0
1	0 0	0 1	1 0	0	0	0
2	0 0	1 0	0 0	1	1	0
3	0 0	1 1	d d	d	d	d
4	0 1	0 0	1 0	0	0	0
5	0 1	0 1	1 1	0	0	0
6	0 1	1 0	0 0	1	0	1
7	0 1	1 1	d d	d	d	d
8	1 0	0 0	1 1	0	0	0
9	1 0	0 1	0 0	1	0	0
10	1 0	1 0	0 0	1	1	1
11	1 0	1 1	d d	d	d	d
12	1 1	0 0	0 0	1	0	0
13	1 1	0 1	0 0	1	1	0
14	1 1	1 0	0 1	1	1	1
15	1 1	1 1	d d	d	d	d

(b)

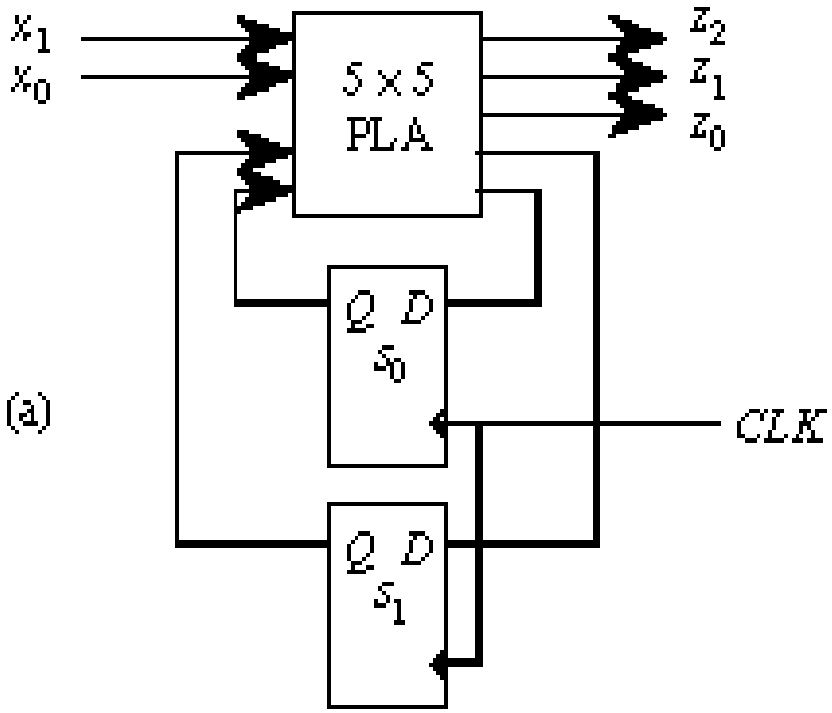
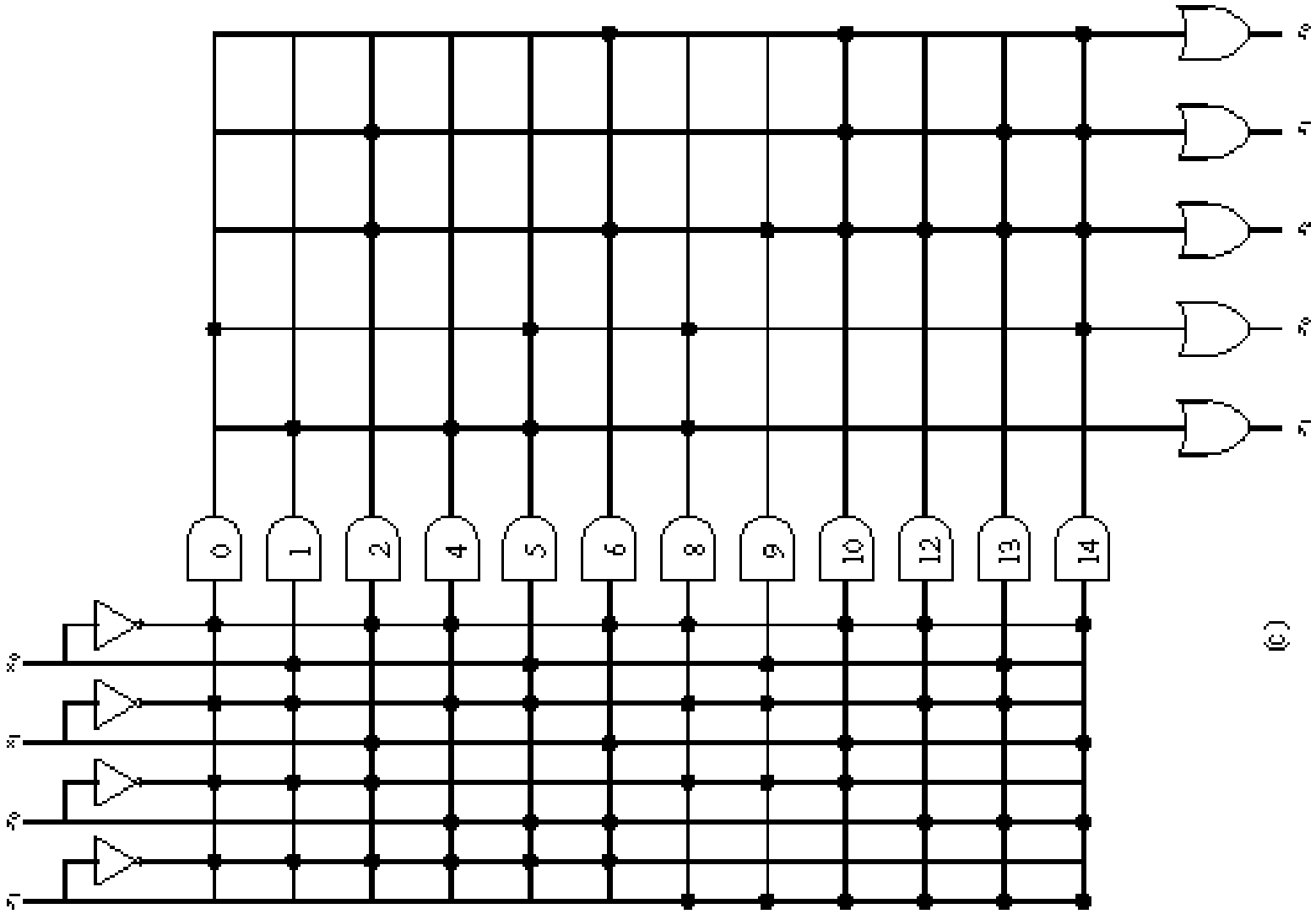
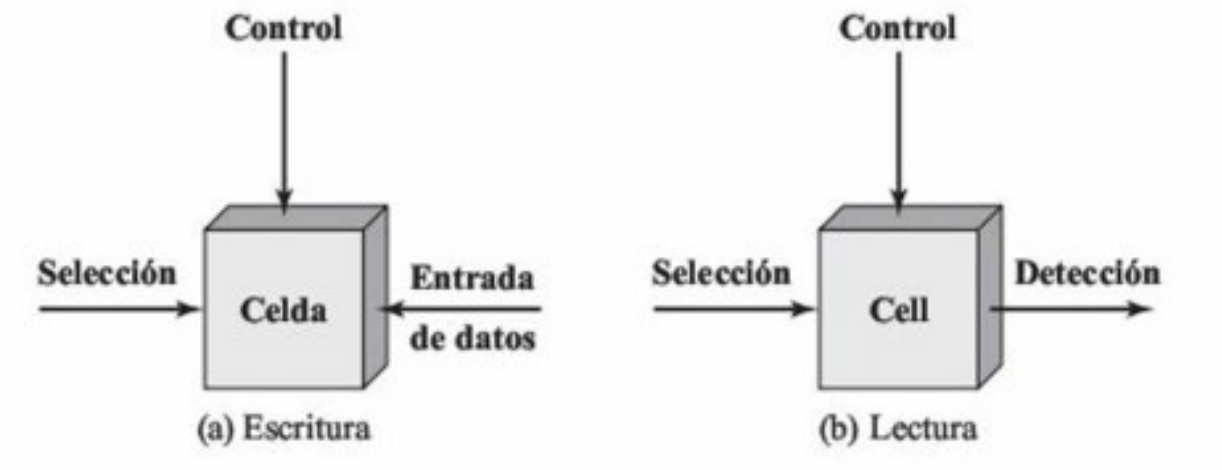


Diagrama lógico del expendedor automático (PLA)

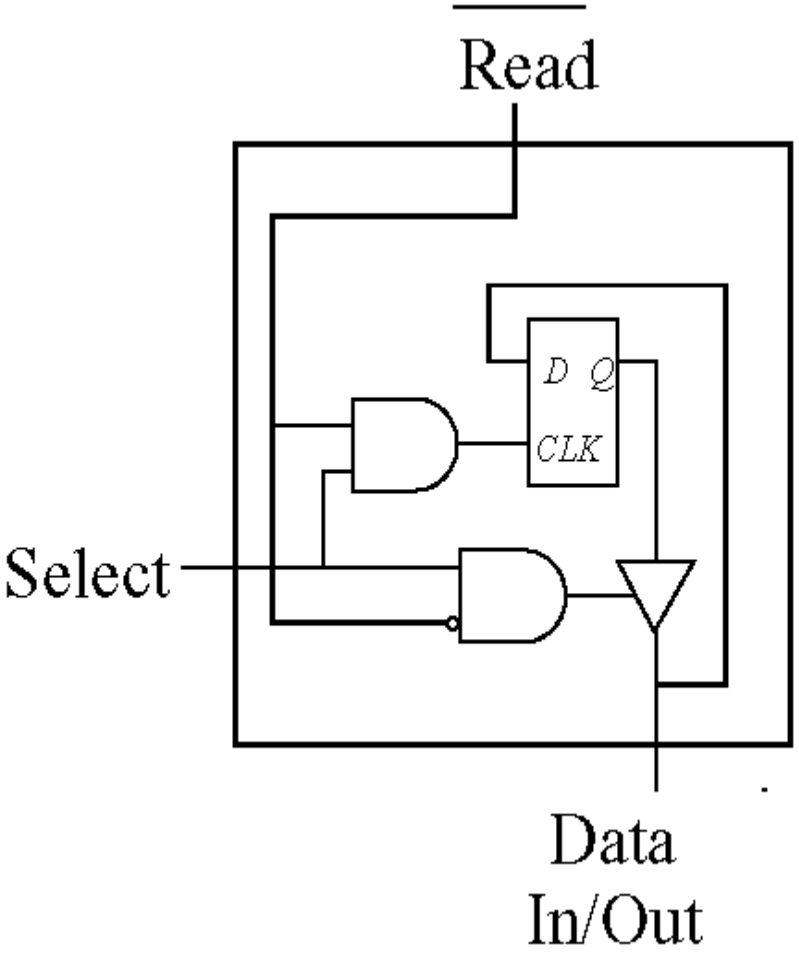


Comportamiento funcional de una celda de memoria

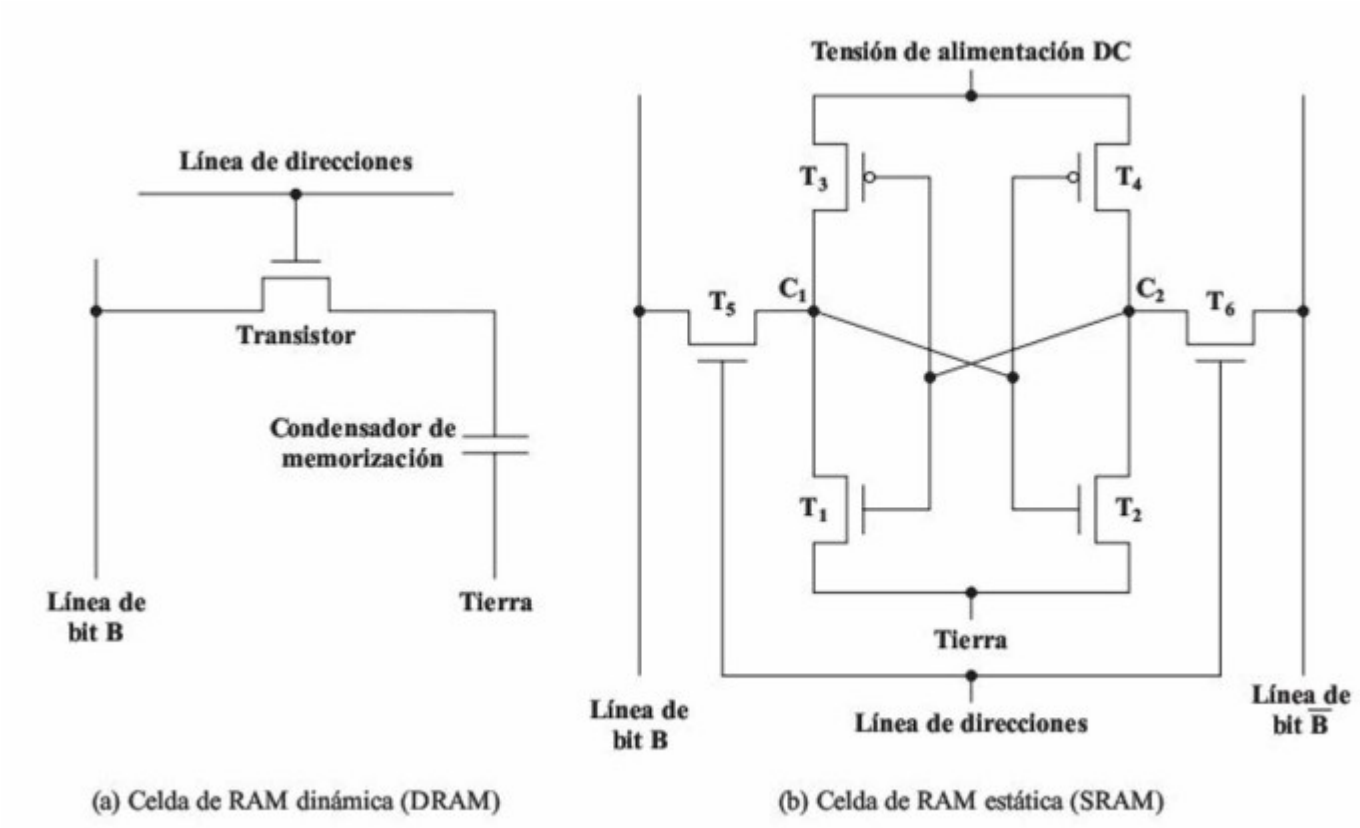


Funcionamiento de una celda de memoria

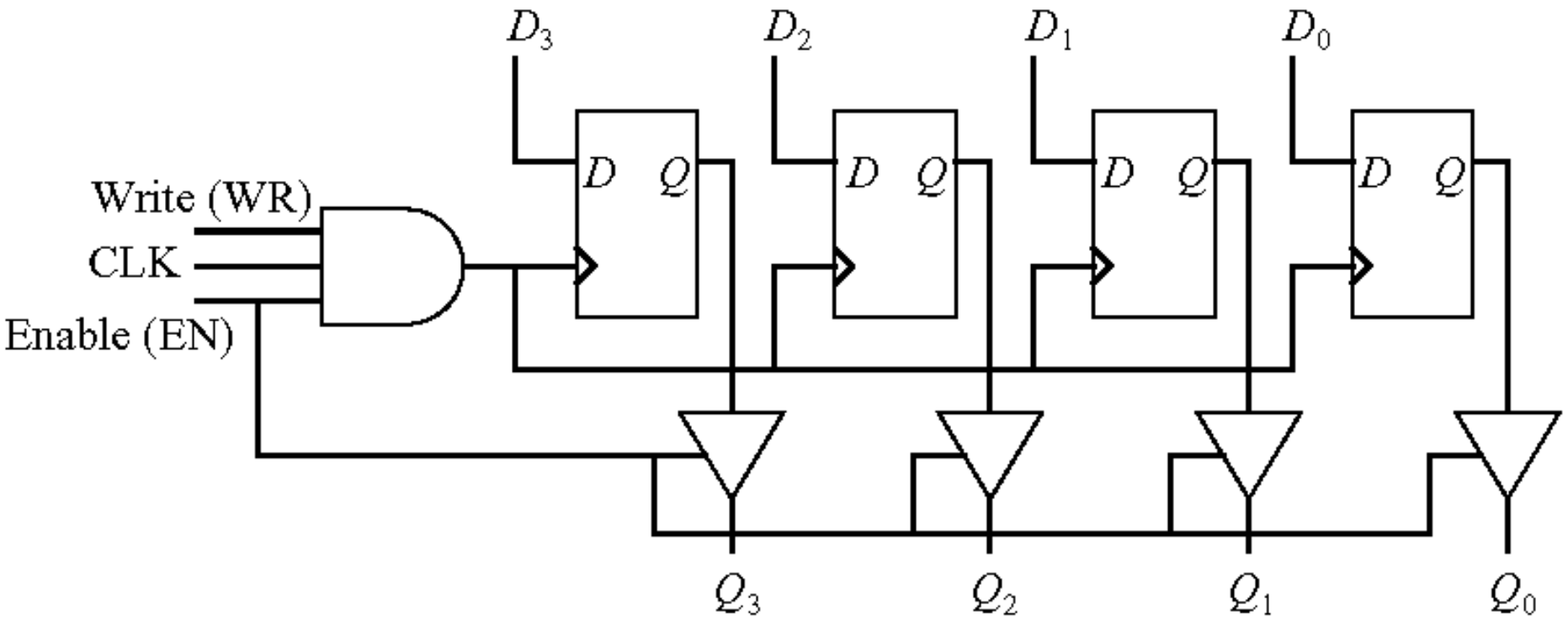
Comportamiento funcional de una celda de memoria



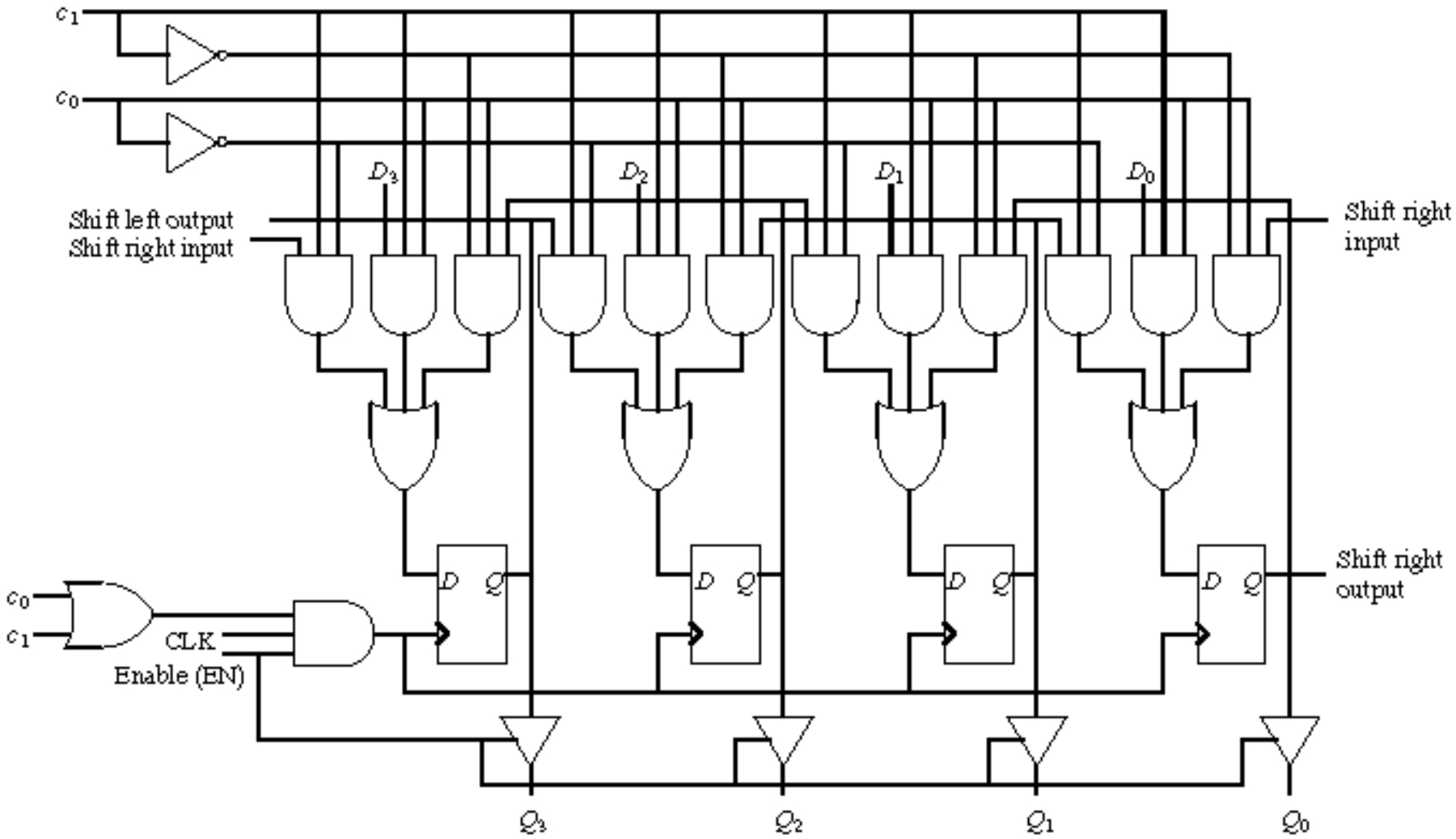
Comportamiento funcional de una celda de memoria



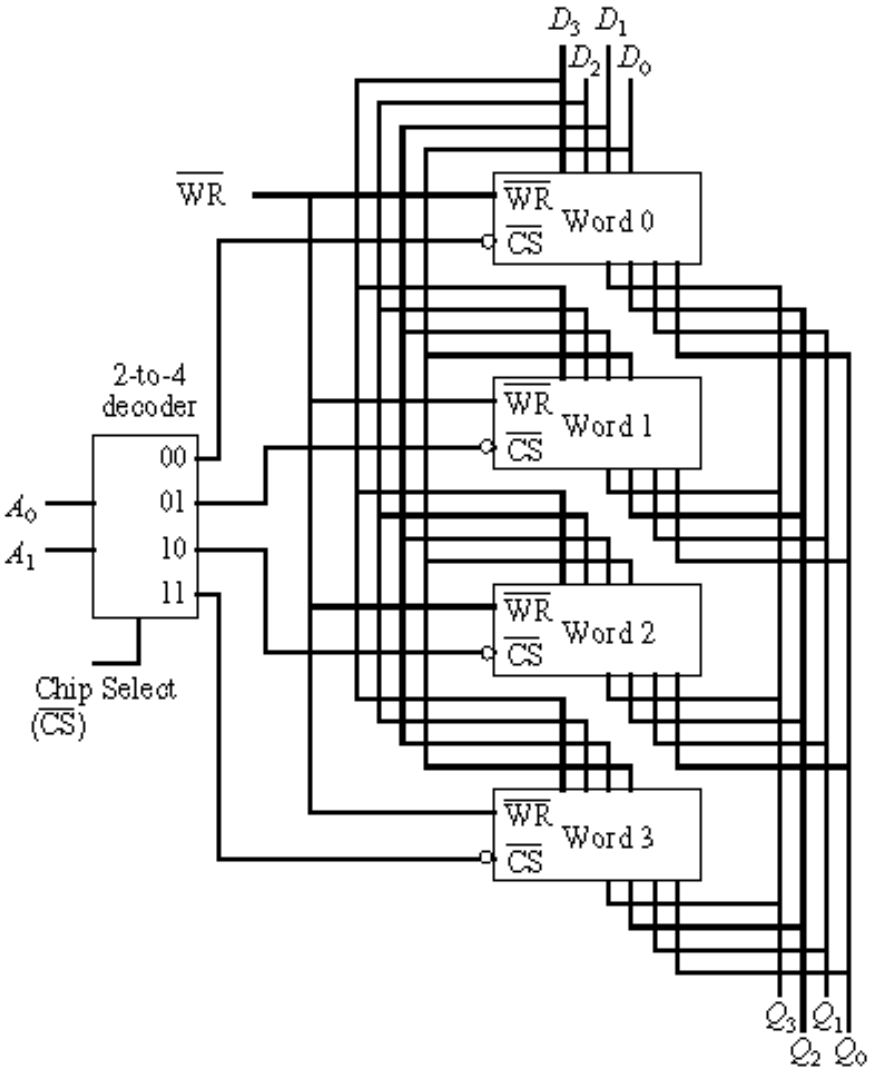
Un registro de 4 bits



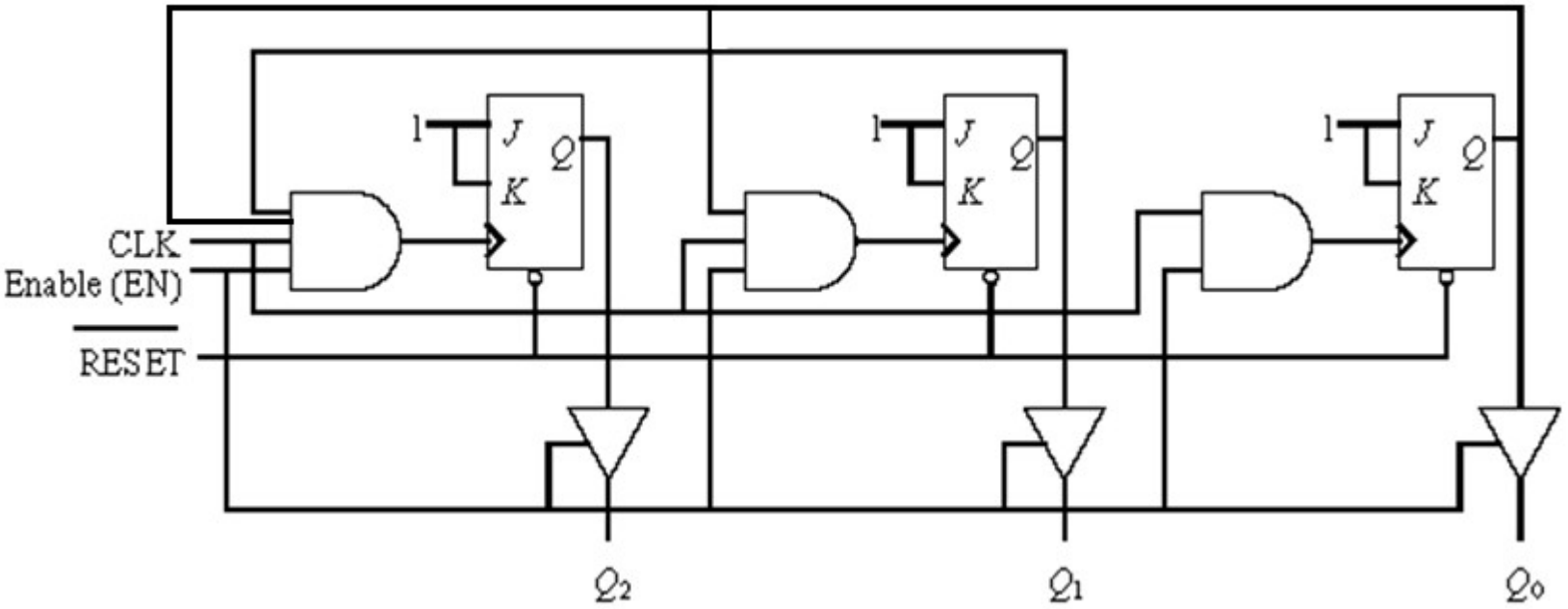
Un registro de desplazamiento izquierda – derecha con lectura y escritura paralela

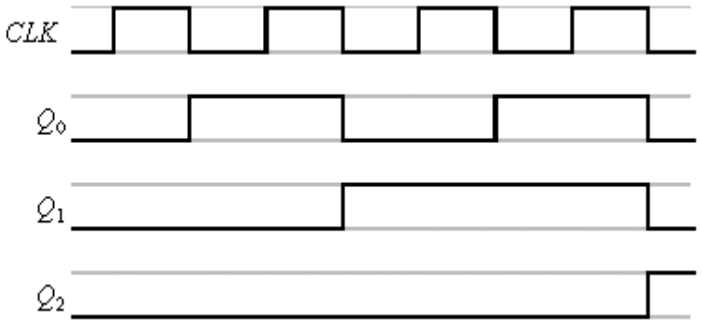


Una memoria con 4 palabras de 4 bits

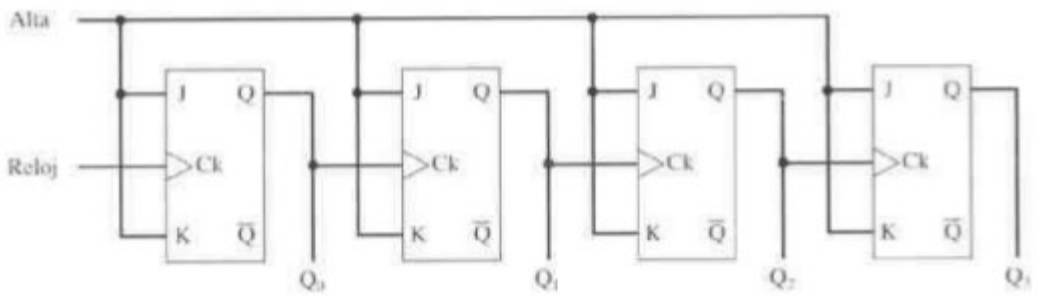


Un contador módulo 8 sincrónico

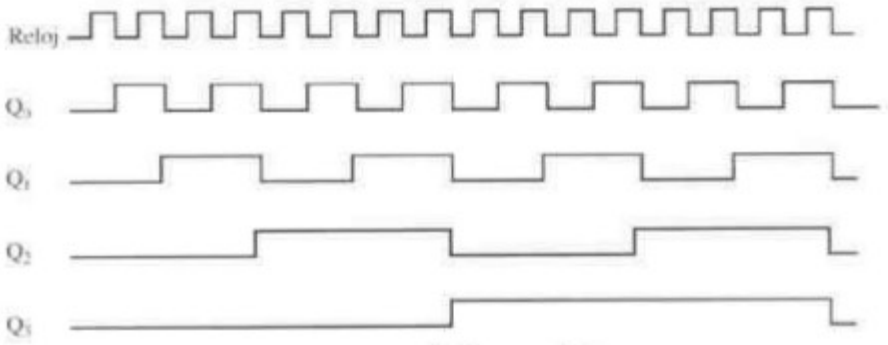




Un contador módulo 8 asincrónico



(a) Circuito secuencial



(b) Diagrama de tiempo