Capítulo III
Circuitos Secuenciales

Cristián Tejos
Primer Semestre 2017

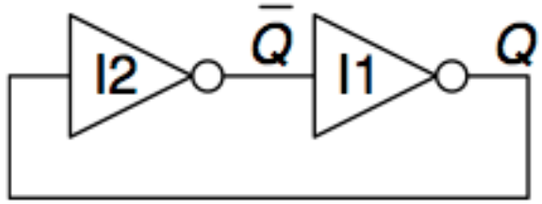
Basado en apuntes de Marcelo Guarini
y el libro
Digital Design and Computer Architecture (1st edition),
D. M. Harris & S. L. Harris, Elsevier 2007

Introducción

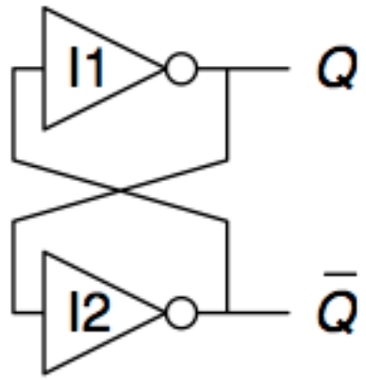
- La salida de un circuito combinacional depende de la entrada presente.
- En los circuitos secuenciales, la salida depende de la entrada presente y de las entradas previas (tienen memoria).
- Pueden explicitamente recordar algunas de las entradas pasadas, o destilar las entradas previas en una cantidad menor de información llamada estado del sistema
- El estado de un circuito secuencial es un conjunto de bits llamadas variables de estado que contienen toda la información acerca del pasado necesaria para explicar el comportamiento futuro del circuito.

“Latches” y “Flip-Flops”

- El bloque fundamental de una memoria es un elemento bi-estable, es decir un elemento con dos estados estables.
- La figura (a) muestra un simple elemento bi-estable construido con dos negadores conectados en loop.
- La figura (b) muestra el mismo circuito redibujado para enfatizar su simetría.
- Tiene dos salidas pero no tiene entradas



(a)

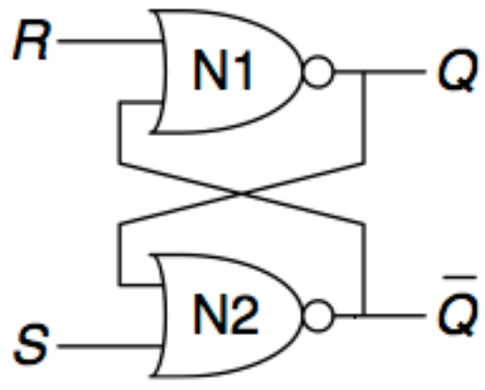


(b)

“Latches” y “Flip-Flops”

- Operación del elemento bi-estable

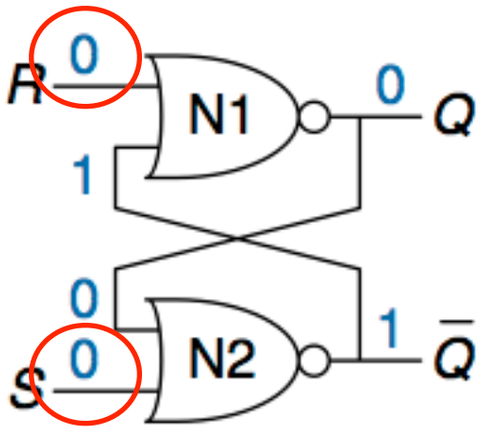
- ¿Cómo poder ingresar información [Latch S-R]?

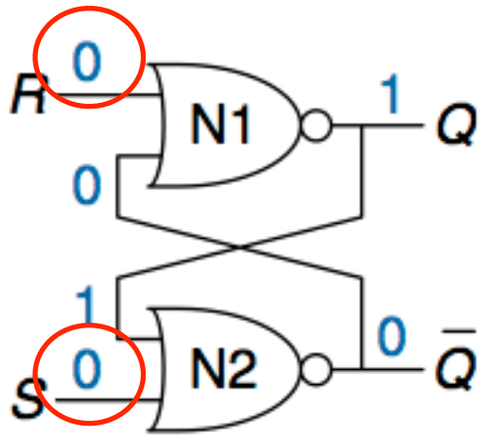


“Latches” y “Flip-Flops”

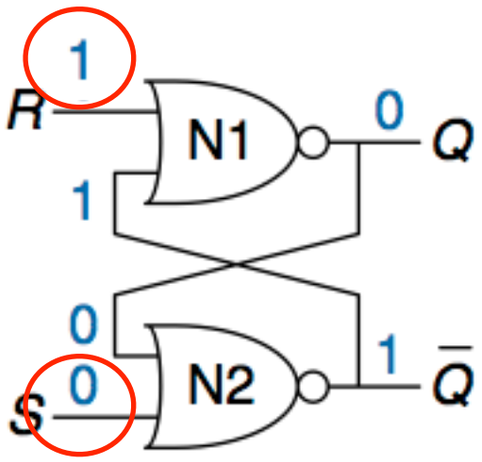
Operación del Latch S-R

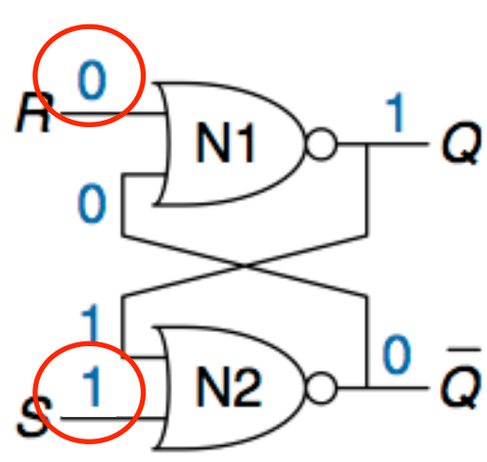
- Memorización





- Ingreso de información

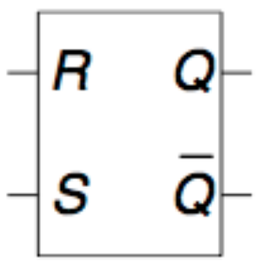




“Latches” y “Flip-Flops”

Tabla de operación y símbolo del Latch S-R

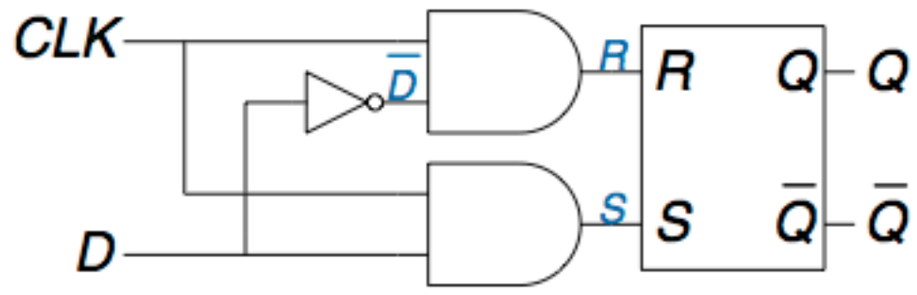
S	R	Q	$\bar{Q}$	
0	0	Q_{prev}	$\bar{Q}_{prev}$	
0	1	0	1	
1	0	1	0	
1	1	0	0	← En general esta combinación no se permite



“Latches” y “Flip-Flops”

Latch tipo D

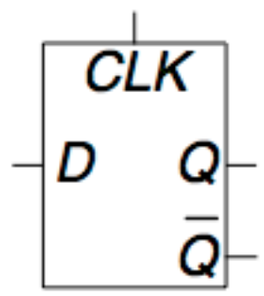
- Circuito



- Tabla de operación

CLK	D	$\overline{D}$	S	R	Q	$\overline{Q}$
0	X	$\overline{X}$	0	0	Q_{prev}	$\overline{Q}_{prev}$
1	0	1	0	1	0	1
1	1	0	1	0	1	0

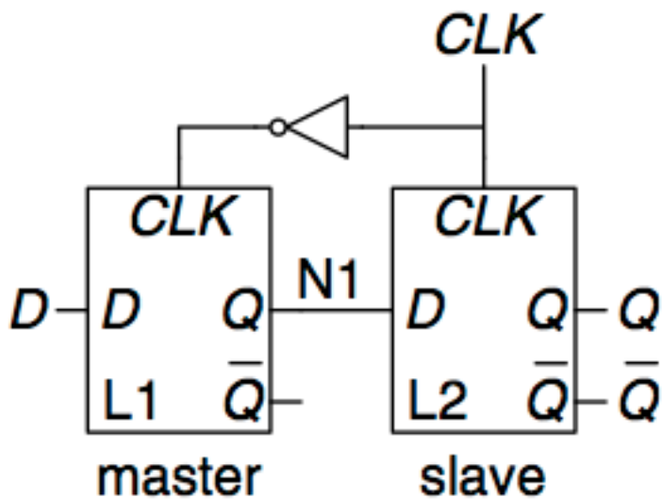
- Símbolo



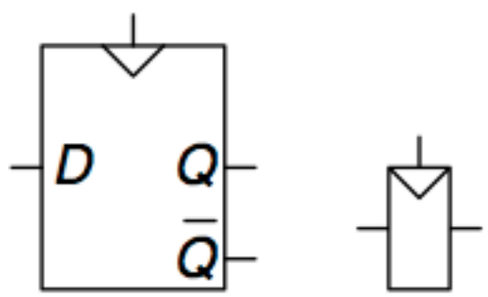
“Latches” y “Flip-Flops”

Flip-Flop tipo D

- Se fabrica utilizando dos latches tipo D controlados por señales de reloj (clock) complementarias.



- Símbolo y símbolo condensado



“Latches” y “Flip-Flops”

- PREGUNTA: ¿Cuántos transistores se requieren para hacer un flip-flop D?

“Latches” y “Flip-Flops”

Registros

- Un registro de N bits es un banco de N flip-flops D que comparten un clock común.

Registro de 4 bits

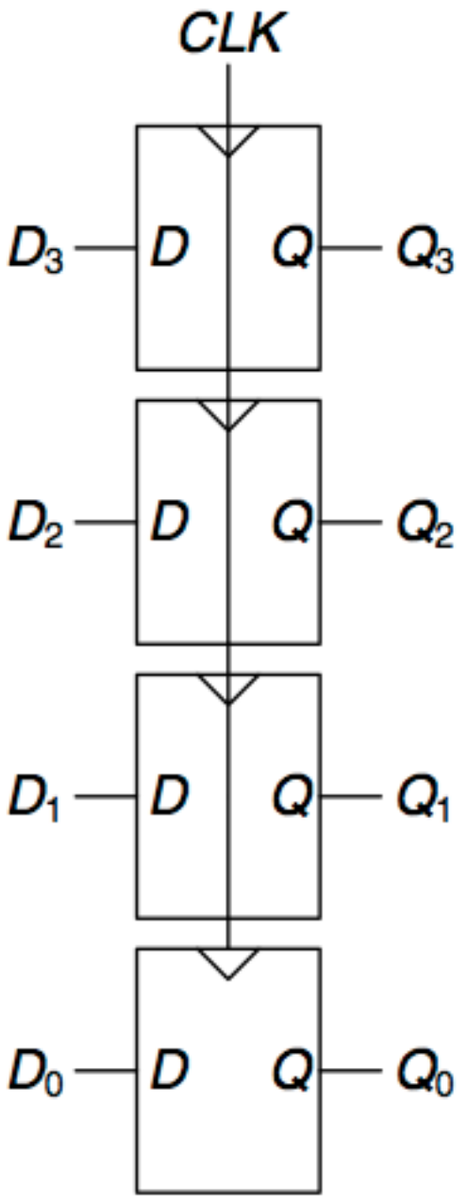
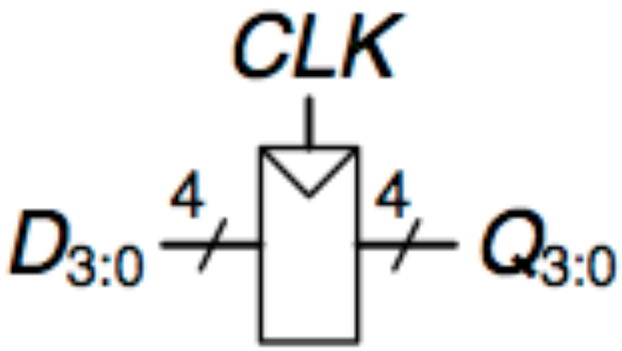


Diagrama condensado

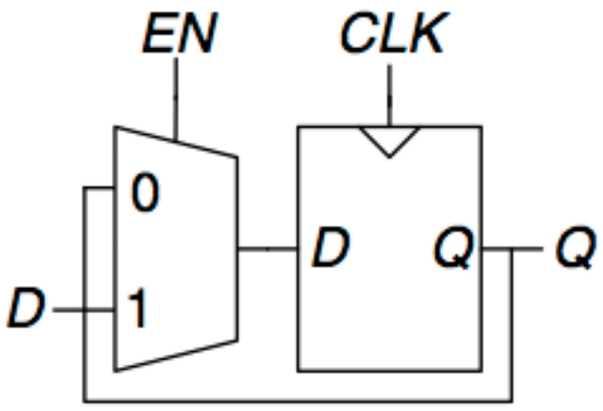


“Latches” y “Flip-Flops”

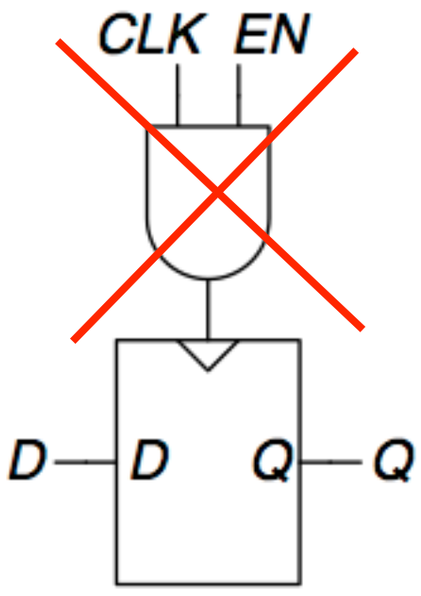
Flip-Flop con habilitación (Enable)

- Agrega una entrada adicional *EN* (ENABLE) que determina si se carga o no el dato en la transición de la señal del reloj

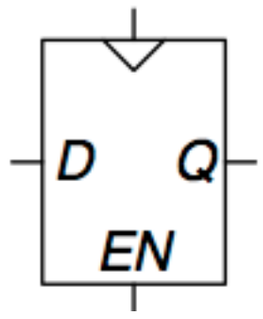
Circuito



Otra forma



Símbolo

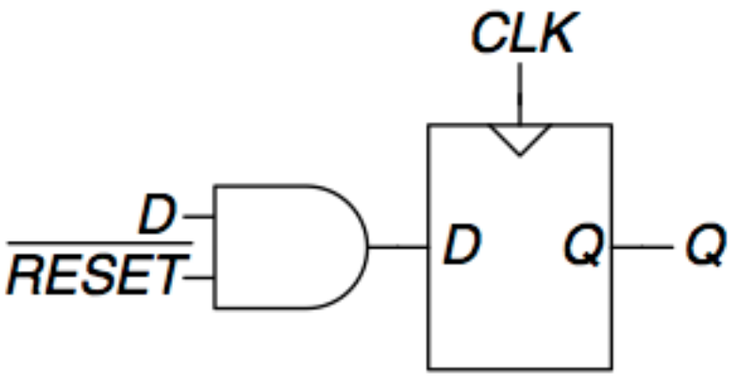


“Latches” y “Flip-Flops”

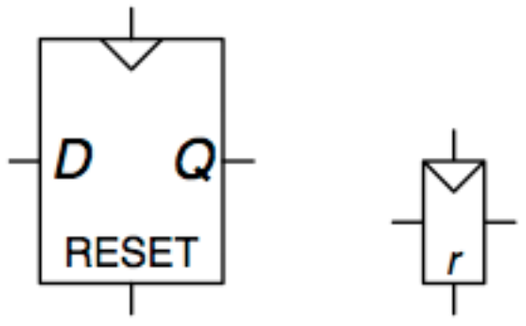
Flip-Flop D con Reset

- Agrega una entrada adicional llamada *RESET*. Cuando ésta es 0, el flip-flop funciona normalmente. Cuando es 1, el flip-flop se pone en cero con el primer flanco positivo del reloj. (reset síncrono)

Circuito



Símbolo

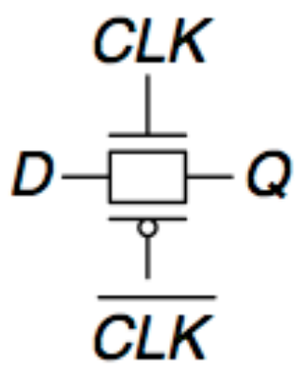


“Latches” y “Flip-Flops”

Diseño de Latches y Flip-Flops a nivel de transistores

La función de un Latch es ser transparente u opaco, más o menos como un interruptor

Un compuerta de transmisión es una forma eficiente de construir un interruptor.



Cuando *CLK* = 1, la compuerta de transmisión está ON, entonces *D* fluye a Q y decimos que el *Latch* es transparente.

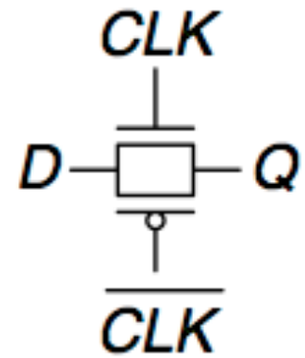
Cuando *CLK* = 0, la compuerta de transmisión está OFF, entonces Q está aislado de D y decimos que el *Latch* está opaco

“Latches” y “Flip-Flops”

Diseño de Latches y Flip-Flops a nivel de transistores

Este simple *Latch D* sufre de dos limitaciones importantes:

Nodo de salida flotante: Cuando el *Latch* está opaco, Q no es mantenido en su nivel por ninguna compuerta. Se dice que Q es un nodo flotante o dinámico. Después de algún tiempo, el ruido y la fuga de cargas puede perturbar el valor de Q.



No tiene *buffers*: La falta de *buffers* causó el mal funcionamiento de varios circuitos comerciales. Un peak de ruido que mueva D a un valor negativo puede encender el transistor nMOS haciendo transparente el circuito, aún cuando el *clock* = 0.
De la misma forma un peak de ruido positivo, > VDD, puede encender el transistor pMOS, aún cuando *clock* = 0.

“Latches” y “Flip-Flops”

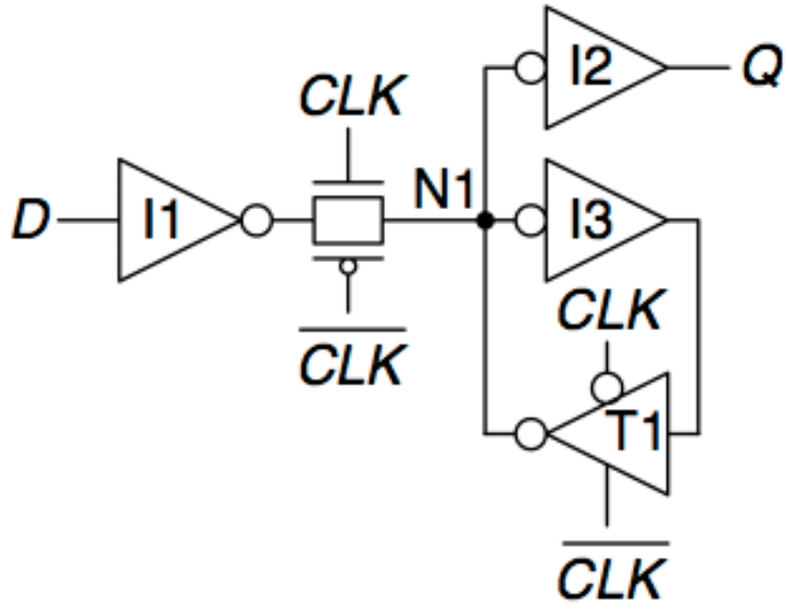
Diseño de Latches y Flip-Flops a nivel de transistores

Latch D

Este circuito es más robusto (12 transistores) y hoy se utiliza comunmente en los circuitos comerciales como *Latch* D. Utiliza I1 e I2 para aislar la entrada y la salida respectivamente.

El estado del *Latch* es mantenido en el nodo N1. El inversor I3 y el *Buffer tristate* T1 proveen la realimentación para hacer que N1 sea un nodo estático.

Si una pequeña cantidad de ruido ocurre en N1 cuando el *CLK* = 0, T1 se encarga de restaurar el valor correcto válido.



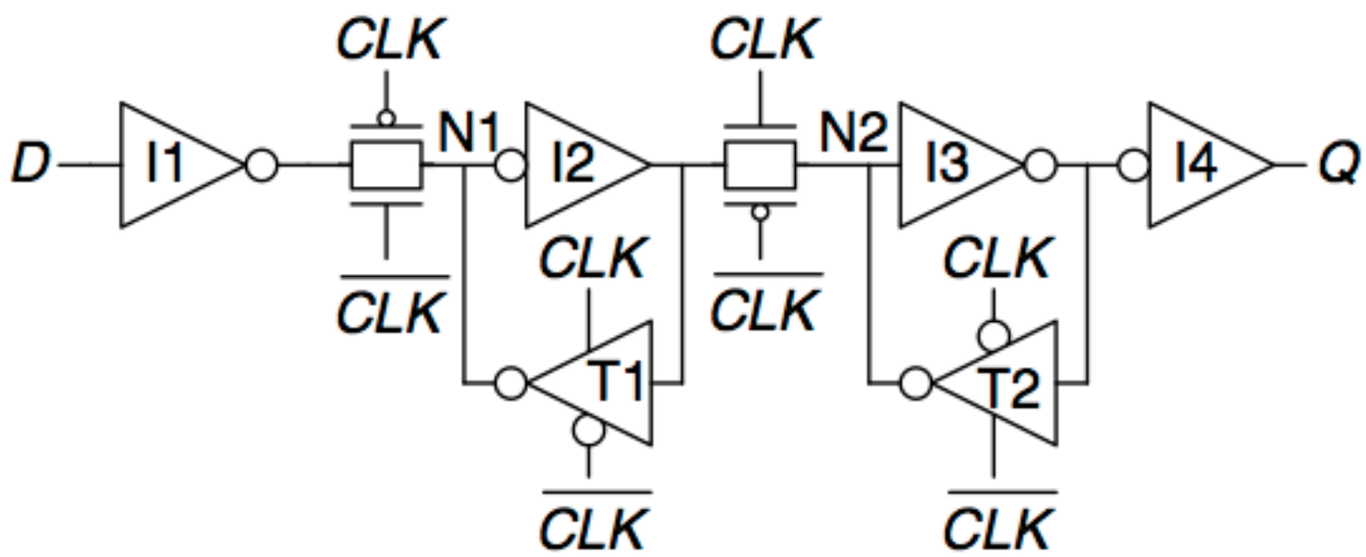
“Latches” y “Flip-Flops”

Diseño de Latches y Flip-Flops a nivel de transistores

Flip-Flop D

Construido utilizando dos *Latches* estáticos.

Algunos inversores internos redundantes se han eliminado para que tenga sólo 20 transistores



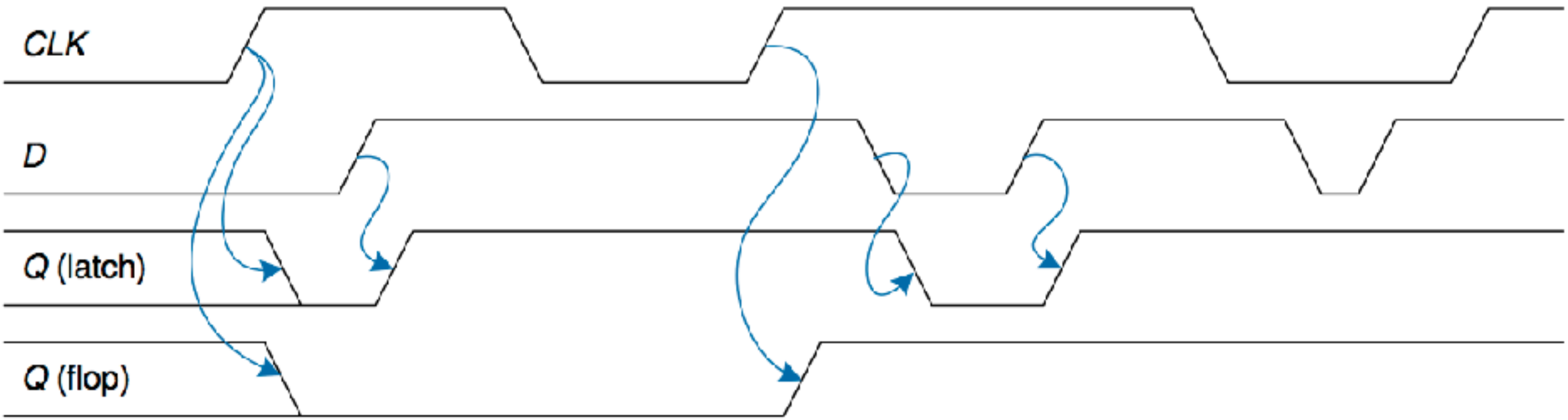
“Latches” y “Flip-Flops”

Lo que hay que recordar

- Los *latches* y *flip-flops* son los bloques escenciales para construír circuitos secuenciales.
- Los *latches* son sensibles al nivel del *CLK*, mientras que los *flip-flops* son gatillados por el flanco.
- El *latch* D es transparente con el *CLK* = 1, dejando fluir *D* hasta *Q*.
- El *flip-flop* D copia el valor de *D* en *Q* al momento del flanco de subida del *CLK*.
- En todo otro momento, los *latches* y los *flip-flops* retienen su estado anterior.
- Un registro es un banco de *flip-flops* D que comparten una misma señal de *CLK*.

“Latches” y “Flip-Flops”

Diferencia entre un *latch* D y un *flip-flop* D



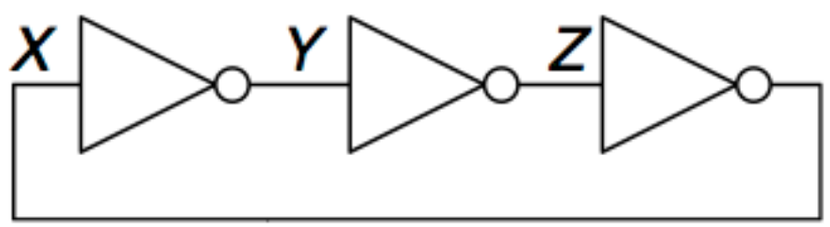
Diseño de circuitos secuenciales

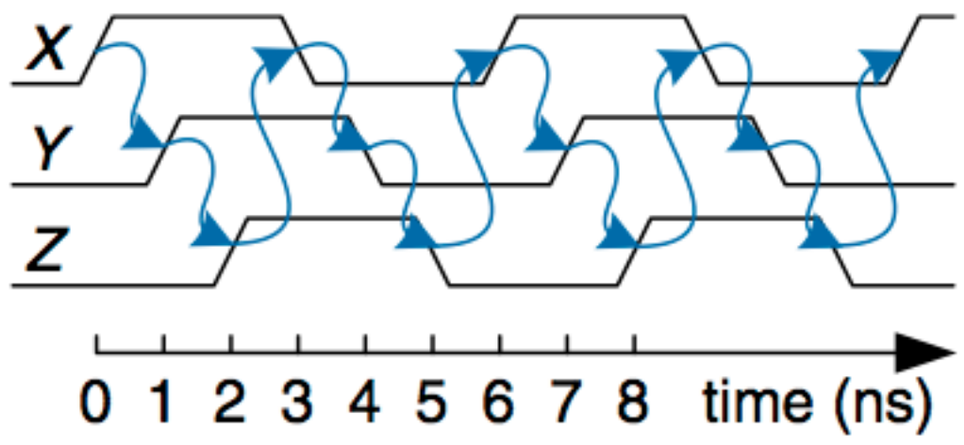
- En general los circuitos secuenciales son aquellos para los cuales la salida no puede ser establecida en base sólo a las variables de entrada.
- Algunos circuitos secuenciales son curiosos y presentan problemas.
- En esta sección introduciremos el concepto de circuito secuencial síncrono.
- Si nos concentramos en circuitos secuenciales síncronos, podremos desarrollar formas fáciles y sistemáticas para analizar y diseñar circuitos secuenciales confiables.

Diseño de circuitos secuenciales

Un par de circuitos problemáticos:

- El oscilador de anillo





Su desempeño depende de muchos factores y en general es muy poco confiable.

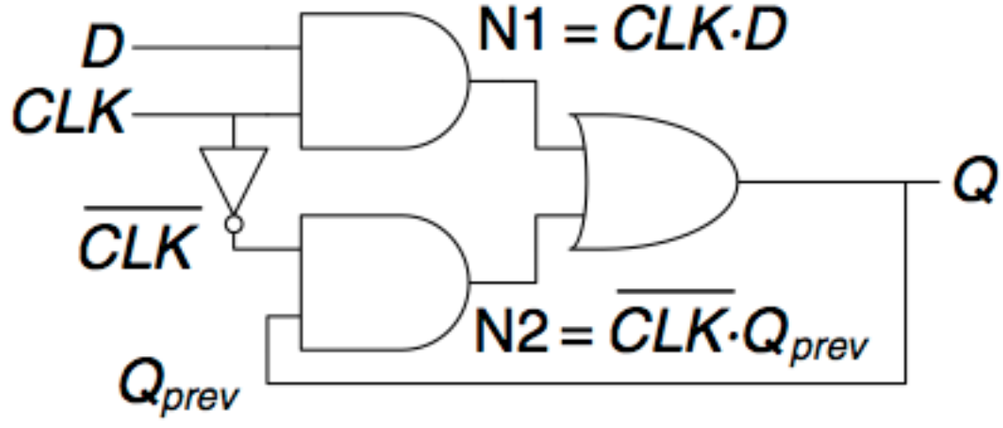
Diseño de circuitos secuenciales

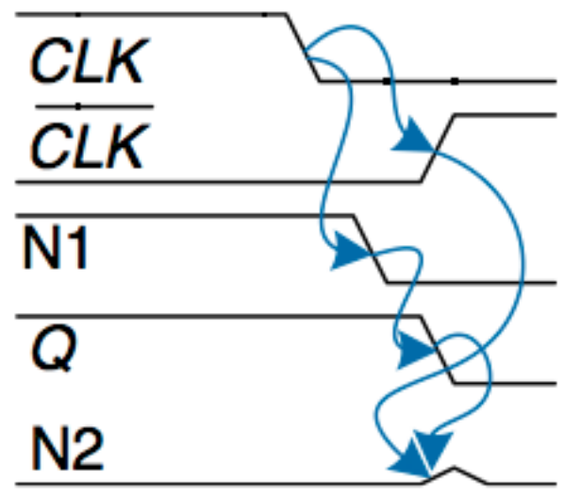
Un par de circuitos problemáticos:

- Simple *latch* D

CLK	D	Q_{prev}	Q
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$$Q = CLK \cdot D + \overline{CLK} \cdot Q_{prev}$$





Desempeño poco confiable. Muy dependiente de los retardos de las compuertas. puede sufrir de *race condition*, como lo muestra el diagrama de tiempos.

En general estas fallas son muy difíciles de detectar.

Diseño de circuitos secuenciales

Circuitos secuenciales síncronos

- Los dos circuitos previos contienen loops llamados caminos cíclicos, en que salidas alimentan directamente a entradas.
- Circuitos que contienen caminos cíclicos pueden tener comportamiento inestable y su análisis es en general muy complejo.
- Para evitar estos problemas los diseñadores rompen el camino cíclico, insertando registros en ciertas partes del camino, transformando el circuito en un conjunto de circuitos combinacionales y registros.
- Los registros contienen el "estado del sistema", que cambia sólo con los flancos del reloj, por lo que se dice que el estado está sincronizado con la señal de reloj.

Diseño de circuitos secuenciales

Circuitos secuenciales síncronos

- Si el *CLK* es suficientemente lento, de tal forma que las entradas a los registros están estables antes del próximo flanco de reloj, todas las *race conditions* se eliminan.

- Adoptando la disciplina de siempre utilizar registros en el camino de realimentación nos lleva a la definición de circuito secuencial síncrono.

- Recordemos que un circuito está definido por sus entradas, sus salidas, su función y sus especificaciones de tiempo.

- Un circuito secuencial tiene un conjunto finito de estados discretos $\{S_0, S_1, \ldots, S_k\}$.

Diseño de circuitos secuenciales

Circuitos secuenciales síncronos

- Un circuito secuencial síncrono tiene una entrada de *CLK* cuyos flancos de subida definen la secuencia de tiempos en que ocurren las transiciones.
- A menudo se usa el término estado presente y próximo estado para distinguir el estado actual de aquel al que se llegará inmediatamente después del próximo flanco de subida.
- La especificación funcional detalla el próximo estado y el valor de cada salida para cada posible valor del estado presente y de las entradas.
- Las especificaciones de tiempo incluyen un límite superior, t_{pcq}, y un límite inferior, t_{ccq}, para el tiempo desde el flanco de subida del *CLK* hasta que la salida cambia.

Diseño de circuitos secuenciales

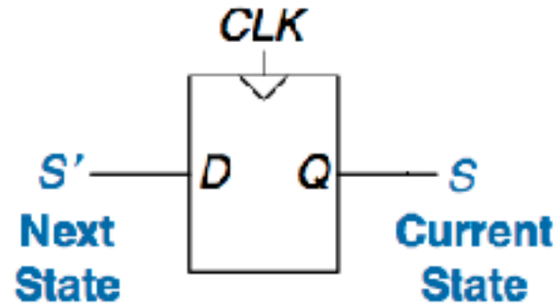
Circuitos secuenciales síncronos

- Las especificaciones de tiempo también incluyen tiempos de *hold*, t_{hold}, y setup, t_{setup}, que indican cuando las entradas deben estar estable respecto del flanco de subida del *CLK*.
- Las reglas de composición de circuitos secuenciales síncronos nos dicen que un circuito es tal si consiste de elementos interconectados como
  - Cada elemento del circuito es un registro o un circuito combinacional
  - Al menos un elemento es un registro
  - Todos los registros reciben la misma señal de *CLK*
  - Cada camino cíclico contiene al menos un registro.

Diseño de circuitos secuenciales

Circuitos secuenciales síncronos

- El circuito secuencial síncrono más simple es un *flipflop* D. Contiene una entrada, D, una señal de reloj, *CLK*, una salida, *Q*, y dos estados {0,1}.

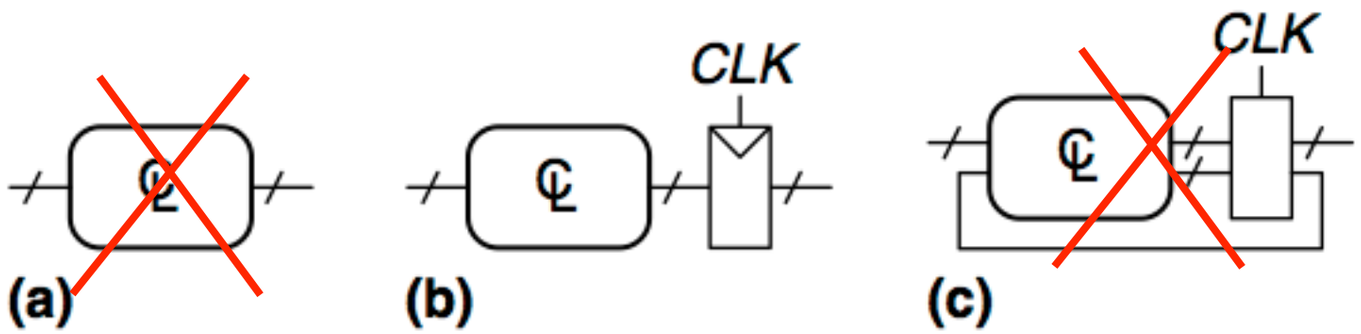


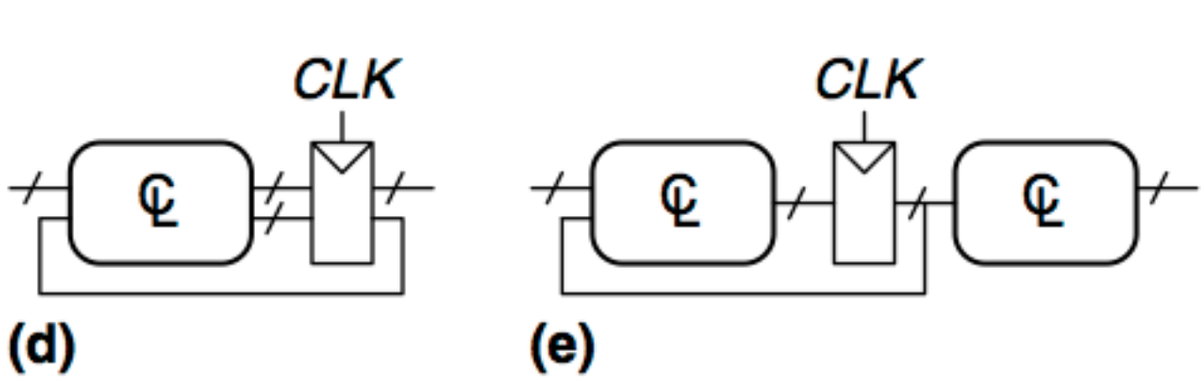
- La especificación funcional es que el próximo estado es D y que la salida Q es el estado presente.

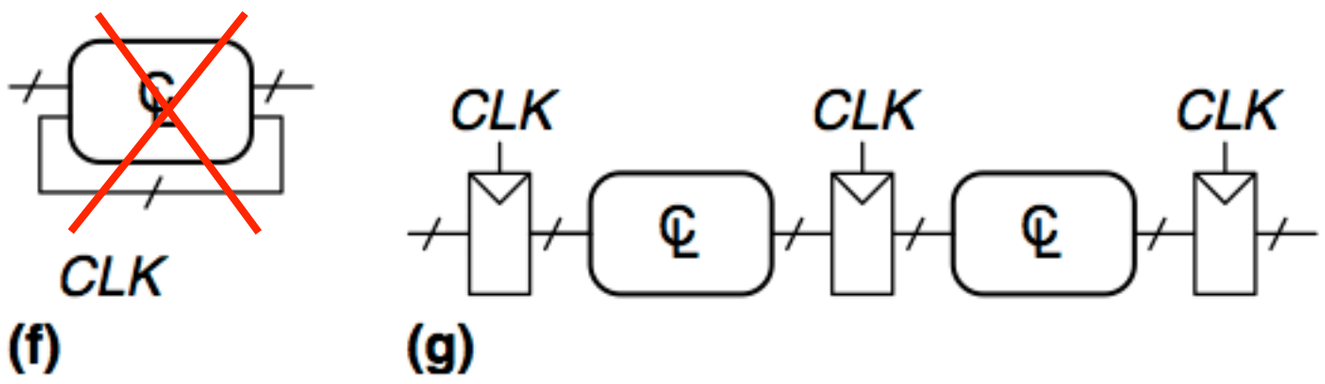
- Dos tipos comunes de máquinas secuenciales síncronas son las máquinas de estados finitas y las máquinas *pipeline*.

Diseño de circuitos secuenciales

Ejemplos de circuitos



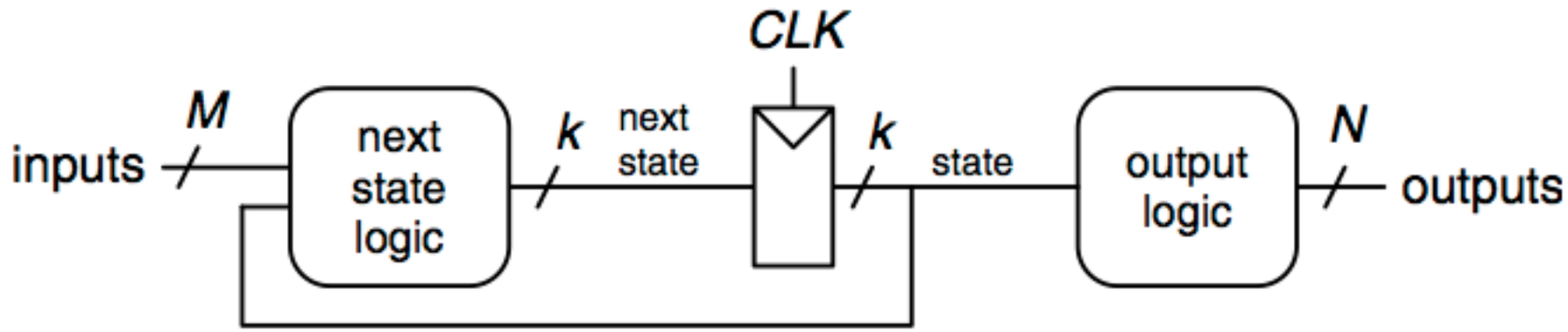




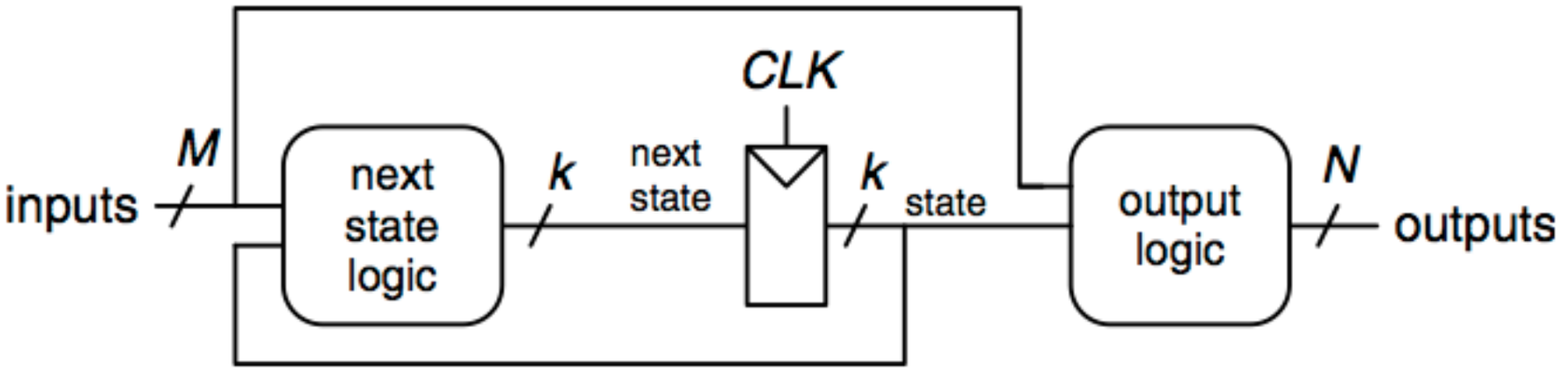
Diseño de circuitos secuenciales

Máquinas de estado finitas

- Máquina de Moore



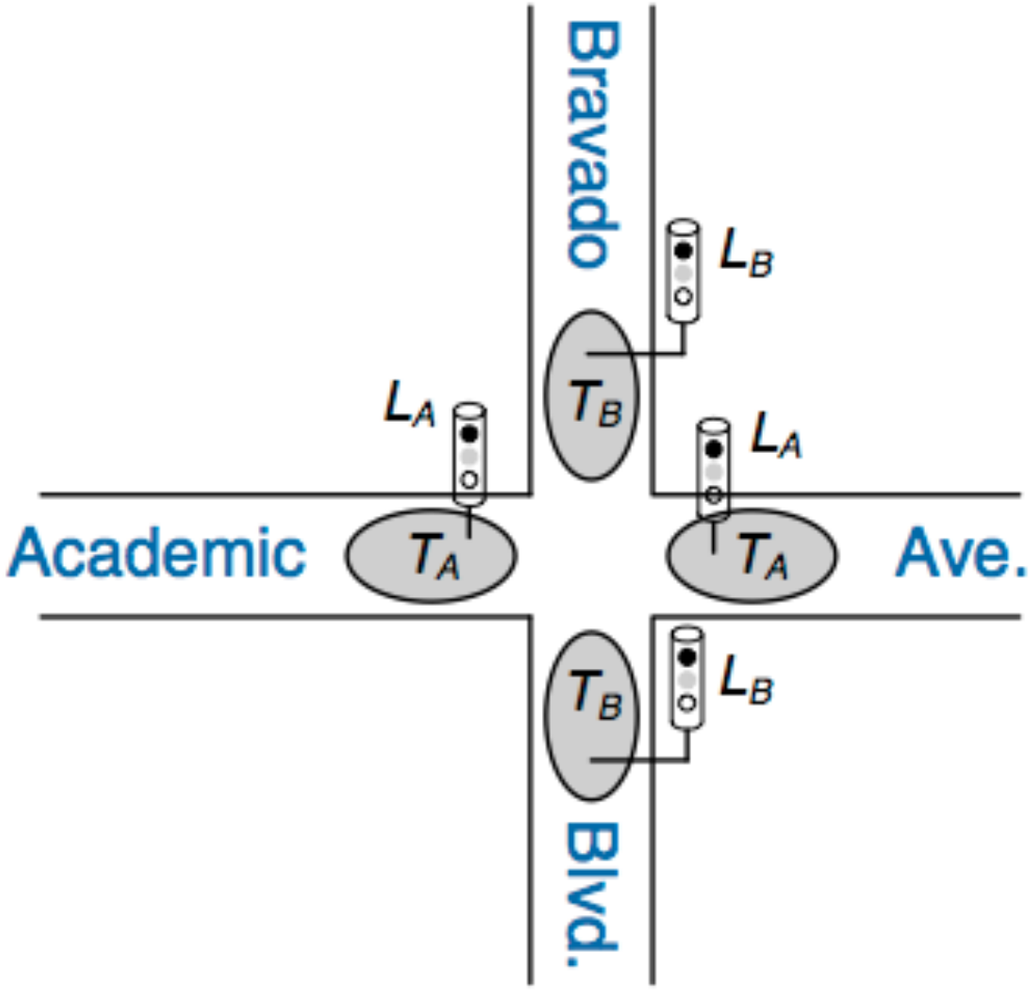
- Máquina de Mealy

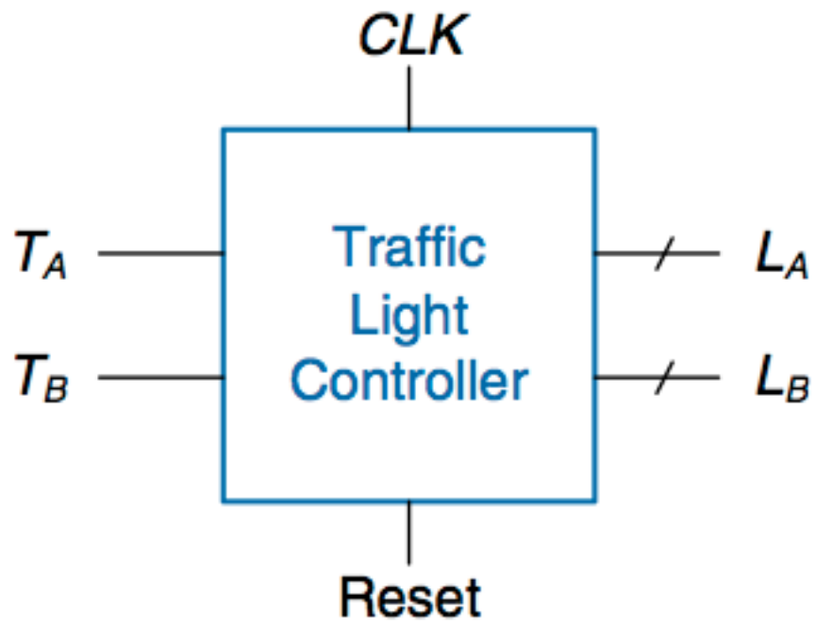


Diseño de circuitos secuenciales

Ejemplo

Simple controlador de tráfico para una intersección. T_A y T_B son loops detectores de vehículos. L_A y L_B son semáforos. La Avda. Academic tiene prioridad. Un reset inicial pone L_A en verde y L_B en rojo. L_A se mantiene en verde hasta que no se detectan autos en Avda. Academic. Cuando no hay más tráfico por Avda. Ac. L_A se pone en amarillo por 5 segundos y después en rojo mientras que L_B se pone verde. Ahora L_B se comporta de la misma forma.





Diseño de circuitos secuenciales

Ejemplo

Diagrama de transición de estados

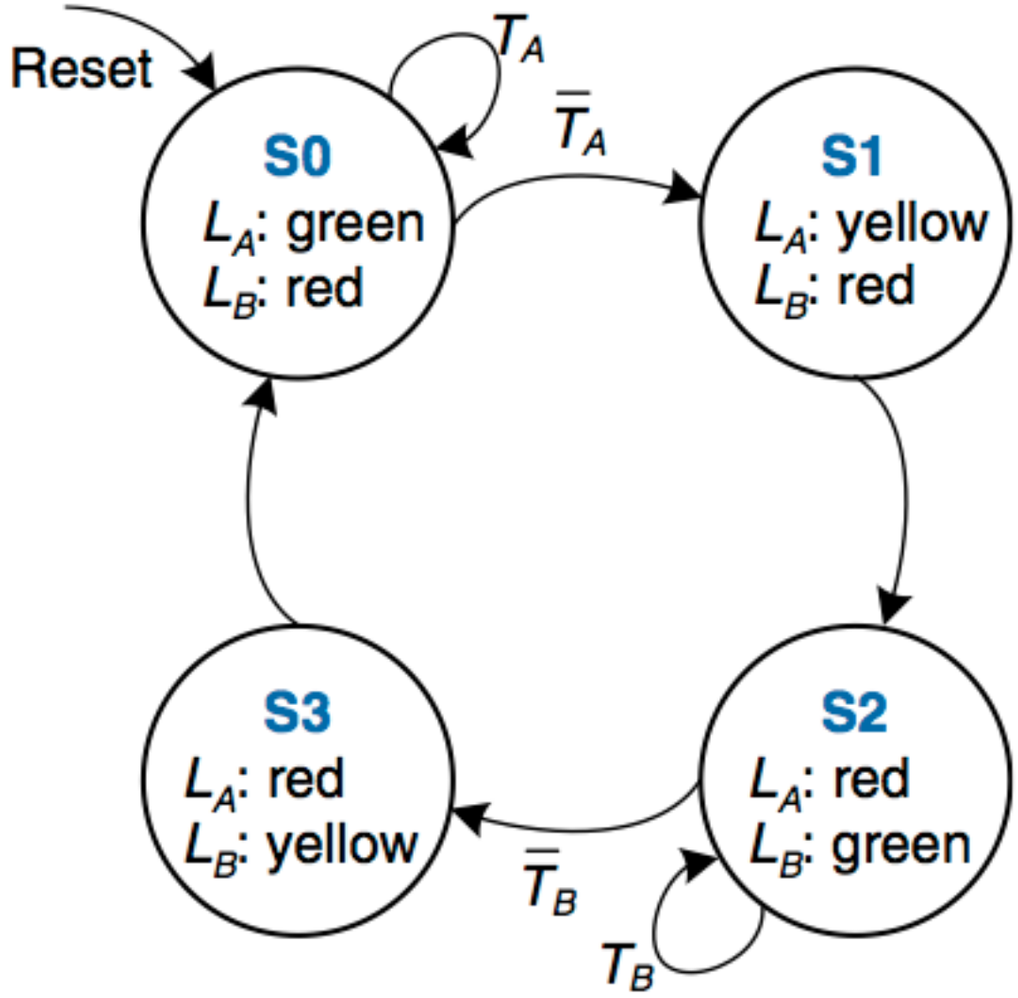


Tabla de transición de estados

Current State S	Inputs T_A	T_B	Next State S'
S0	0	X	S1
S0	1	X	S0
S1	X	X	S2
S2	X	0	S3
S2	X	1	S2
S3	X	X	S0

Diseño de circuitos secuenciales

Ejemplo

Codificación de los estados

State	Encoding $S_{1:0}$
S0	00
S1	01
S2	10
S3	11

Codificación de las salidas

Output	Encoding $L_{1:0}$
green	00
yellow	01
red	10

Diseño de circuitos secuenciales

Ejemplo

Tabla de transición codificada

Current State S_1	S_0	Inputs T_A	T_B	Next State S'_1	S'_0
0	0	0	X	0	1
0	0	1	X	0	0
0	1	X	X	1	0
1	0	X	0	1	1
1	0	X	1	1	0
1	1	X	X	0	0

$$S'_1 = \overline{S}_1 S_0 + S_1 \overline{S}_0 \overline{T}_B + S_1 \overline{S}_0 T_B$$

$$S'_0 = \overline{S}_1 \overline{S}_0 \overline{T}_A + S_1 \overline{S}_0 \overline{T}_B$$

$$S'_1 = S_1 \oplus S_0$$

$$S'_0 = \overline{S}_1 \overline{S}_0 \overline{T}_A + S_1 \overline{S}_0 \overline{T}_B$$

Diseño de circuitos secuenciales

Ejemplo

Tabla de Salidas

Current State S_1	S_0	Outputs L_{A1}	L_{A0}	L_{B1}	L_{B0}
0	0	0	0	1	0
0	1	0	1	1	0
1	0	1	0	0	0
1	1	1	0	0	1

$$L_{A1} = S_1$$

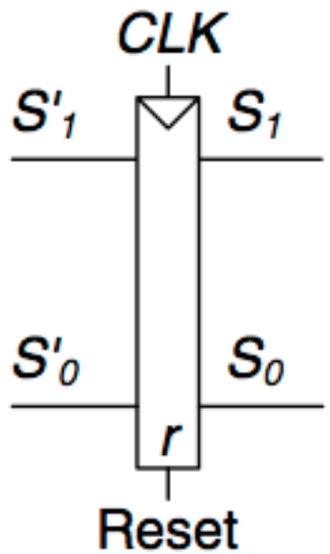
$$L_{A0} = \overline{S}_1 S_0$$

$$L_{B1} = \overline{S}_1$$

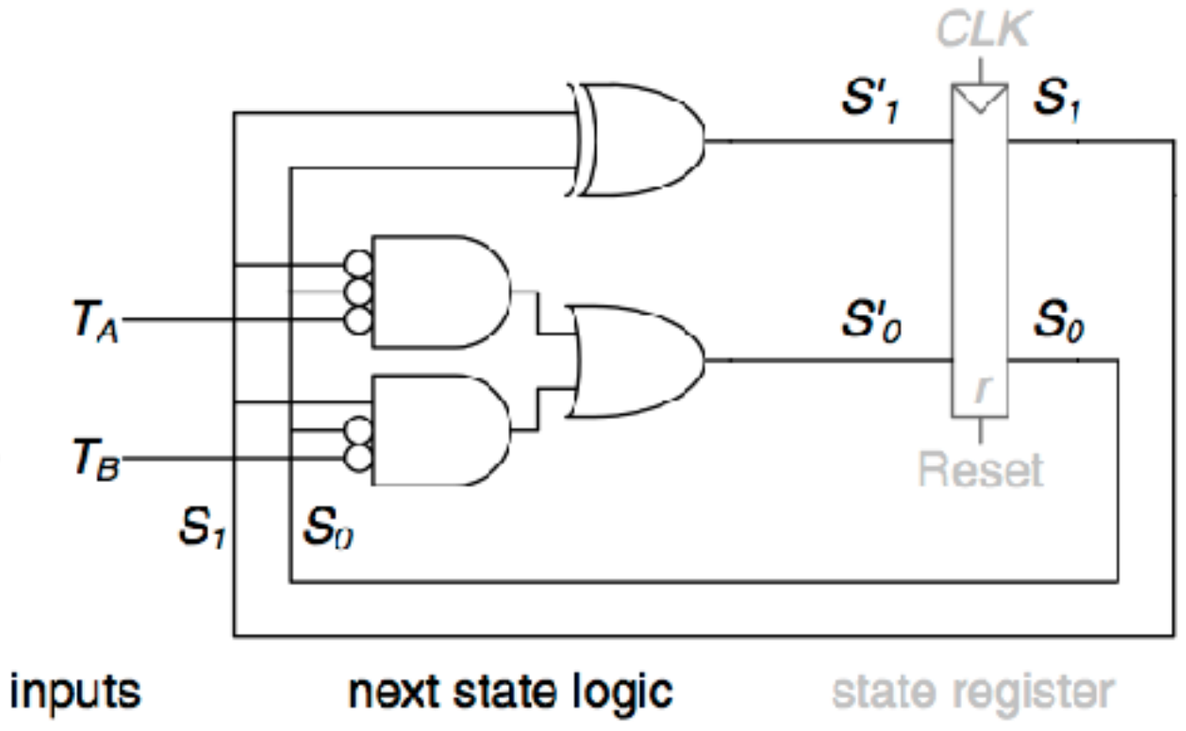
$$L_{B0} = S_1 S_0$$

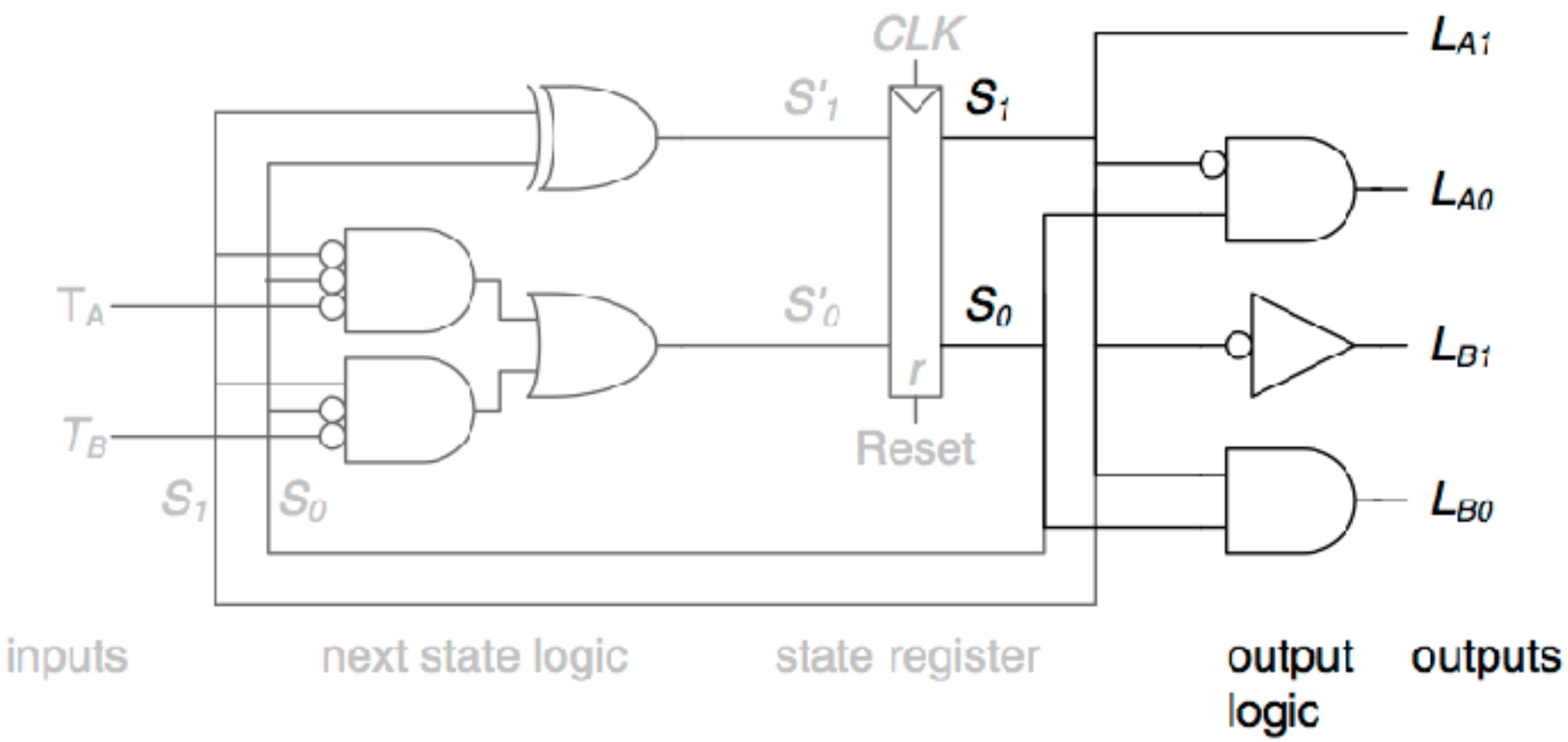
Diseño de circuitos secuenciales

Registro de estados



Lógica para generación de próximo estado

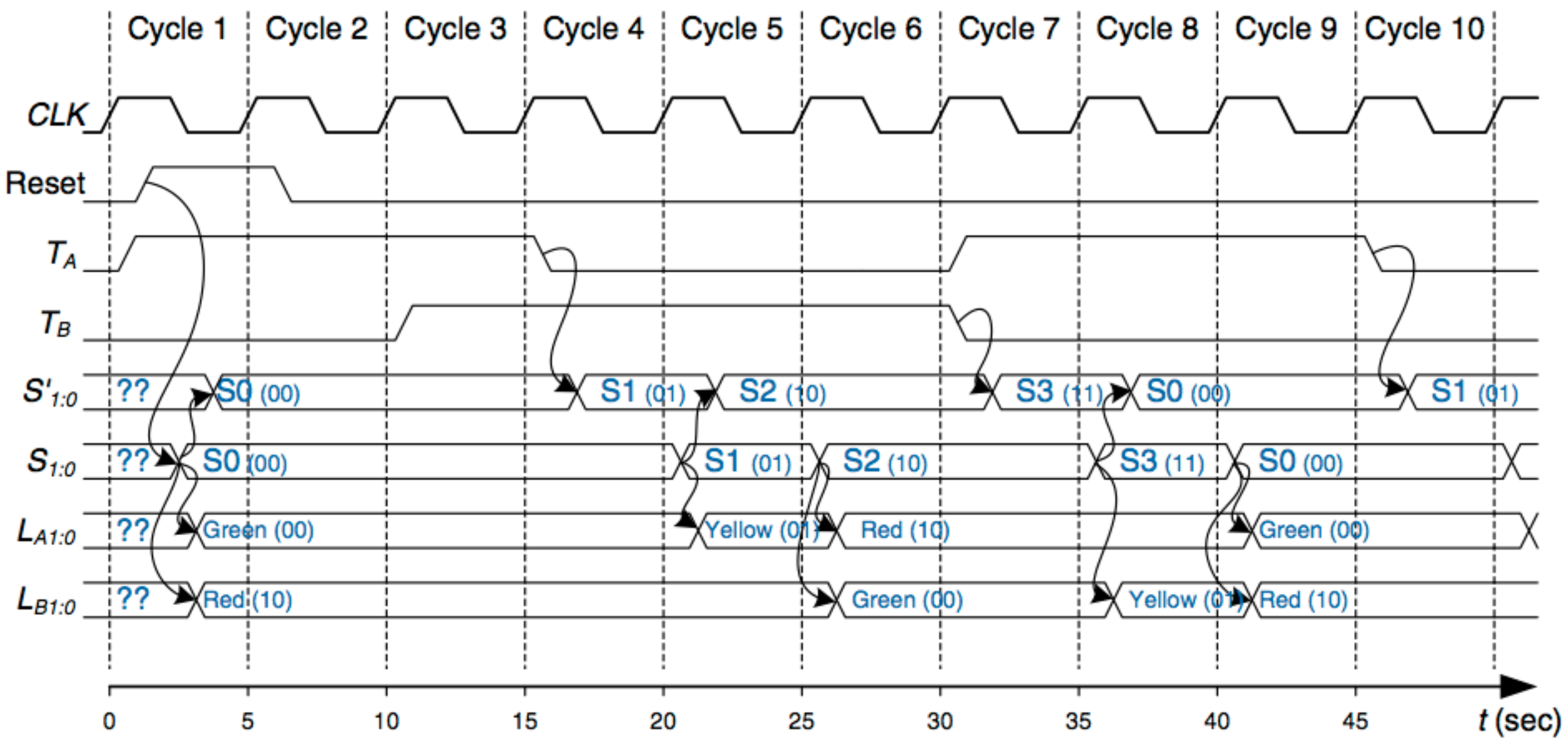




Máquina de estado completa

Diseño de circuitos secuenciales

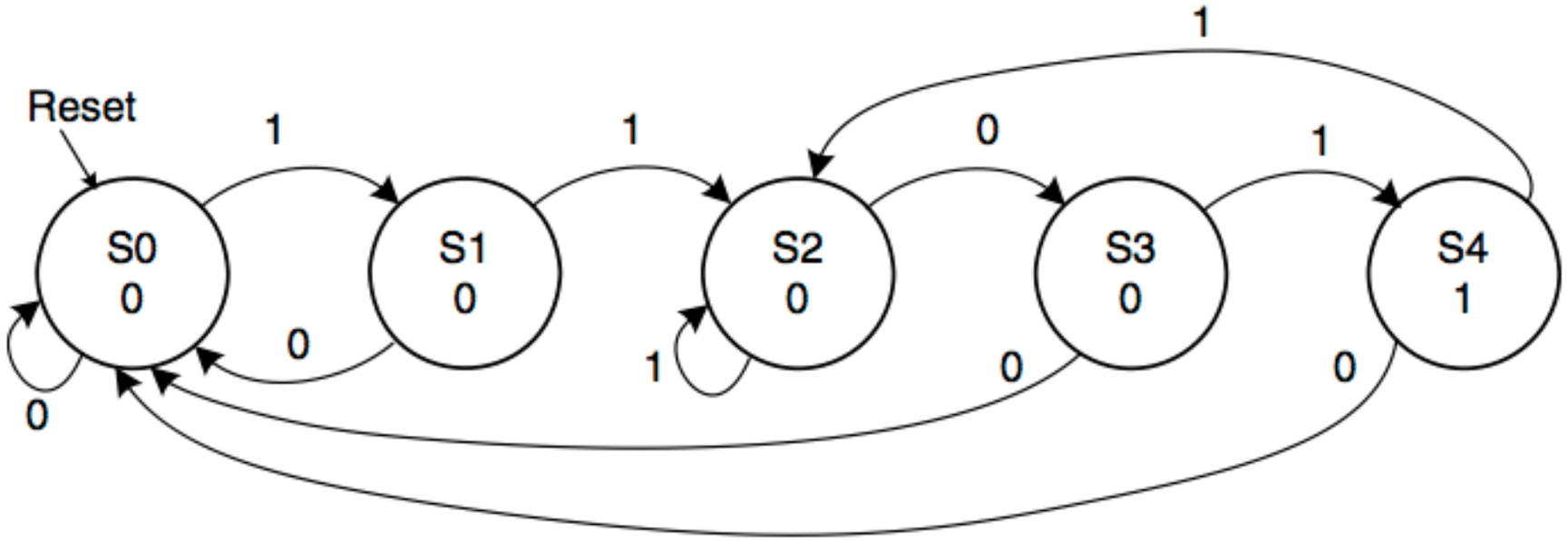
Diagrama de tiempos para el controlador de semáforo



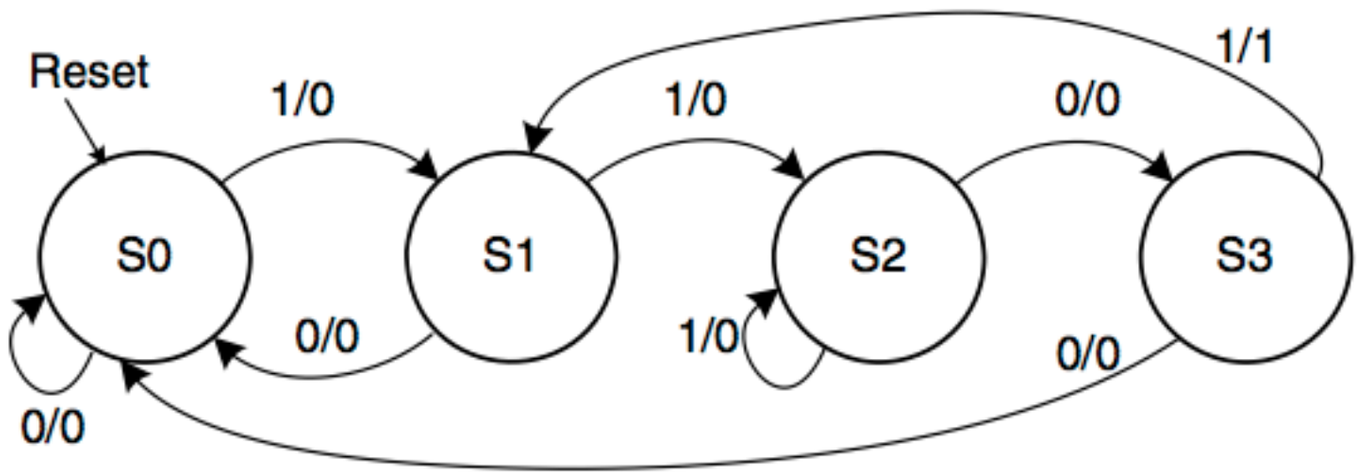
Diseño de circuitos secuenciales

Comparación entre una máquina de Moore y una de Mealy

- Detector de secuencia 1101 (máquina de Moore)



- Detector de secuencia 1101 (máquina de Mealy)



Diseño de circuitos secuenciales

Comparación entre una máquina de Moore y una de Mealy

- Tabla de transición de estados de

Current State *S*	Input *A*	Next State *S'*
S0	0	S0
S0	1	S1
S1	0	S0
S1	1	S2
S2	0	S3
S2	1	S2
S3	0	S0
S3	1	S4
S4	0	S0
S4	1	S2

- Tabla de salidas de estados de Moore

Current State *S*	Output *Y*
S0	0
S1	0
S2	0
S3	0
S4	1

Diseño de circuitos secuenciales

Comparación entre una máquina de Moore y una de Mealy

- Tabla de transición de estados de Moore codificada

Current State S_2	S_1	S_0	Input A	Next State S'_2	S'_1	S'_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	0
0	0	1	1	0	1	0
0	1	0	0	0	1	1
0	1	0	1	0	1	0
0	1	1	0	0	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	0
1	0	0	1	0	1	0

- Tabla de salidas de Moore codificada

Current State S_2	S_1	S_0	Output Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1

Diseño de circuitos secuenciales

Comparación entre una máquina de Moore y una de Mealy

- Tabla de transición - salida de Mealy

Current State S	Input A	Next State S'	Output Y
S0	0	S0	0
S0	1	S1	0
S1	0	S0	0
S1	1	S2	0
S2	0	S3	0
S2	1	S2	0
S3	0	S0	0
S3	1	S1	1

Diseño de circuitos secuenciales

Comparación entre una máquina de Moore y una de Mealy

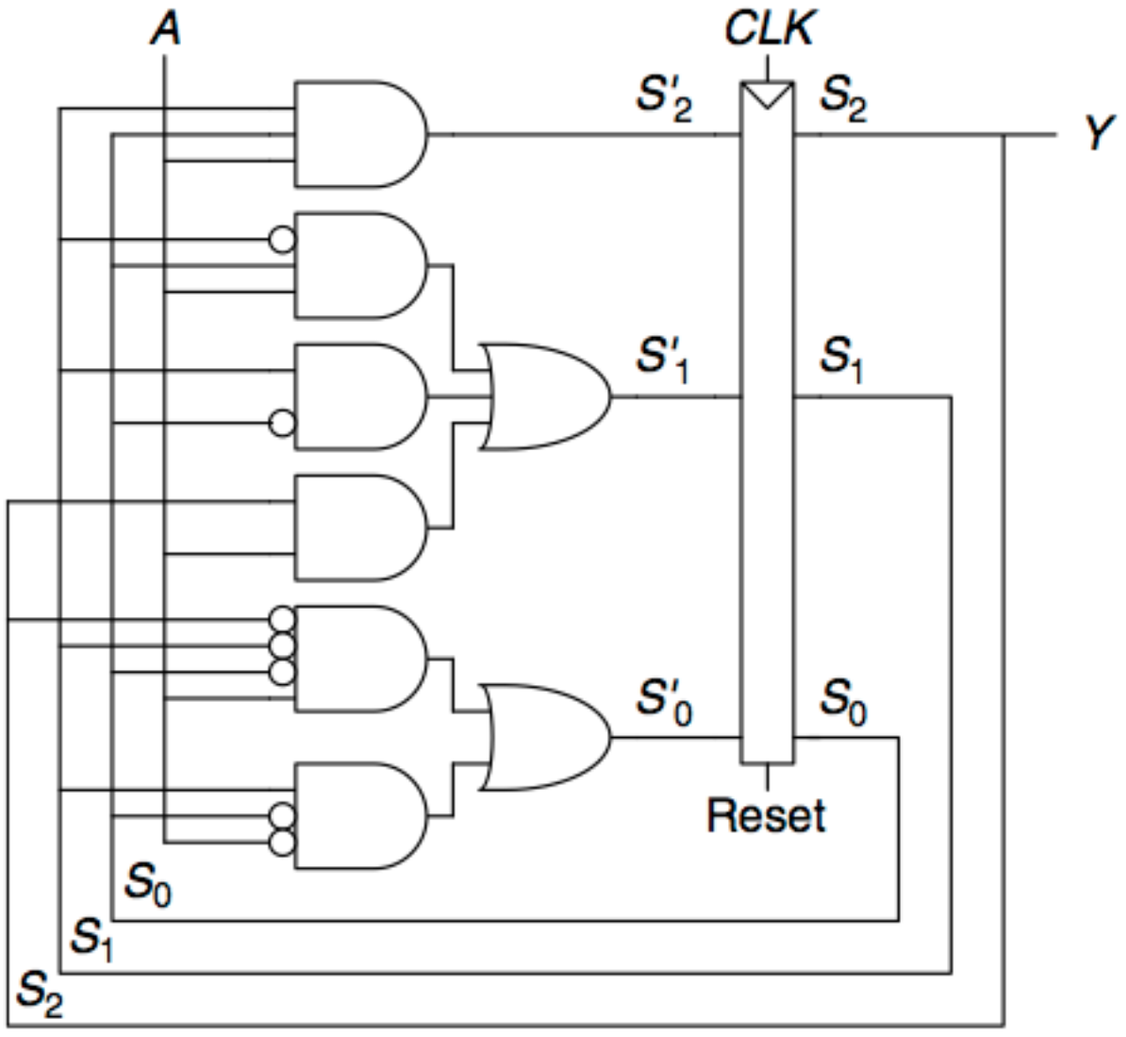
- Tabla de transición - salida de Mealy codificada

Current State S_1	S_0	Input A	Next State S'_1	S'_0	Output Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	1	1	0
1	0	1	1	0	0
1	1	0	0	0	0
1	1	1	0	1	1

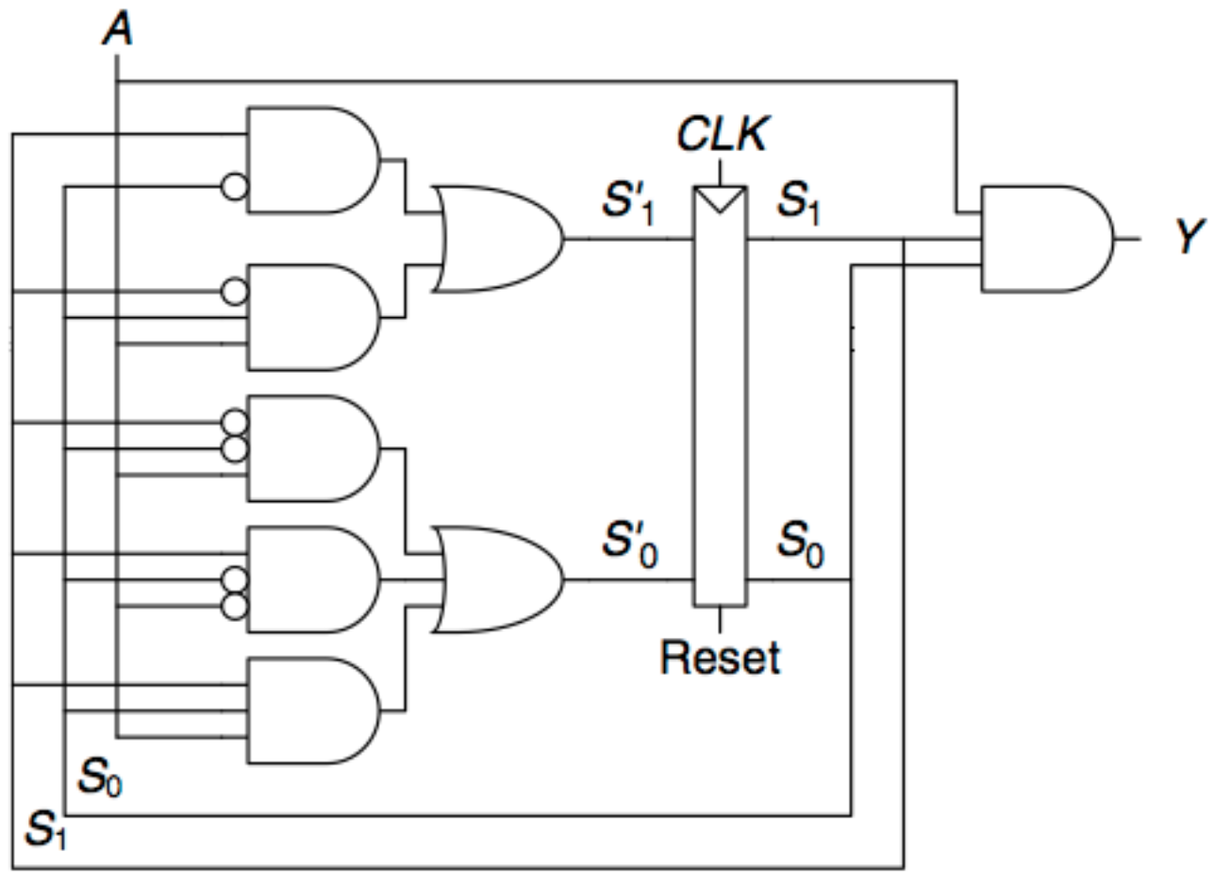
Diseño de circuitos secuenciales

Comparación entre una máquina de Moore y una de Mealy

- Máquina de Moore



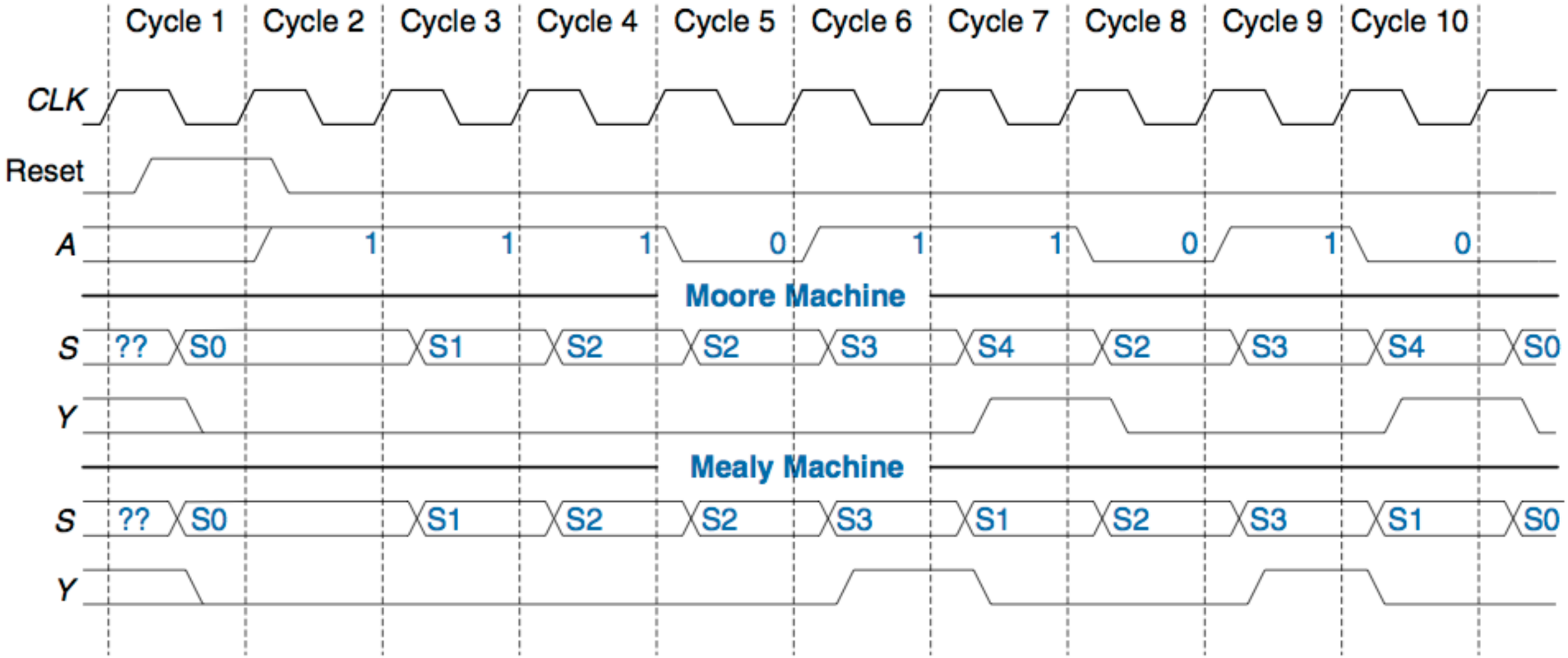
- Máquina de Mealy



Diseño de circuitos secuenciales

Comparación entre una máquina de Moore y una de Mealy

- Diagrama de tiempos



Diseño de circuitos secuenciales

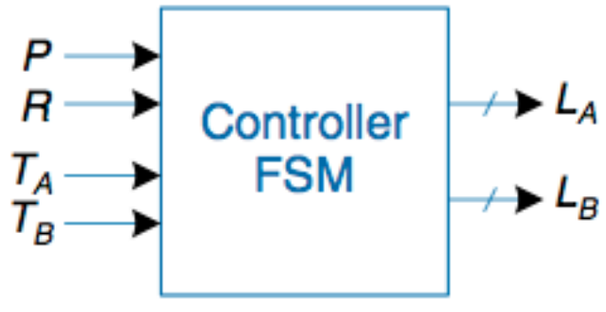
Máquinas de estado No Factorizadas y Factorizadas

- Al ejemplo del semáforo le agregaremos un modo de desfile que permita poner en verde la luz de la calle Bravado mientras se reliaza un desfile.
- Para esto se utilizan dos botones, *P* y *R*.
- Cuando *P* se aprieta durante un ciclo se ingresa al modo de desfile.
- Cuando *R* se aprieta durante un ciclo se sale del modo de parada.
- Cuando se entra al modo de desfile el controlador prosigue su ciclo normal hasta que L_B se pone verde, luego permanece en ese estado, con L_B verde hasta salir del modo.

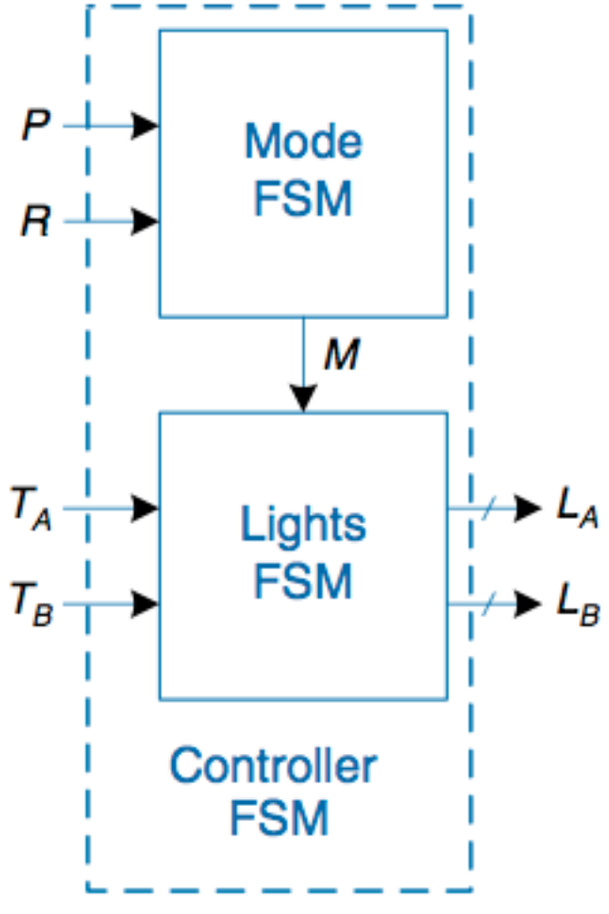
Diseño de circuitos secuenciales

Máquinas de estado No Factorizadas y Factorizadas

- Solución No



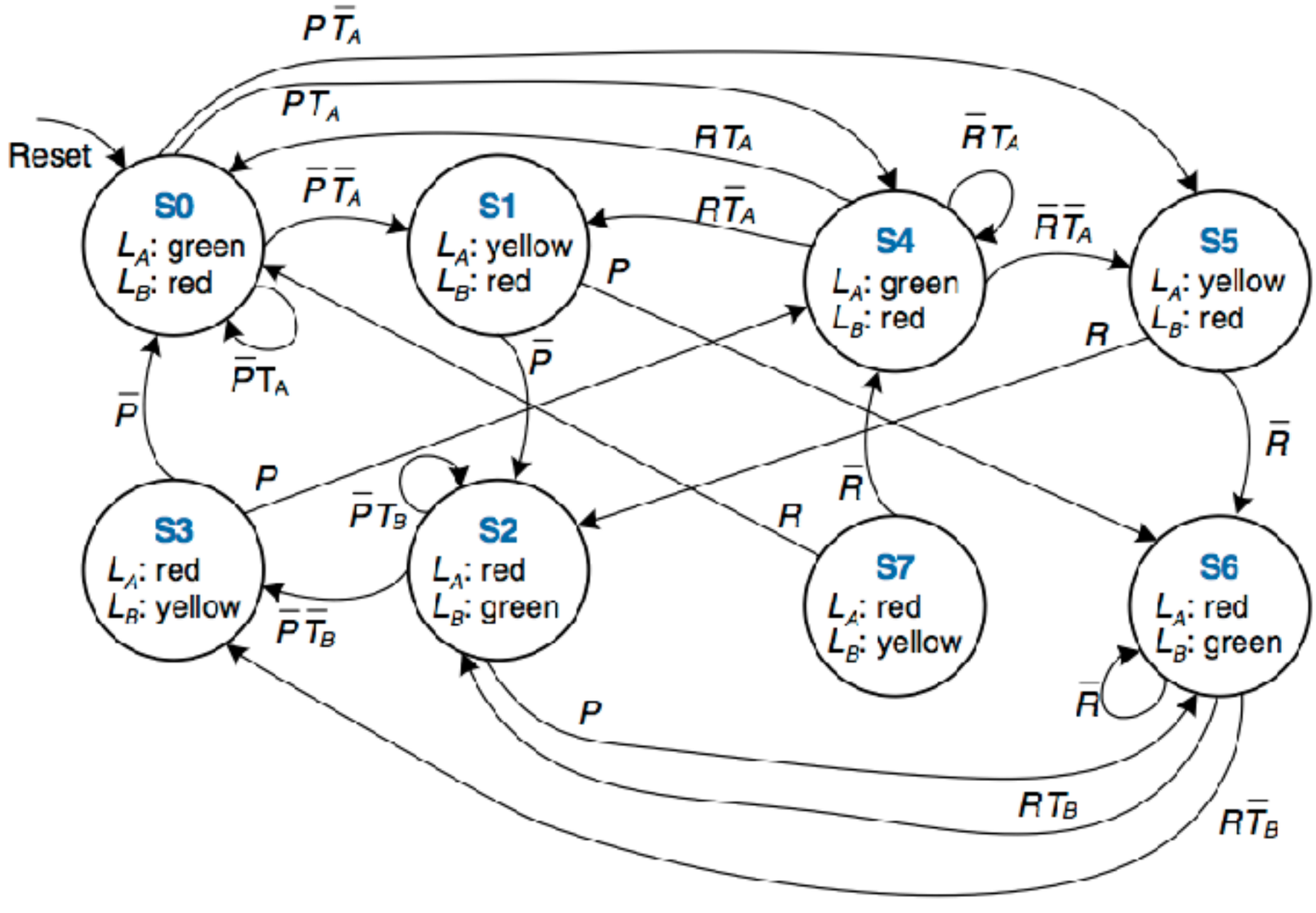
- Solución Factorizada



Diseño de circuitos secuenciales

Máquinas de estado No Factorizadas y Factorizadas

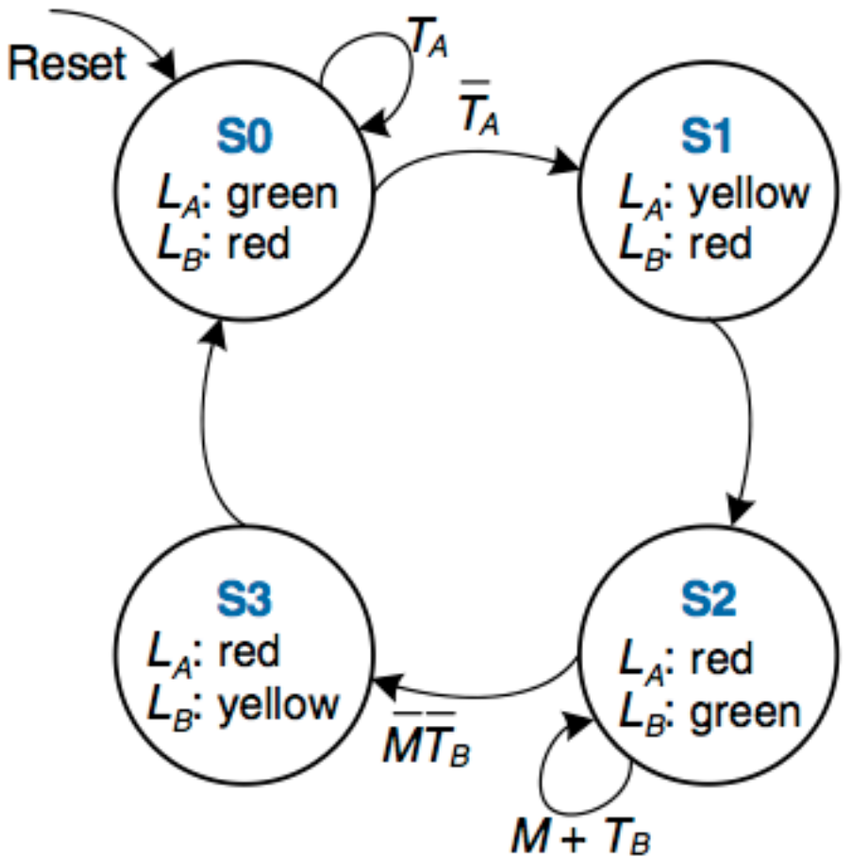
- Solución No



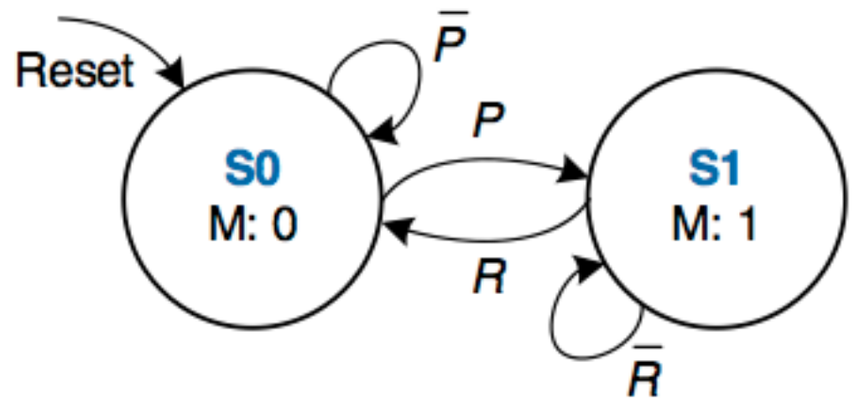
Diseño de circuitos secuenciales

Máquinas de estado No Factorizadas y Factorizadas

- Solución Factorizada



Lights FSM



Mode FSM

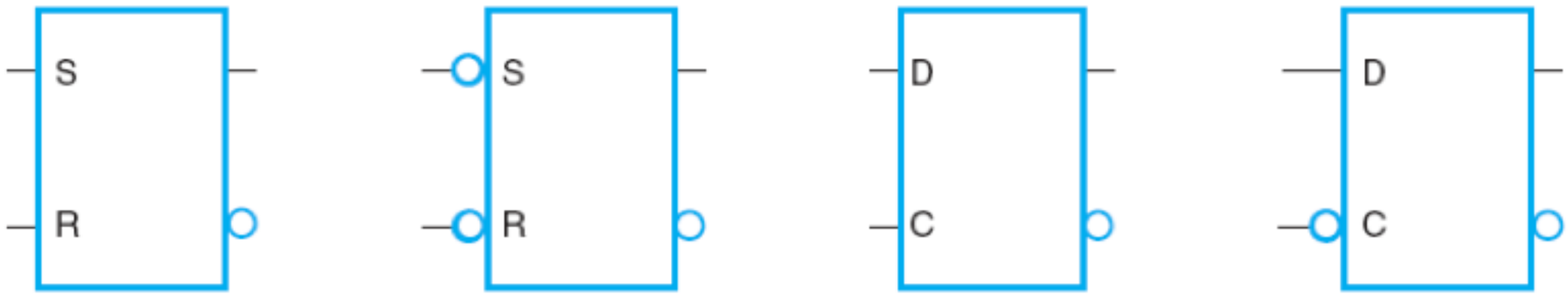
Diseño de circuitos secuenciales

Resumen de procedimientos para diseñar Máquinas de Estados Finitos

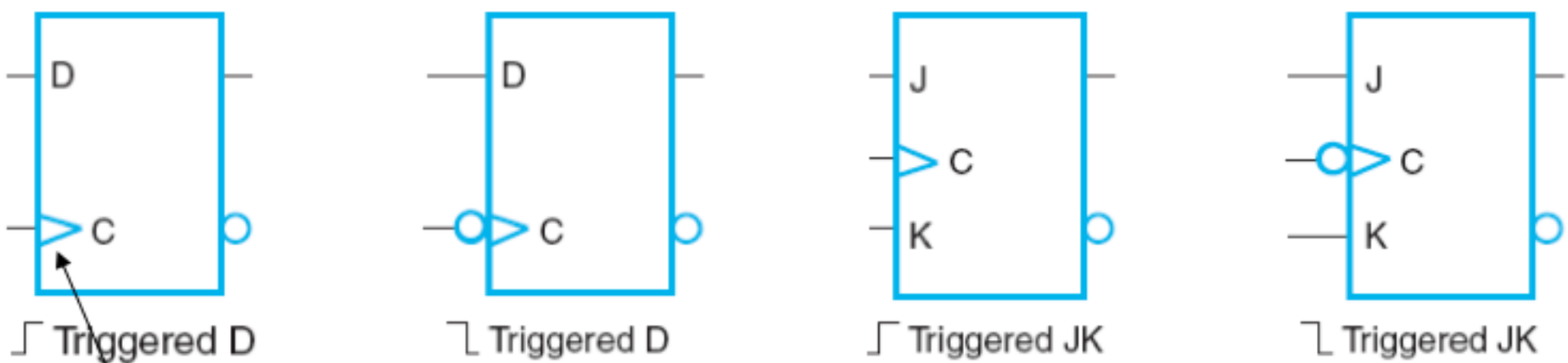
- Identificar entradas y salidas.
- Dibujar un diagrama de transición de estados.
- Para una máquina de Moore:
  - Escribir la tabla de transición de estados.
  - Escribir la tabla de salida.
- Para una máquina de Mealy:
  - Escribir la tabla combinada de transición de estados y salidas.
- Seleccionar la codificación para los estados.
- Escribir las ecuaciones Booleanas para el próximo estado y las salidas.
- Dibujar el esquemático del circuito.

Diseño de circuitos secuenciales

Resumen: Diagrama estándares de *Latches*



Diagramas estándares de *Flipflops* gatillados por flanco



Diseño de circuitos secuenciales

Tablas características de los *Flipflops*:

(a) *JK* Flip-Flop

J	K	$Q(t+1)$	Operation
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$\overline{Q}(t)$	Complement

(b) *SR* Flip-Flop

S	R	$Q(t+1)$	Operation
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	?	Undefined

(c) *D* Flip-Flop

D	$Q(t+1)$	Operation
0	0	Reset
1	1	Set

(d) *T* Flip-Flop

T	$Q(t+1)$	Operation
0	$Q(t)$	No change
1	$\overline{Q}(t)$	Complement

Diseño de circuitos secuenciales

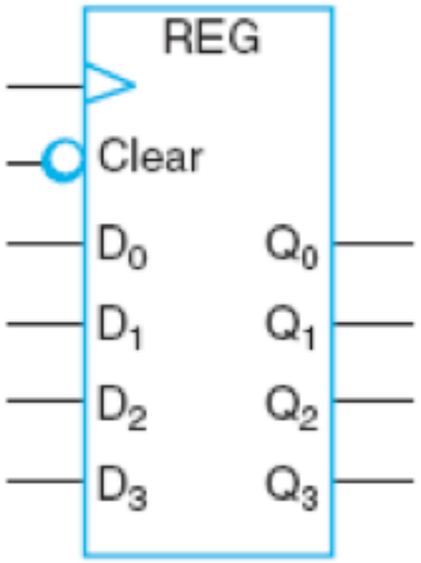
- Registros
- Registros de desplazamiento (shift registers)
- Contadores de cascada (ripple counters)
- Contadores síncronos binarios
- Otros contadores

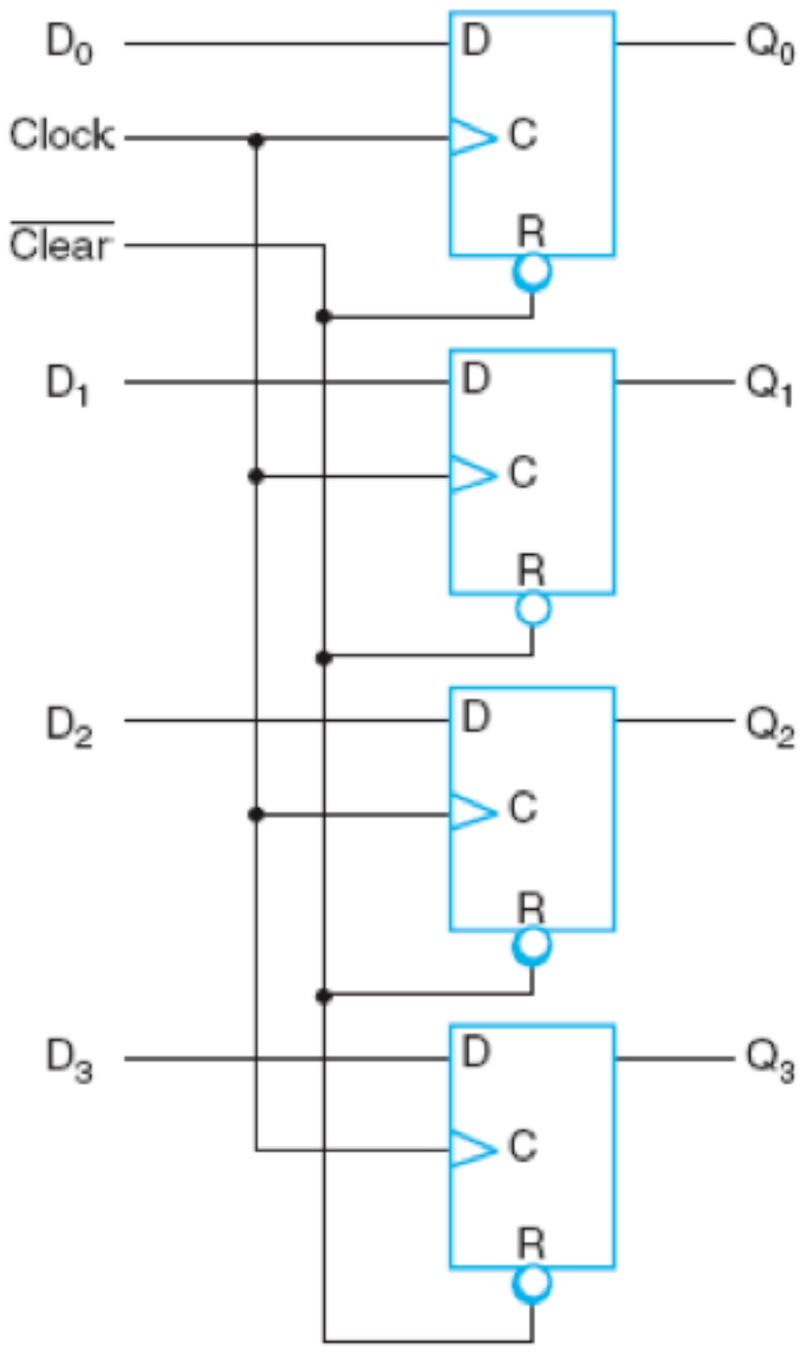
Diseño de circuitos secuenciales

Registros

Los registros son un conjunto de flip-flops y lógica combinacional asociada que sirven para almacenar información binaria (un bit por cada flip-flop)

Ejemplo de un registro de 4 bits con clear.

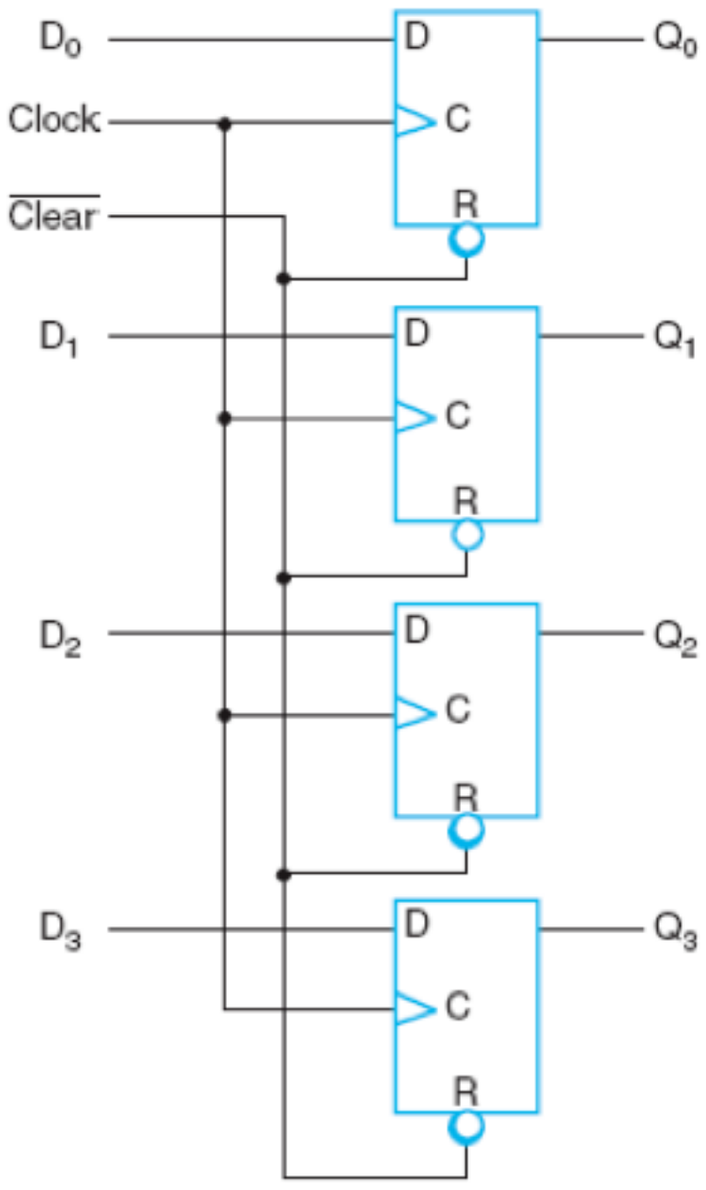




Diseño de circuitos secuenciales

Registros

Registros con carga paralela: todos los bits son cargados al mismo tiempo.

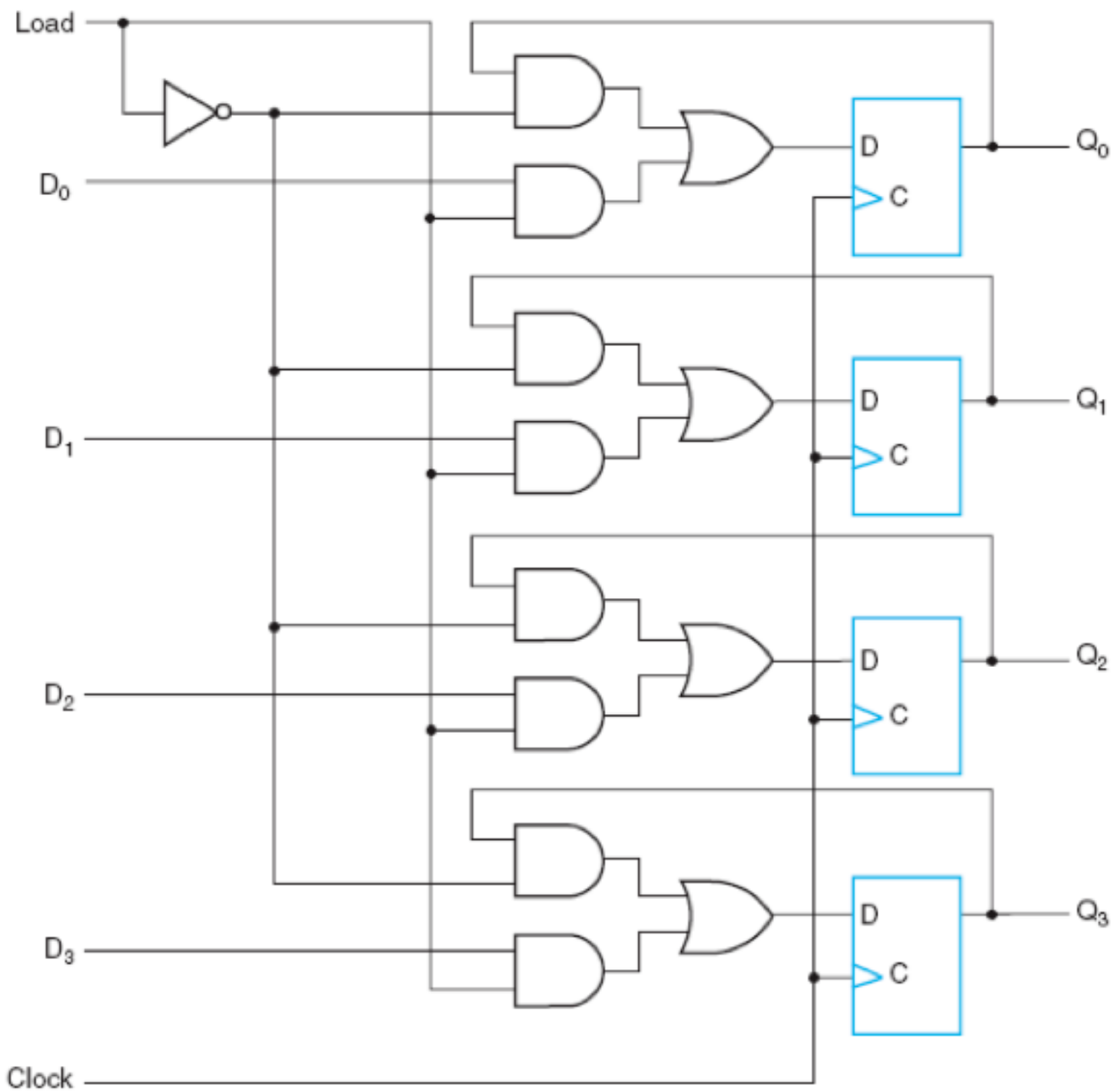


Para evitar que cambien los datos, se emplea una señal de “load”.

Diseño de circuitos secuenciales

Registros

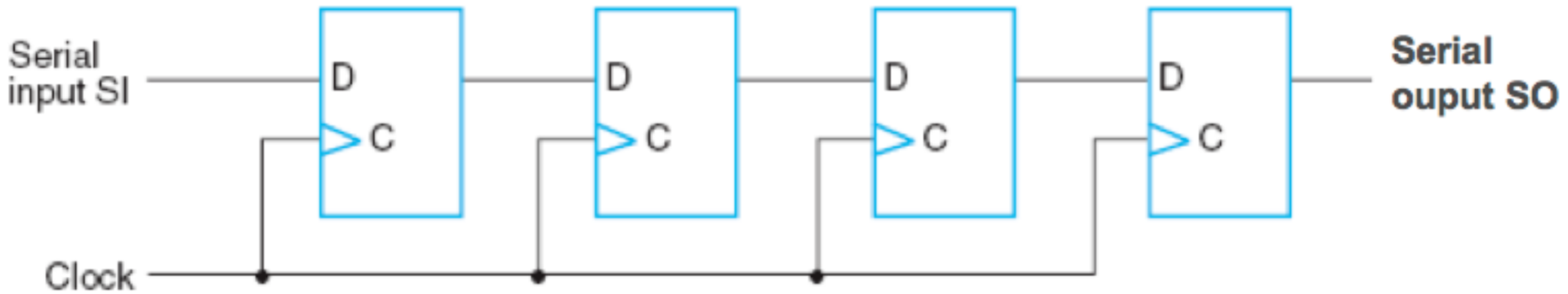
Más seguro es controlar la carga a la entrada de los flip-flops



Diseño de circuitos secuenciales

Registros de desplazamiento

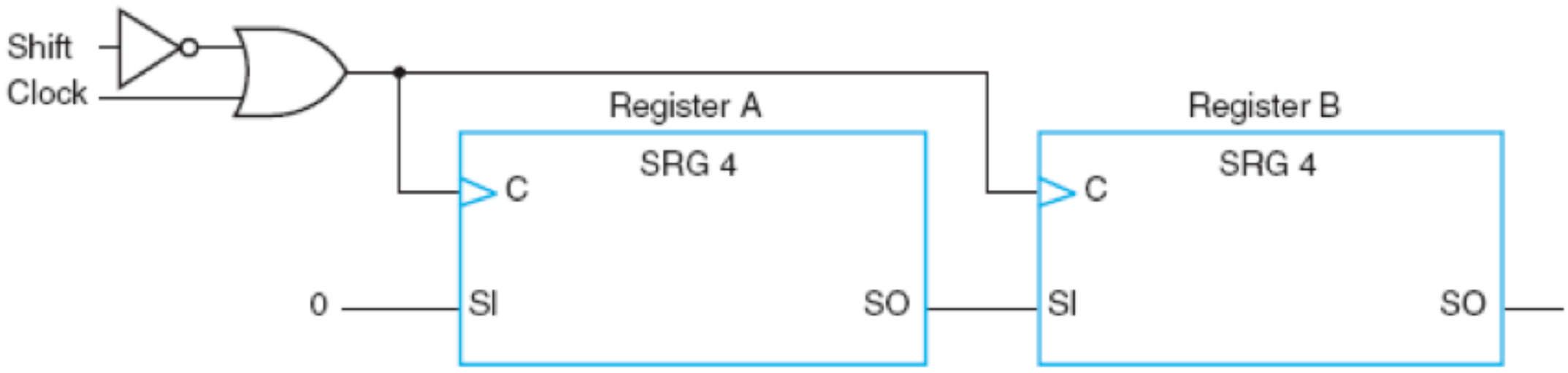
"Shift registers"



Diseño de circuitos secuenciales

Registros de desplazamiento

Transferencia en serie

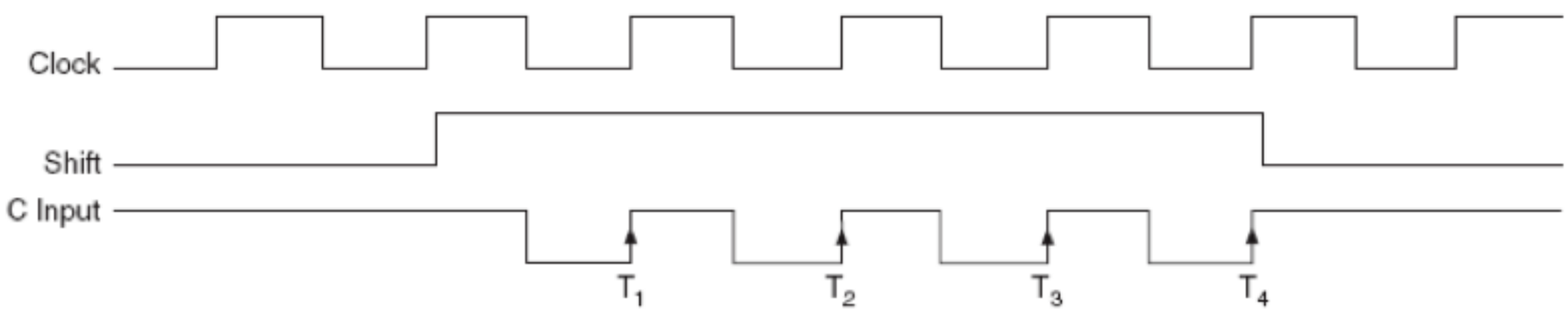


Diseño de circuitos secuenciales

Registros de desplazamiento

Transferencia en serie

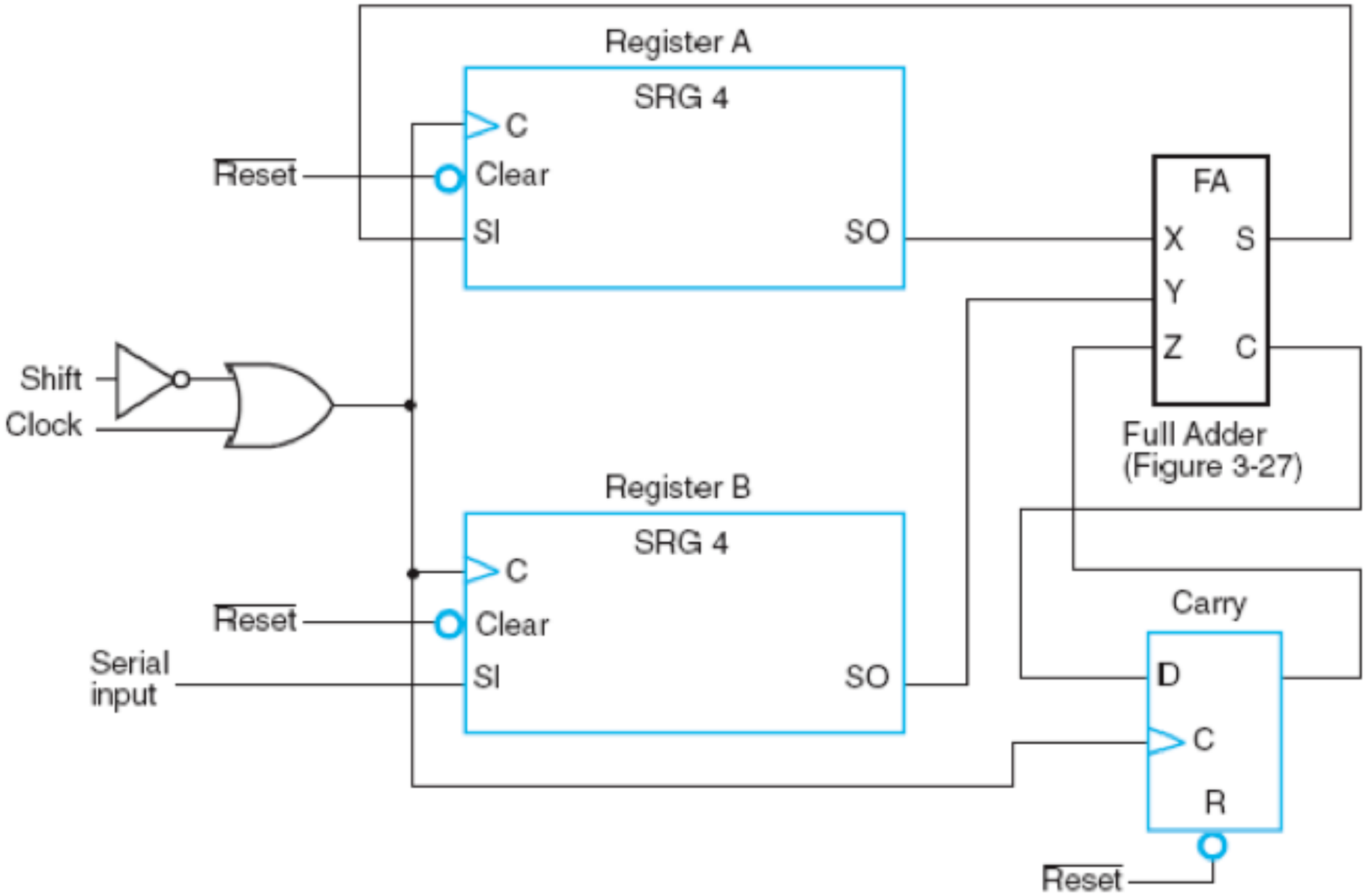
Timing pulse	Shift Register A				Shift Register B			
Initial value	1	0	1	1	0	0	1	0
After T_1	0	1	0	1	1	0	0	1
After T_2	0	0	1	0	1	1	0	0
After T_3	0	0	0	1	0	1	1	0
After T_4	0	0	0	0	1	0	1	1



Diseño de circuitos secuenciales

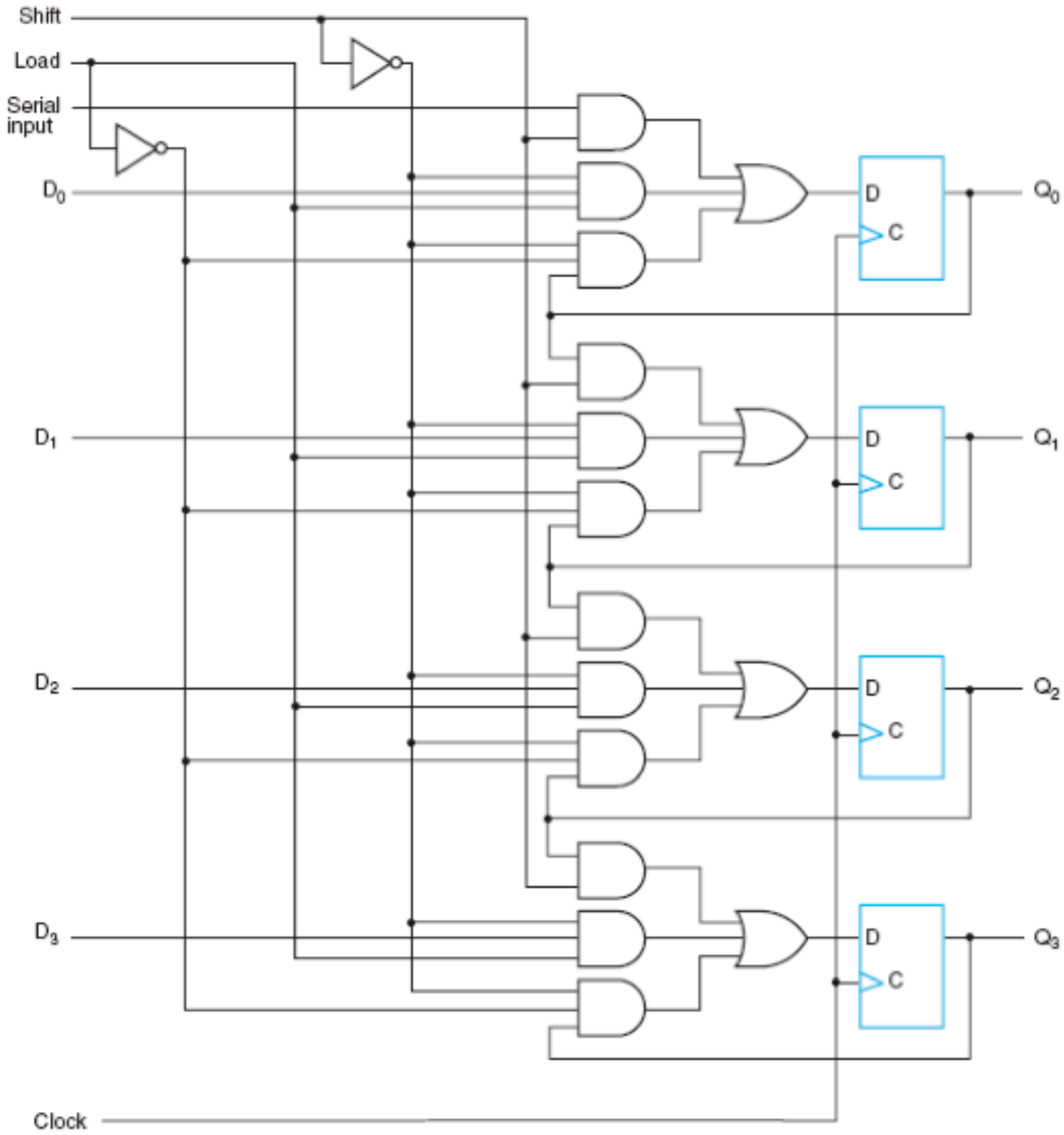
Registros de desplazamiento

Ejemplo de aplicación: “sumador serie”



Diseño de circuitos secuenciales

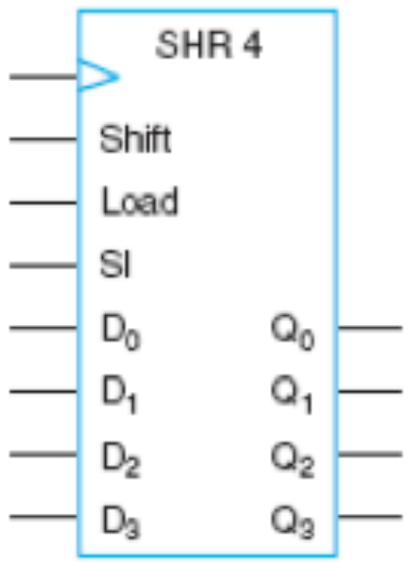
Registros de desplazamiento con carga paralela



Diseño de circuitos secuenciales

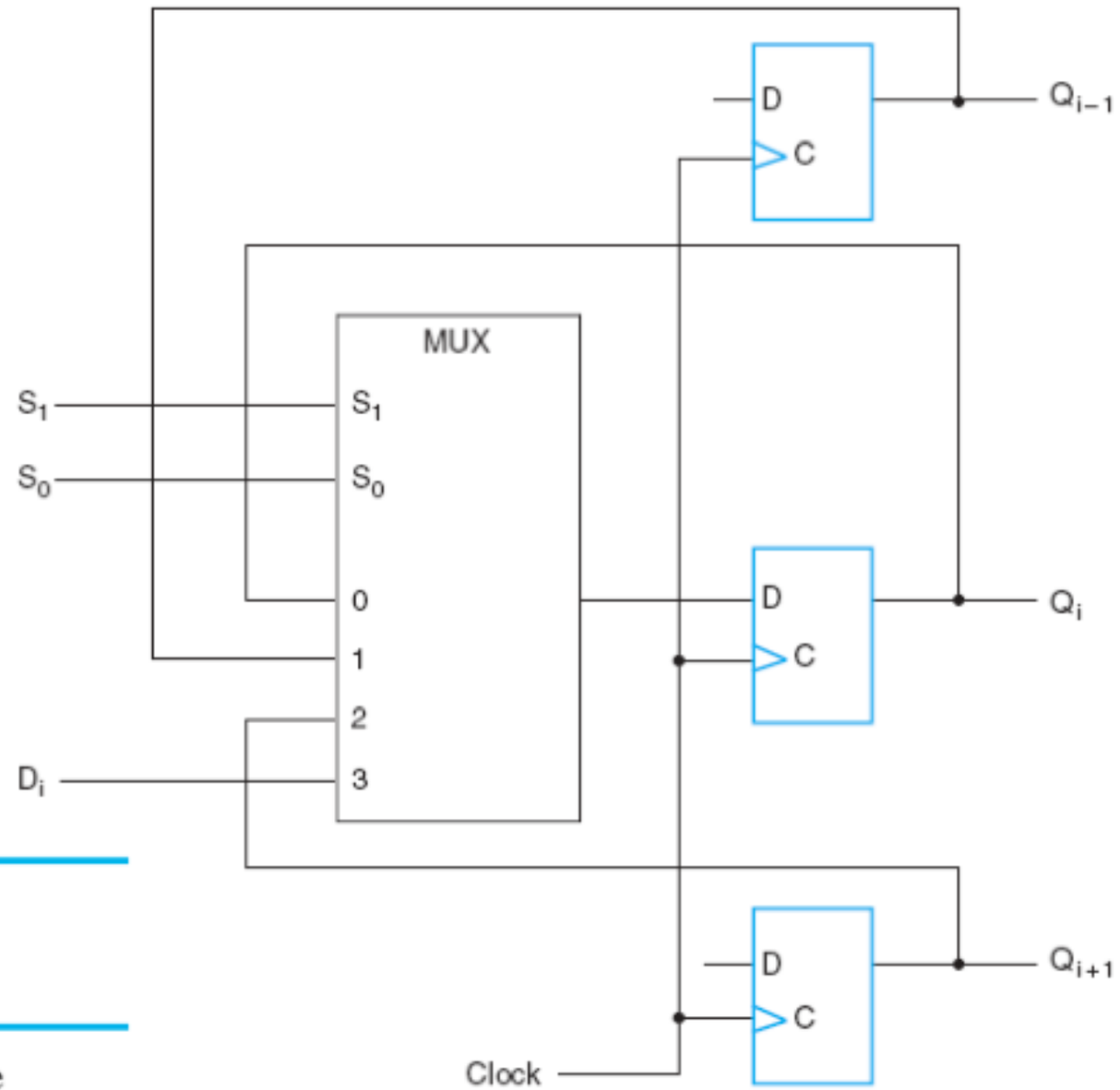
Registros de desplazamiento con carga paralela

Shift	Load	Operation
0	0	No change
0	1	Load parallel data
1	×	Shift down from Q_0 to Q_3



Diseño de circuitos secuenciales

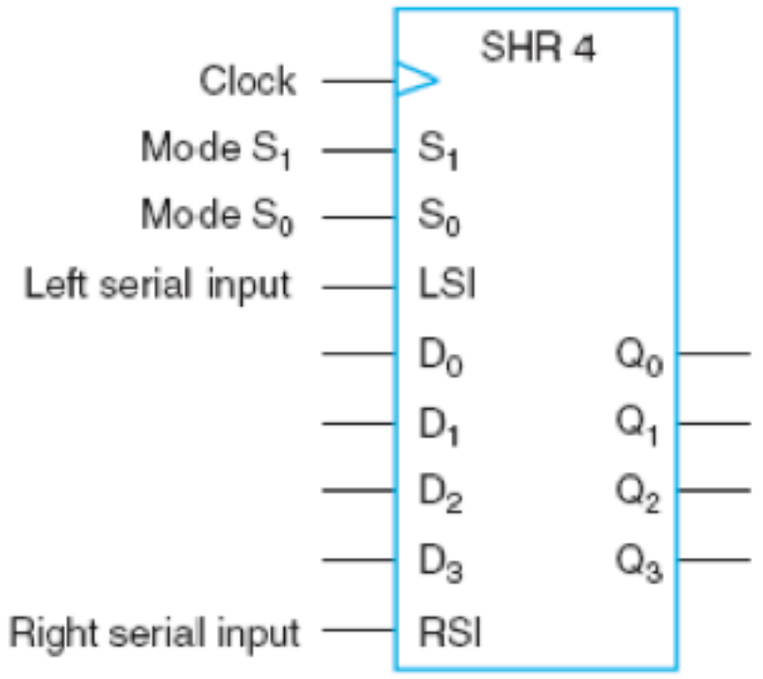
Shift register bi-direccional



Mode control		Register Operation
S_1	S_0	
0	0	No change
0	1	Shift down
1	0	Shift up
1	1	Parallel load

Diseño de circuitos secuenciales

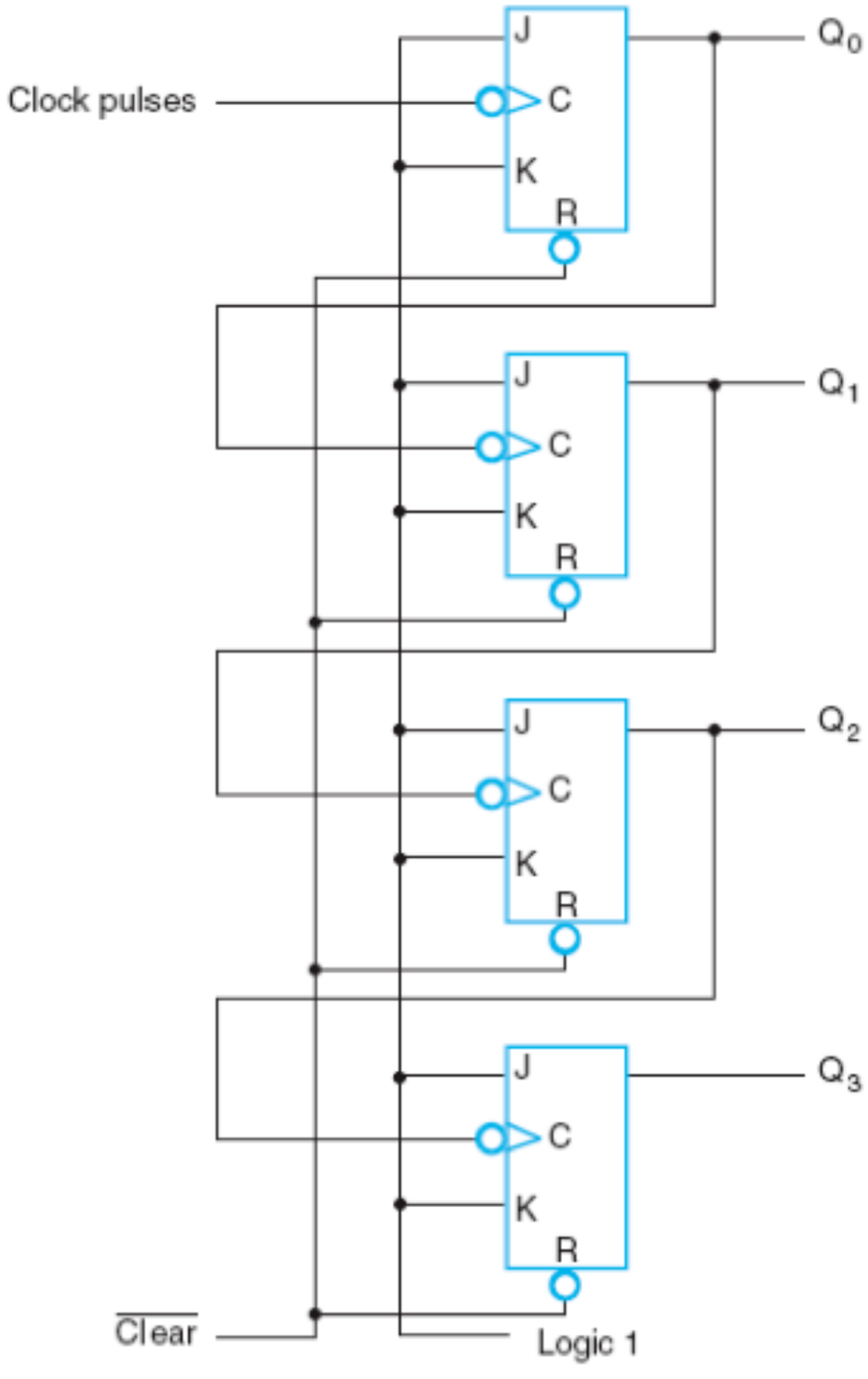
Shift register bi-direccional



Mode control		Register Operation
S_1	S_0	
0	0	No change
0	1	Shift down
1	0	Shift up
1	1	Parallel load

Diseño de circuitos secuenciales

Contador en cascada

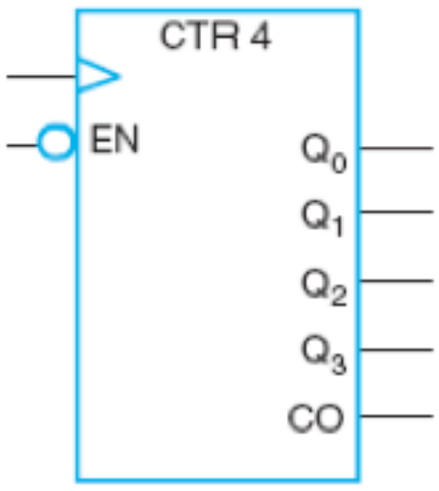


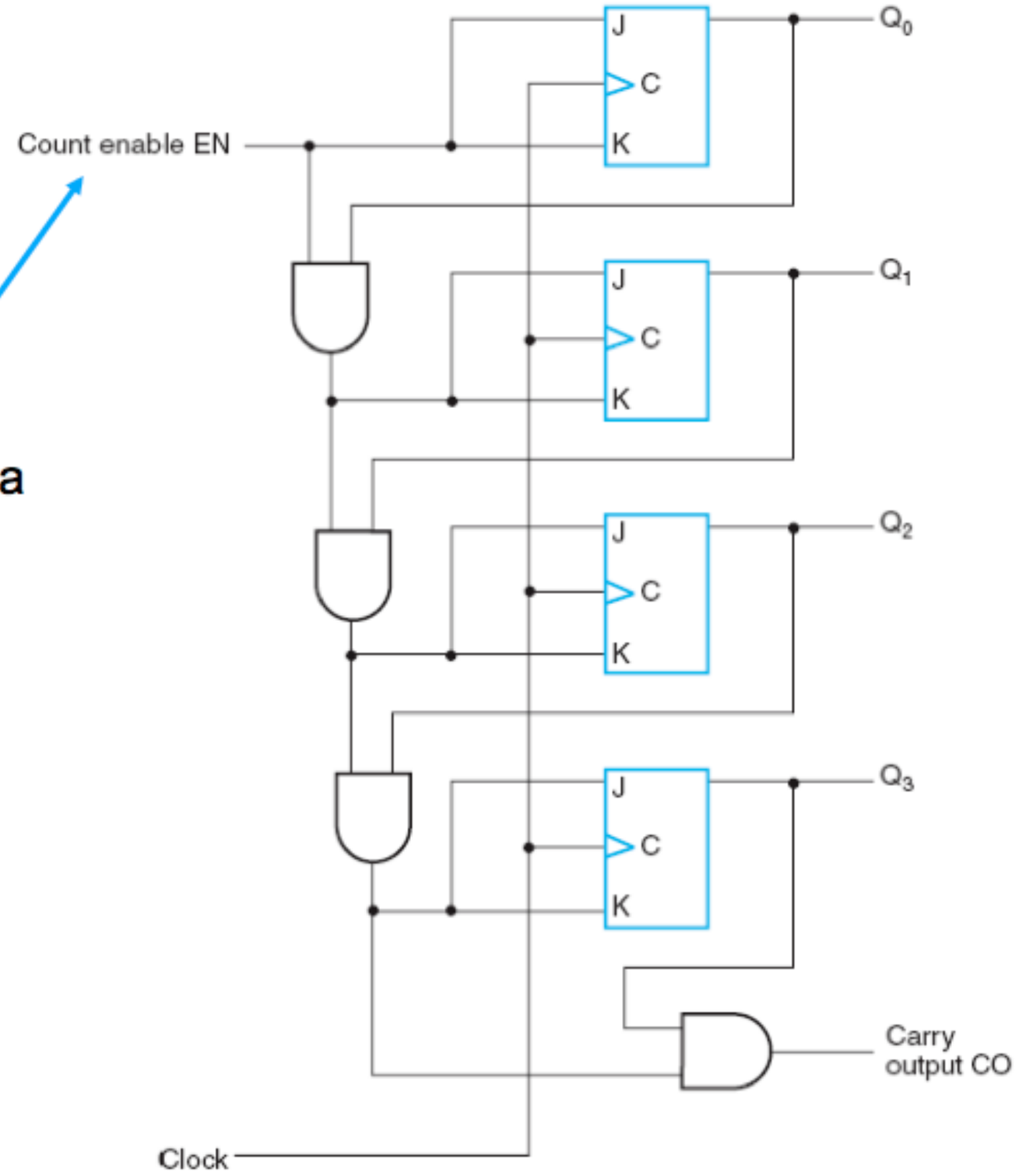
Q_3	Q_2	Q_1	Q_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Diseño de circuitos secuenciales

Contador síncrono

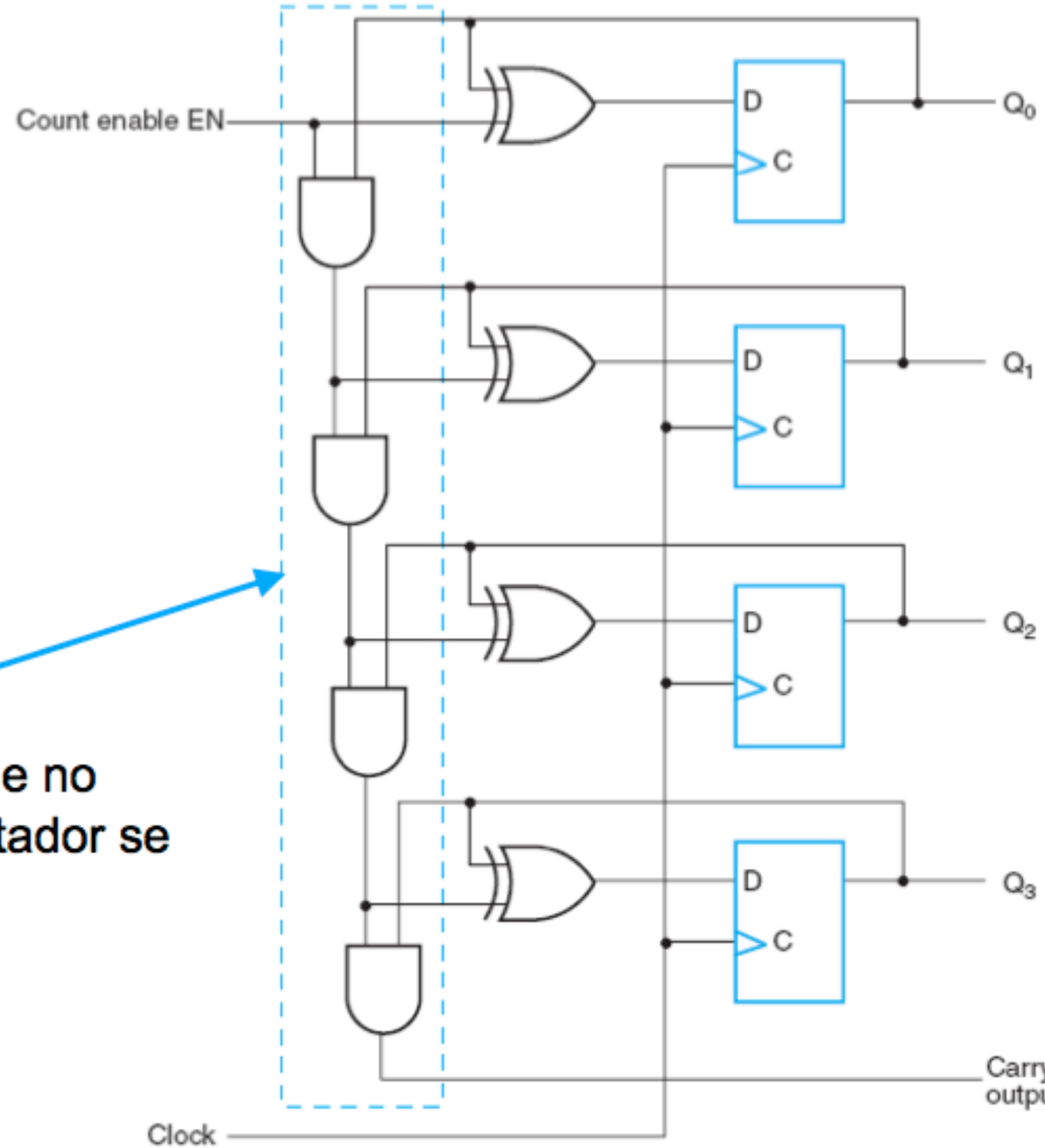
Para permitir la cuenta





Diseño de circuitos secuenciales

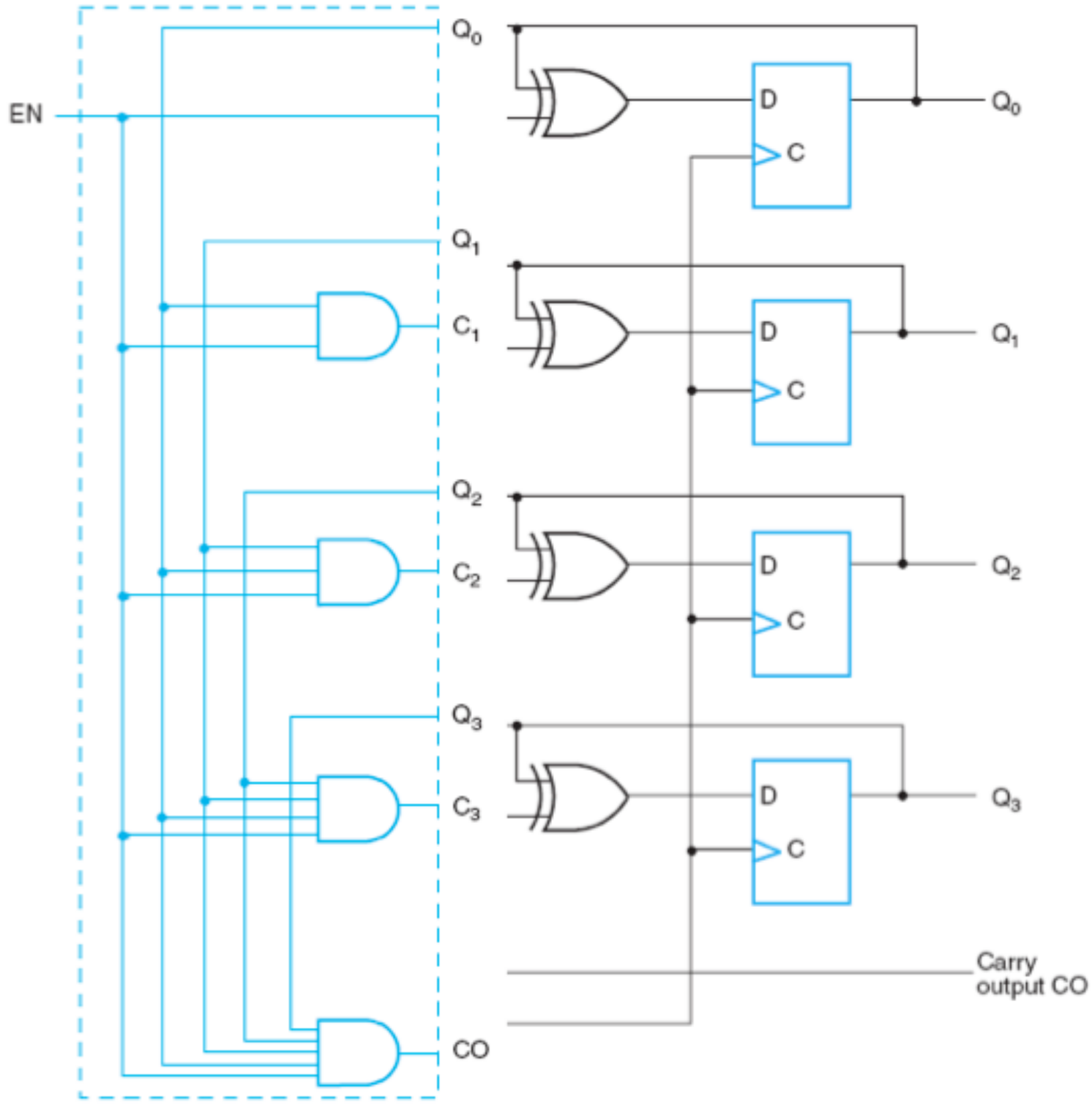
El mismo diseño con flip-flops D



Efecto ripple, que puede no ser deseable (este contador se llama contador serial)

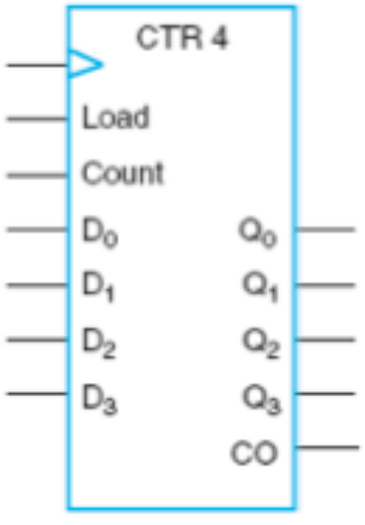
Diseño de circuitos secuenciales

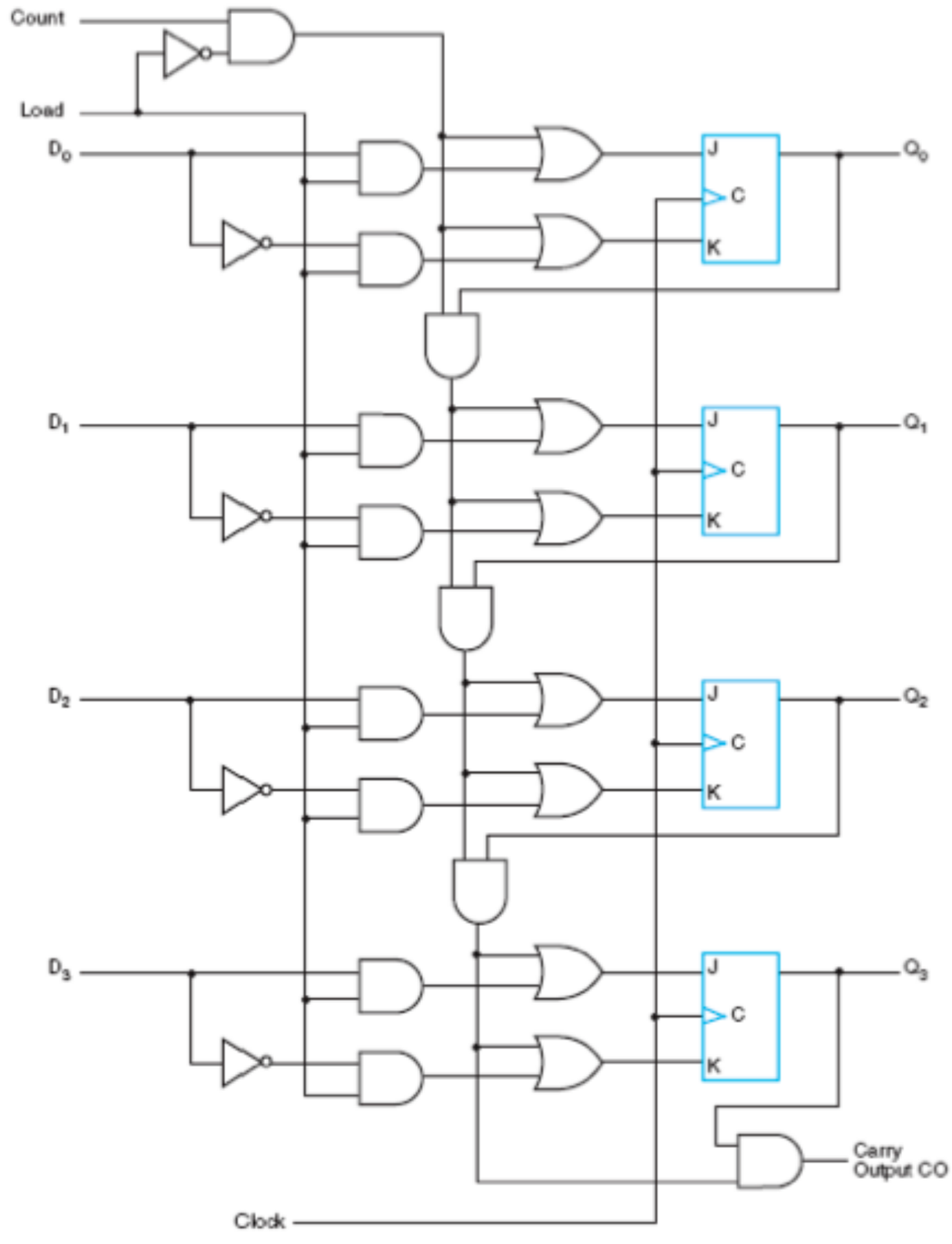
solución para
el efecto
ripple



Diseño de circuitos secuenciales

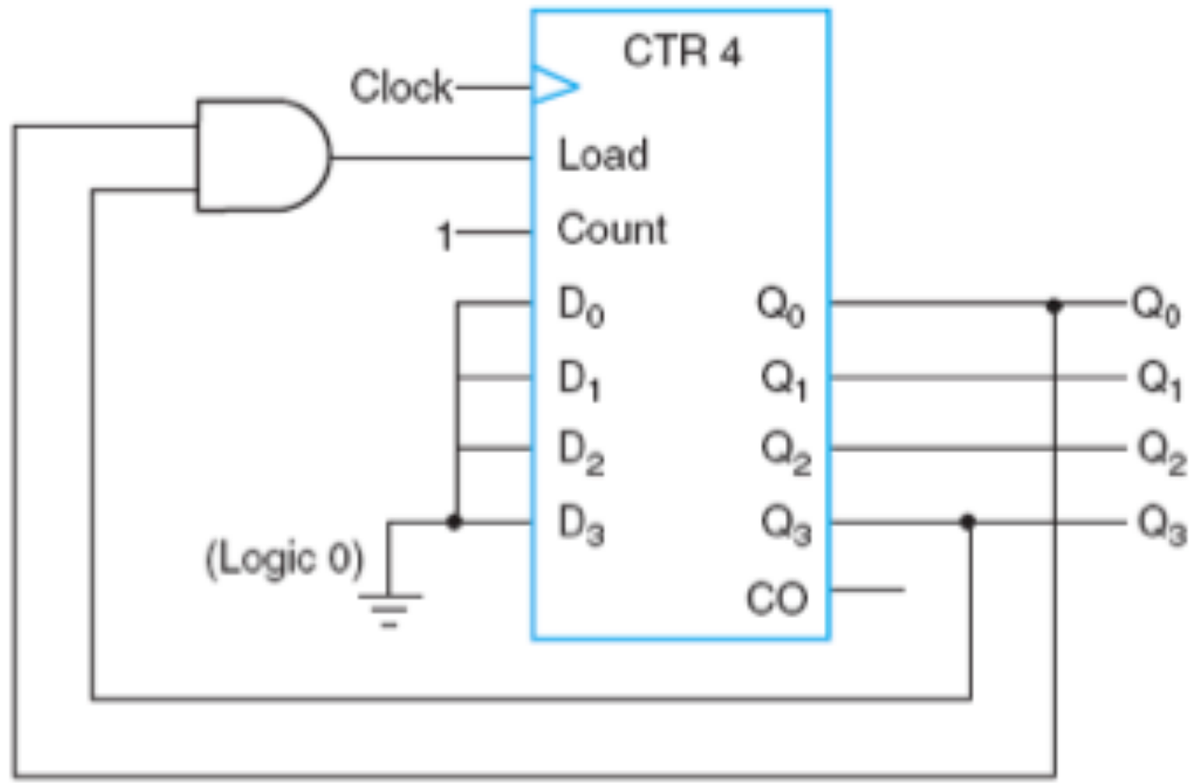
Contador binario con carga paralela





Diseño de circuitos secuenciales

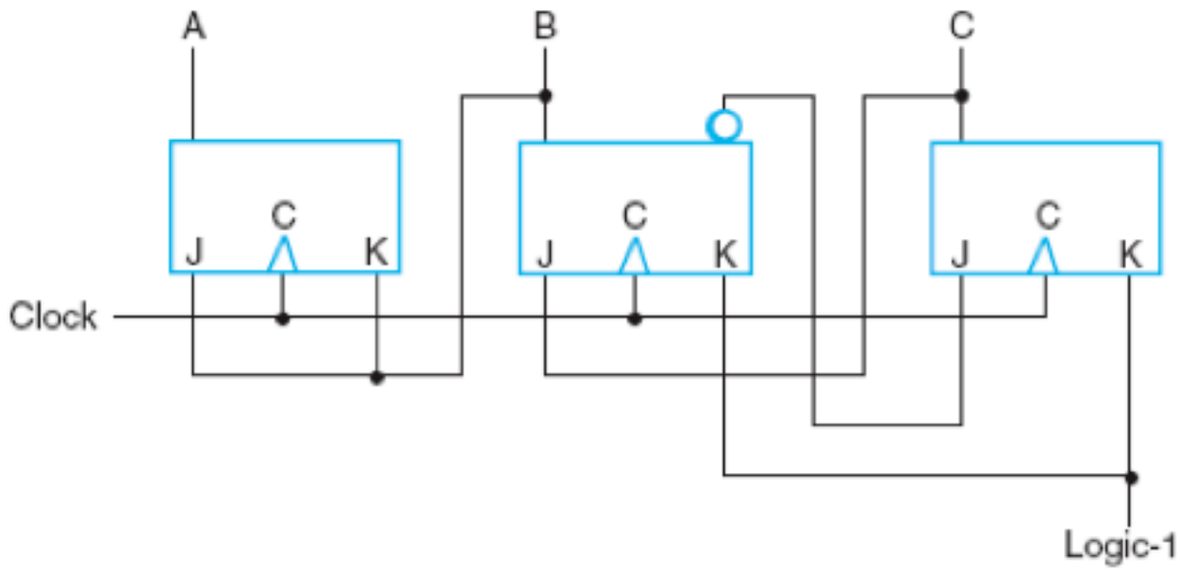
Contador BCD



Diseño de circuitos secuenciales

Contador arbitrario, implementación con JK

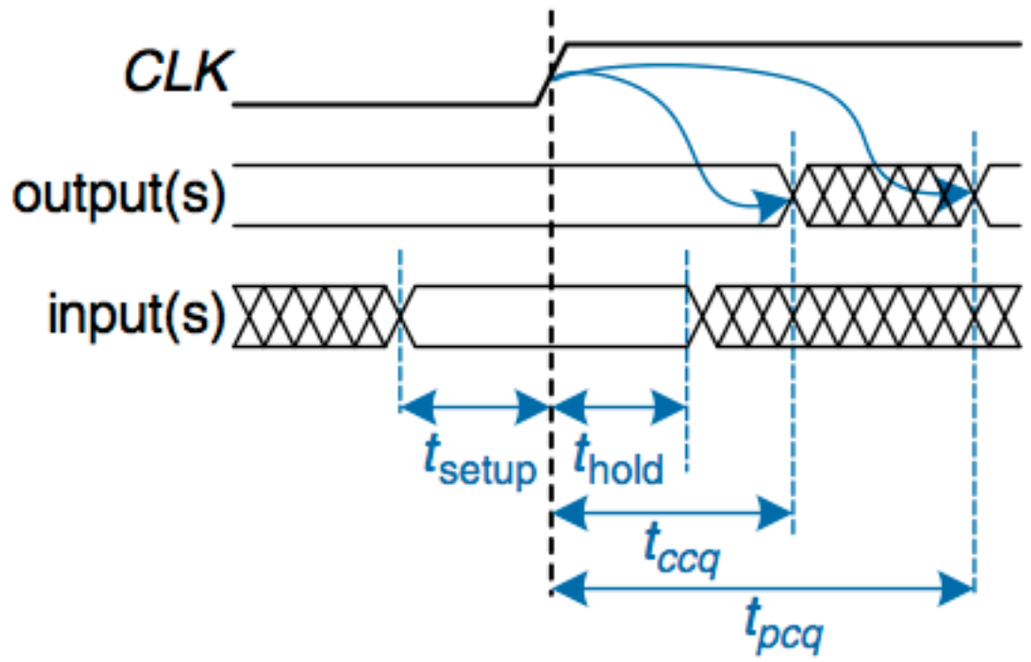
Present State			Next State			Flip-Flop Inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	×	0	×	1	×
0	0	1	0	1	0	0	×	1	×	×	1
0	1	0	1	0	0	1	×	×	1	0	×
1	0	0	1	0	1	×	0	0	×	1	×
1	0	1	1	1	0	×	0	1	×	×	1
1	1	0	0	0	0	×	1	×	1	0	×



Diseño de circuitos secuenciales

Temporización

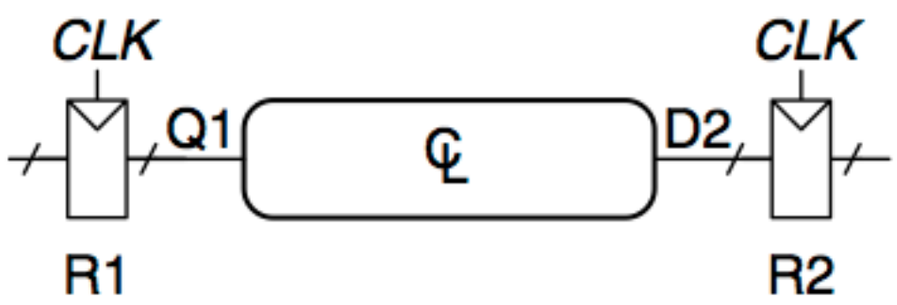
- Las entradas de un circuito secuencial síncrono deben mantenerse estables durante los tiempos de *setup* y de *hold* (tiempo de apertura) al rededor del flanco activo del *clock*.

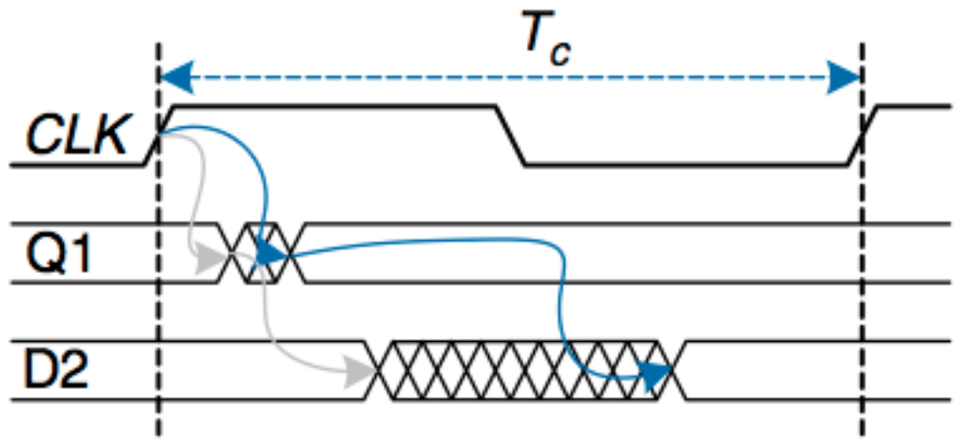


Diseño de circuitos secuenciales

Temporización

- Camino genérico en un circuito secuencial

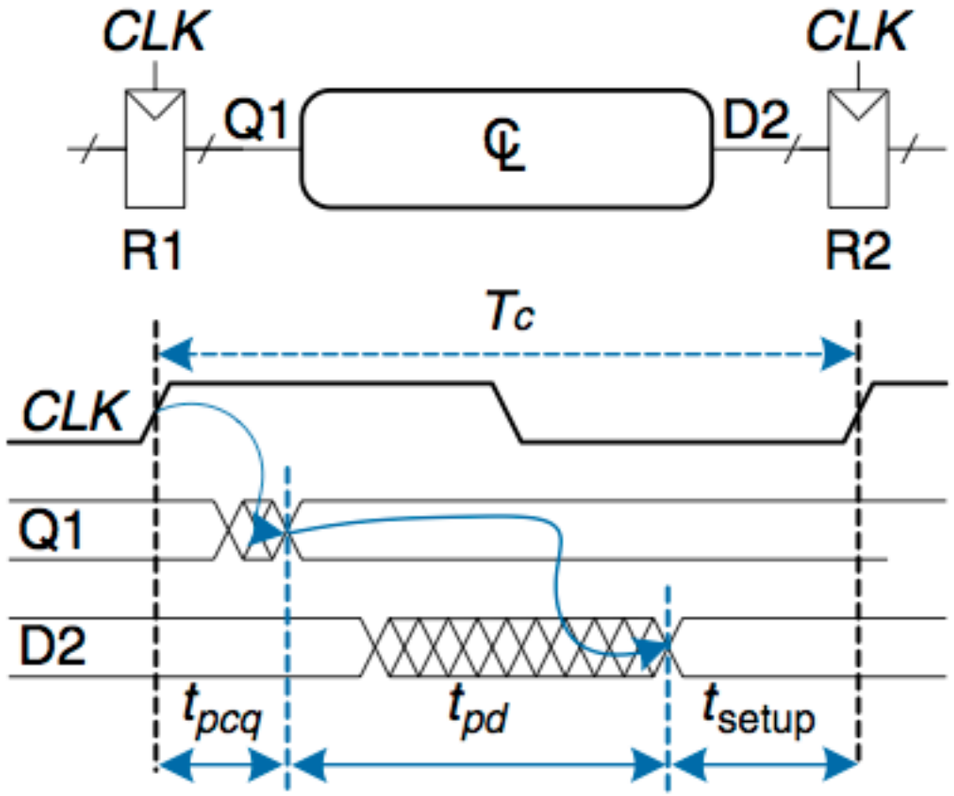




Diseño de circuitos secuenciales

Temporización

- Retardo máximo para la restricción de tiempo de



- Ecuación para el periodo mínimo del reloj

$$T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}}$$

Diseño de circuitos secuenciales

Temporización

- En circuitos comerciales el período del *clock* es determinado por el Director de Ingeniería (o el departamento de marketing)
- El retardo de propagación t_{pcq} y t_{setup} los especifica el fabricante.
- Normalmente la única variable bajo control del diseñador es el retardo de la lógica combinacional, entonces

$$t_{pd} \leq T_c - \underbrace{(t_{pcq} + t_{\text{setup}})}$$

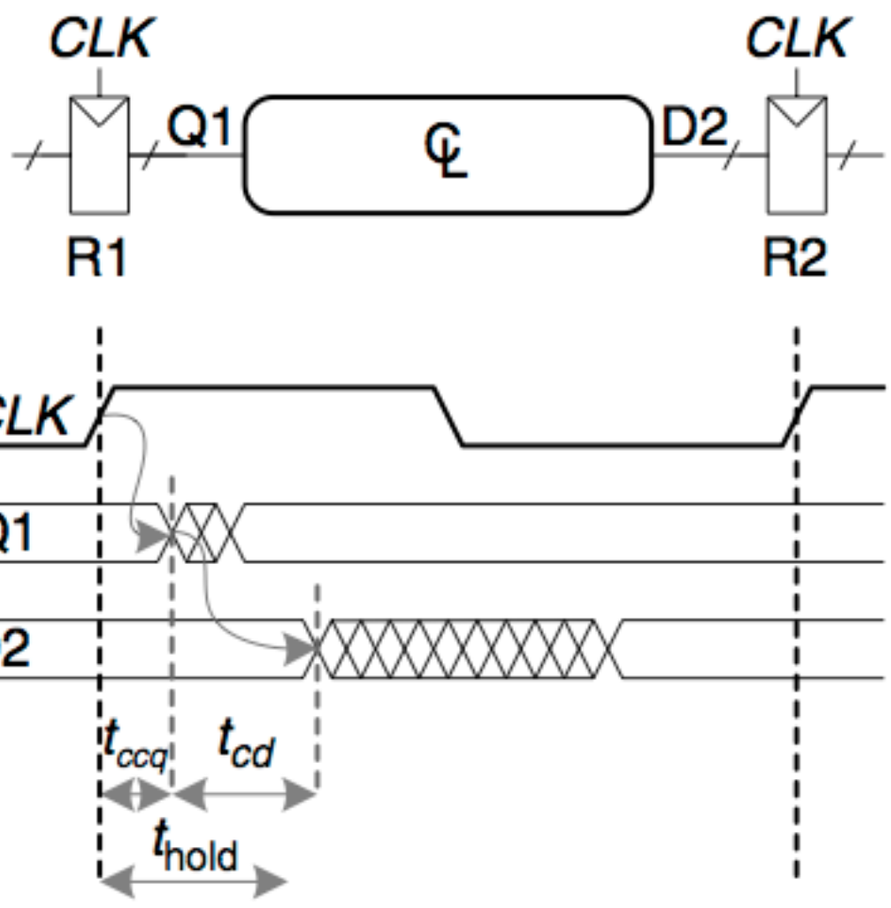
sequencing overhead

Diseño de circuitos secuenciales

Temporización

- Restricción de tiempo de *hold*.

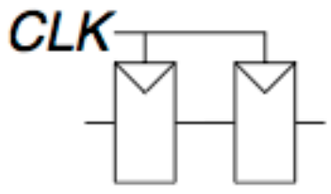
$$t_{ccq} + t_{cd} \geq t_{\text{hold}}$$



Recordar que t_{ccq} y t_{hold} son características impuestas por el fabricante. Rearreglando tenemos que

$$t_{cd} \geq t_{\text{hold}} - t_{ccq}$$

Esta se llama restricción *min-delay* porque limita el retardo mínimo de la lógica combinacional



En este caso $t_{cd} = 0$, entonces

$$t_{\text{hold}} \leq t_{ccq}$$

Luego, un *flip-flop* confiable debe tener un tiempo de *hold* menor que su retardo de contaminación.

Normalmente los *flip-flops* se diseñan con un $t_{\text{hold}} = 0$

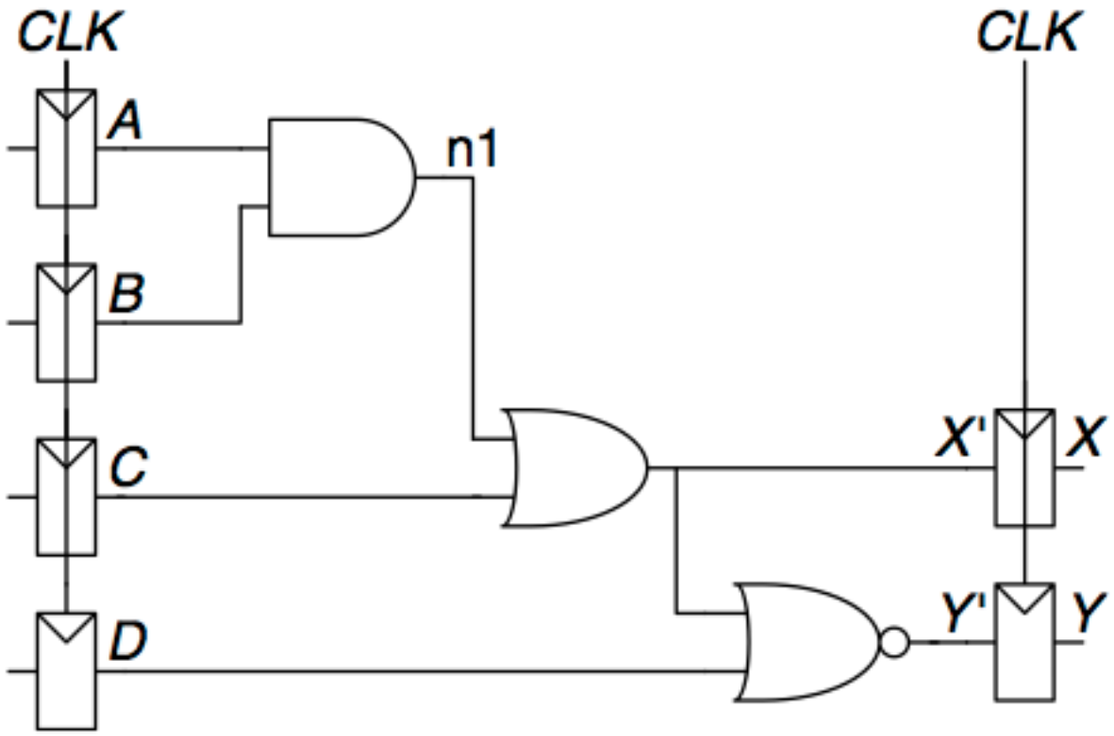
Diseño de circuitos secuenciales

Temporización

- En general los ciruitos secuenciales tienen restricciones de *setup* y de *hold* que dictan los retardos máximos y mínimos de la lógica combinacional entre *flip-flops*.
- Los *flip-flops* modernos son diseñados de tal forma que el mínimo retardo a través de la lógica combinacional puede ser cero.
- La restricción de retardo máximo limita el número de compuertas consecutivas en el camino crítico de un circuito de alta velocidad.

Diseño de circuitos secuenciales

Ejemplo de análisis temporal



$$t_{ccq} = 30\text{ps}$$

$$t_{pcq} = 80\text{ps}$$

$$t_{\text{setup}} = 50\text{ps}$$

$$t_{\text{hold}} = 60\text{ps}$$

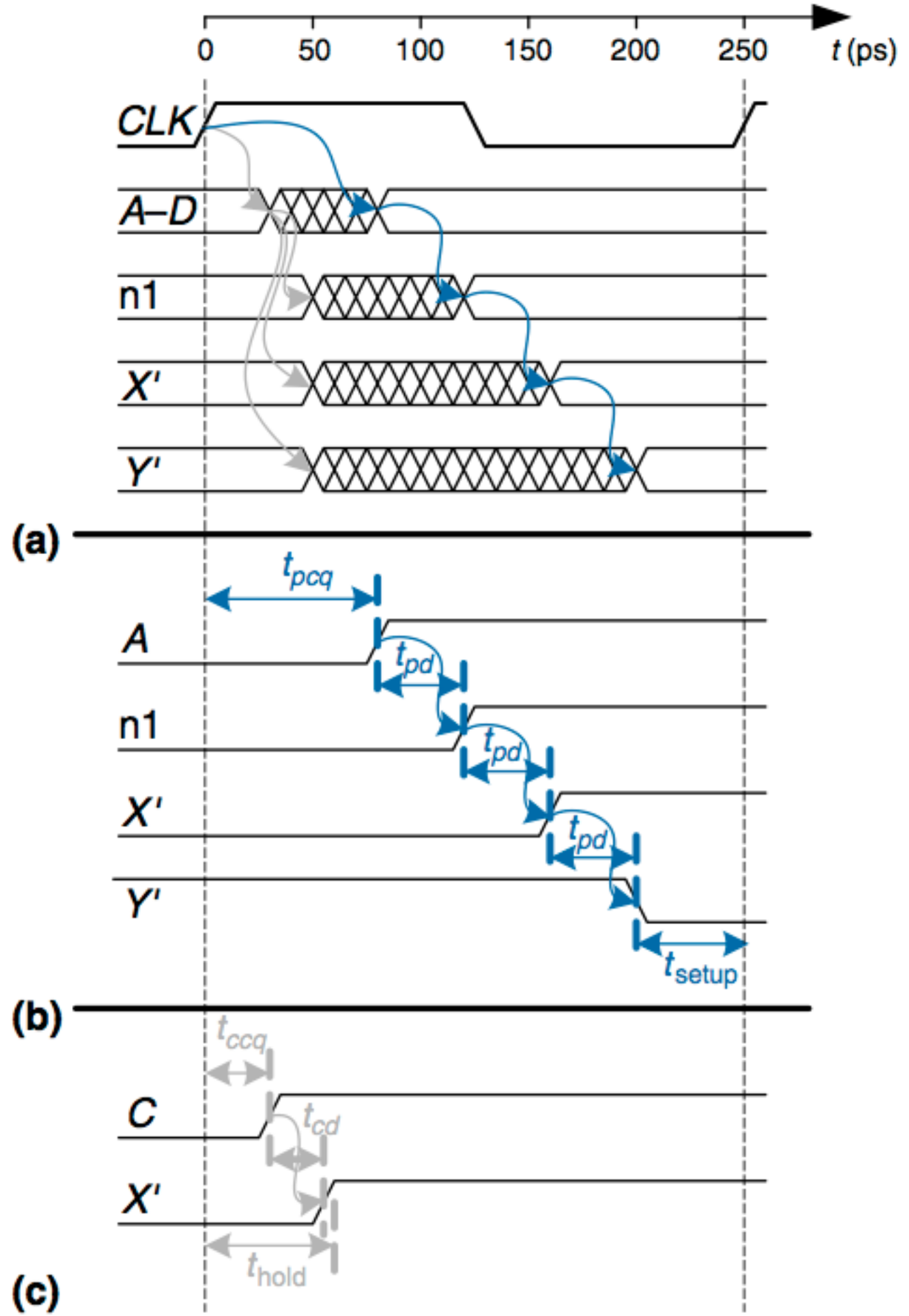
Cada compuerta lógica tiene un t_c = 25 ps y un t_p = 40 ps.

$$T_c \geq t_{pcq} + 3\,t_{pd} + t_{\text{setup}} = 80 + 3 \times 40 + 50 = 250\text{ ps}$$

$$f_c = 1/T_c = 4\text{ GHz.}$$

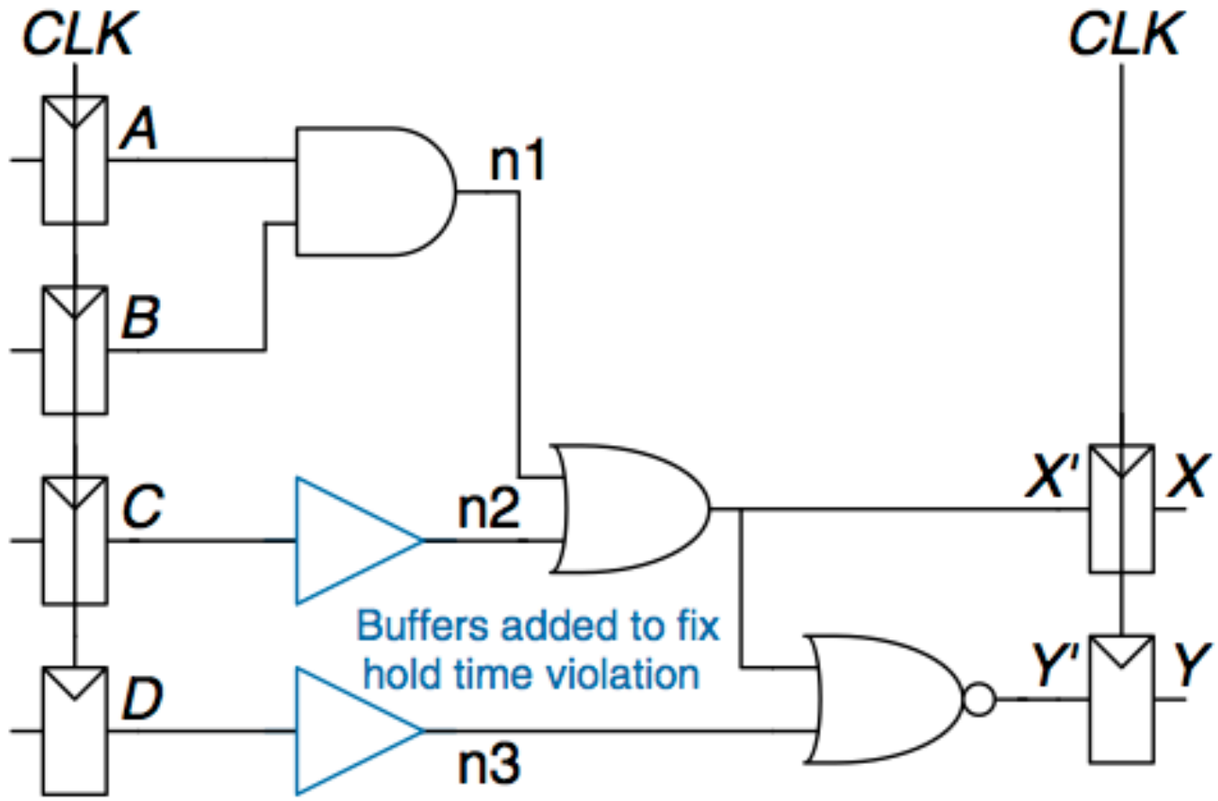
Diseño de circuitos secuenciales

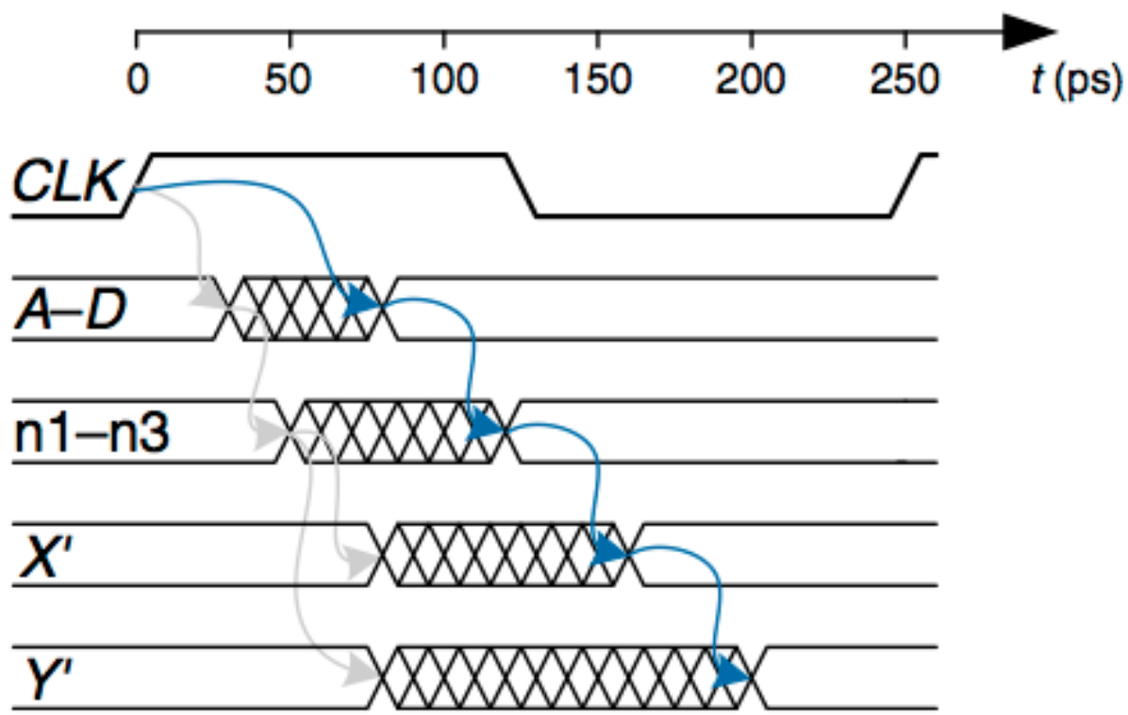
Ejemplo de análisis temporal



Diseño de circuitos secuenciales

Corrección de la violación de tiempo de *hold*.

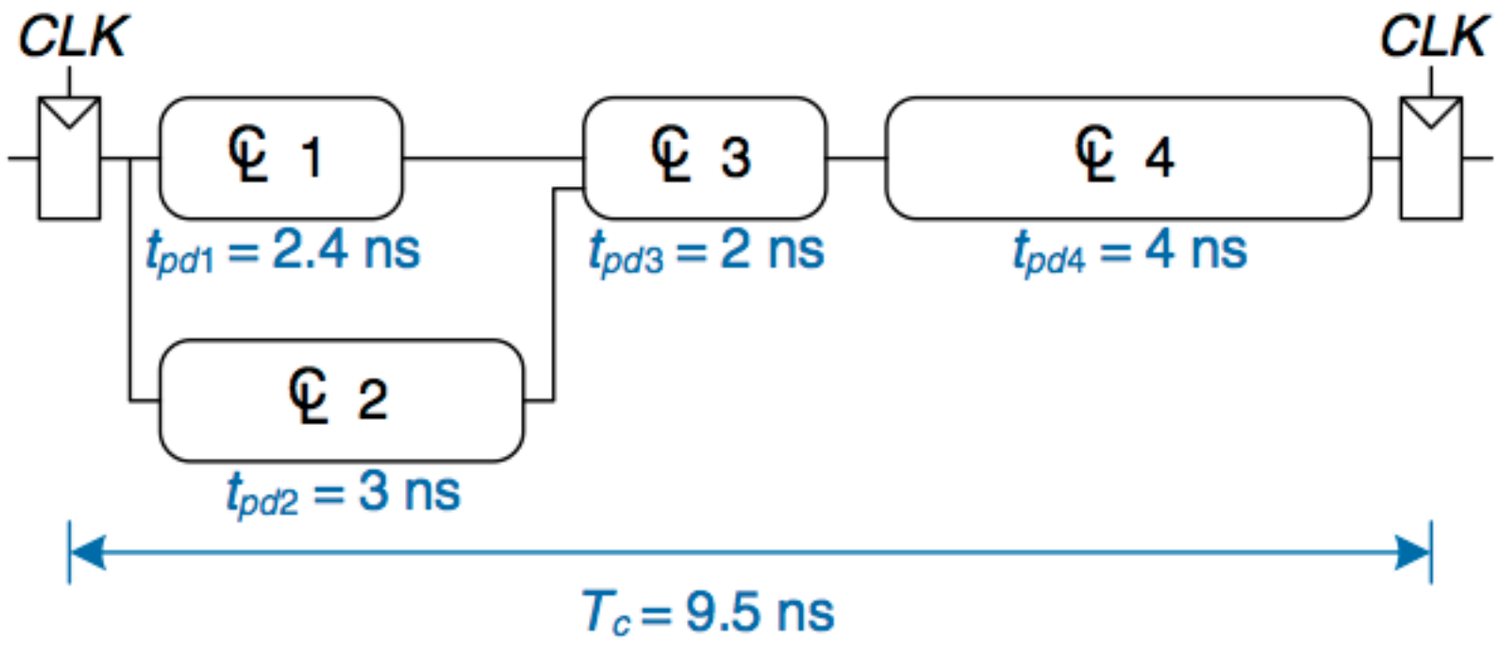




Diseño de circuitos secuenciales

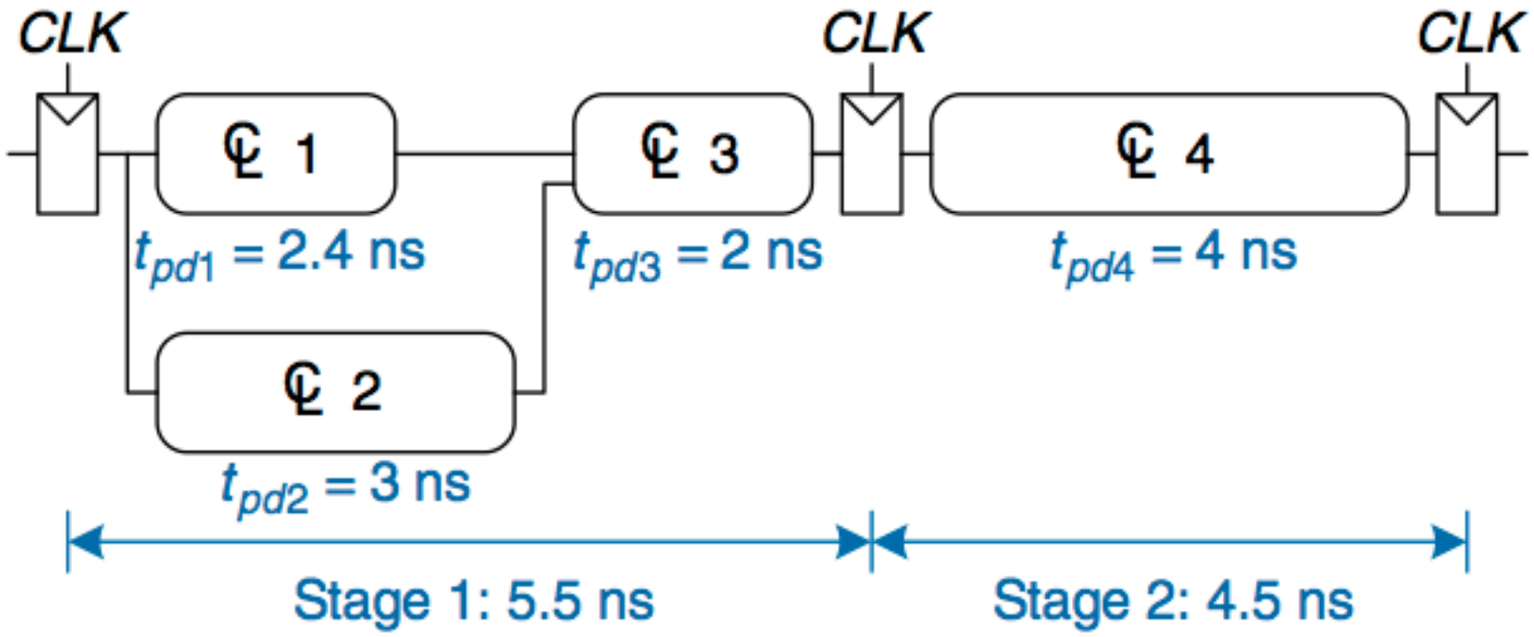
Pipelining para mejorar la velocidad en una máquina de estados finita.

Circuito sin pipelining



t_{pcq}=0.3, t_{setup}=0.2

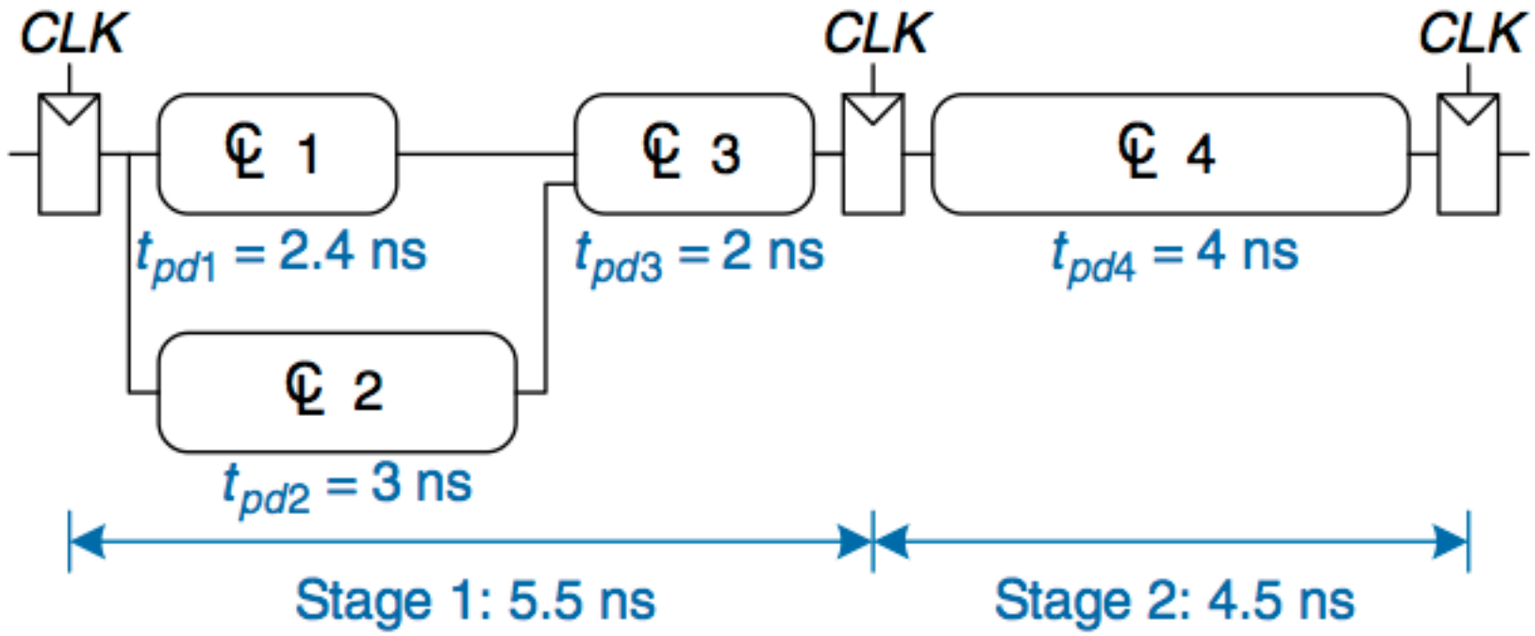
Circuito con dos etapas de pipelining



Diseño de circuitos secuenciales

Pipelining para mejorar la velocidad en una máquina de estados finita.

Circuito con dos etapas de pipelining



t_{pcq}=0.3, t_{setup}=0.2

Circuito con tres etapas de pipelining