7. Flip-flops, registros, contadores y un procesador simple

Mis agradecimientos a Tomás Ladrix por la ayuda en la preparación de estas láminas

angel@uc.cl

IEE2713 Sistemas Digitales

7.1 – 7.7

INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

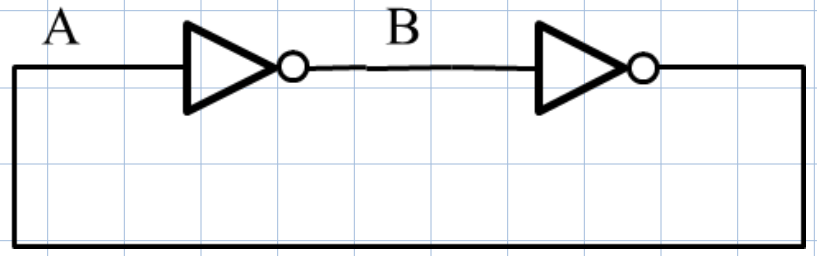
Terminología de elementos con memoria

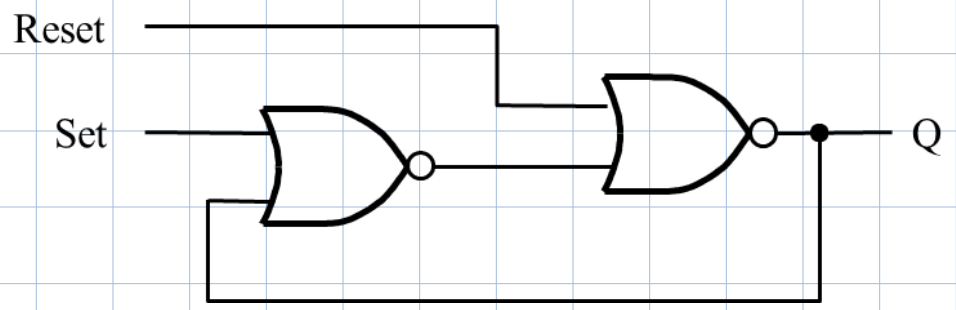
angel@uc.cl

IEE2713 Sistemas Digitales

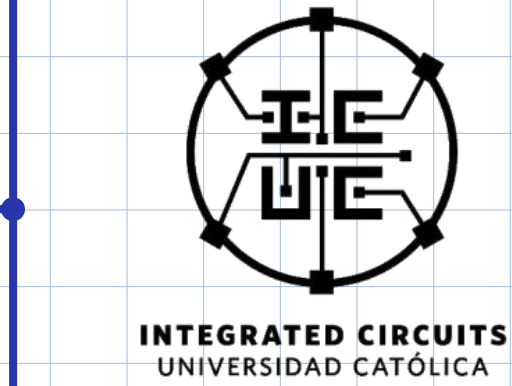
7.1 Circuitos con Memoria

- Circuitos Combinacionales:
  - La salida depende solo de la entrada
- Circuitos Secuenciales:
  - La salida también depende del “estado” del circuito
  - Estos circuitos incluyen elementos con memoria para definir el estado en el que están
    - Este estado es cambiado por las entradas al circuito
- La idea de una memoria es un circuito que mantiene su valor en el tiempo:

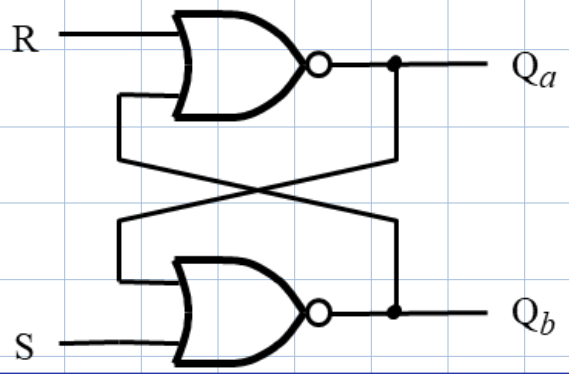




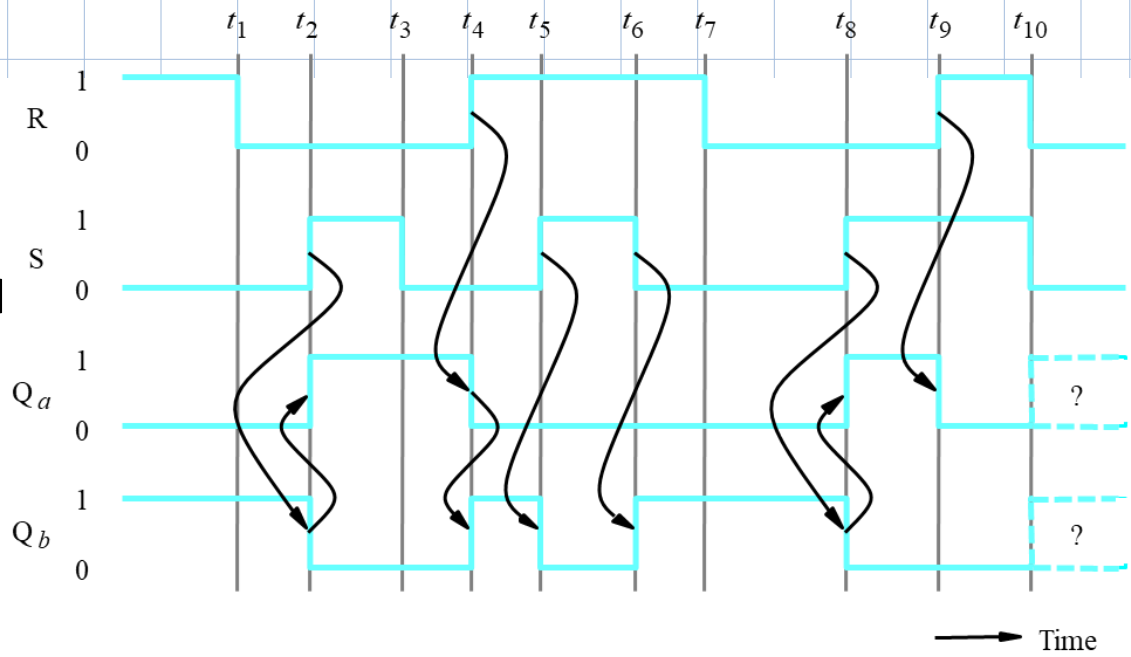
7.1 Latch básico



- Un elemento con memoria es capaz de mantener su estado
- Una implementación con compuertas NOR es conocida como un **latch básico**
- Dos entradas que permiten cambiar el estado del latch:
  - **S (set)**: Cuando se activa **S**, "setea" o establece el valor de **Q** en 1
  - **R (reset)**: Cuando se activa **R**, "resetea" o borra el valor de **Q** en 0



S	R	Q_a	Q_b	
0	0	0/1	1/0	(no change)
0	1	0	1	
1	0	1	0	
1	1	0	0	

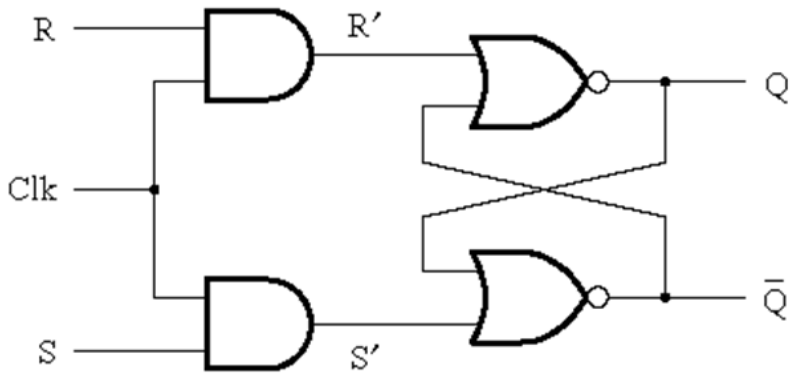


En este diagrama de tiempo, asumimos que los NOR no tienen retardo

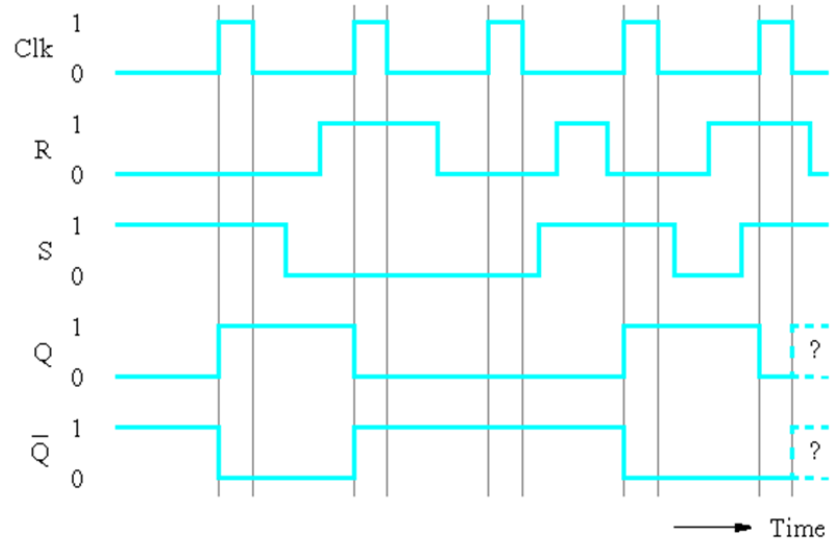
7.2 Latch SR comandado por compuerta

- Nos gustaría implementar una compuerta de memoria que cambie su valor sólo cuando se necesita
- En este caso **Clk** cumple la función de **enable**
  - Sólo cambia el valor del estado **Q mientras Clk = 1**
- El input **Clk (clock)** es muy importante y se verá en mayor profundidad más adelante

¿Qué pasa cuando S = R = 1?



Clk	S	R	Q(t + 1)
0	x	x	Q(t) (no change)
1	0	0	Q(t) (no change)
1	0	1	0
1	1	0	1
1	1	1	x

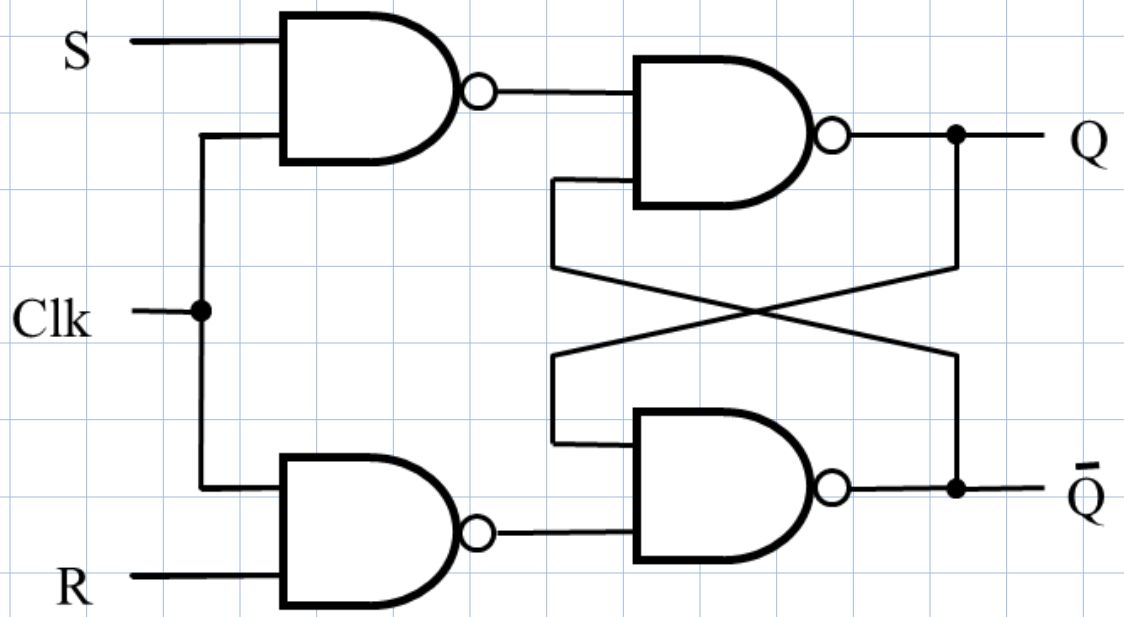


7.2.1 Latch SR con compuertas NAND

INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

Otra implementación posible se consigue utilizando compuertas NAND

Clk	S	R	Q(t + 1)
0	x	x	Q(t) (no change)
1	0	0	Q(t) (no change)
1	0	1	0
1	1	0	1
1	1	1	x



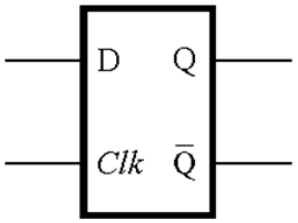
7.3 Latch D comandado por compuerta

INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

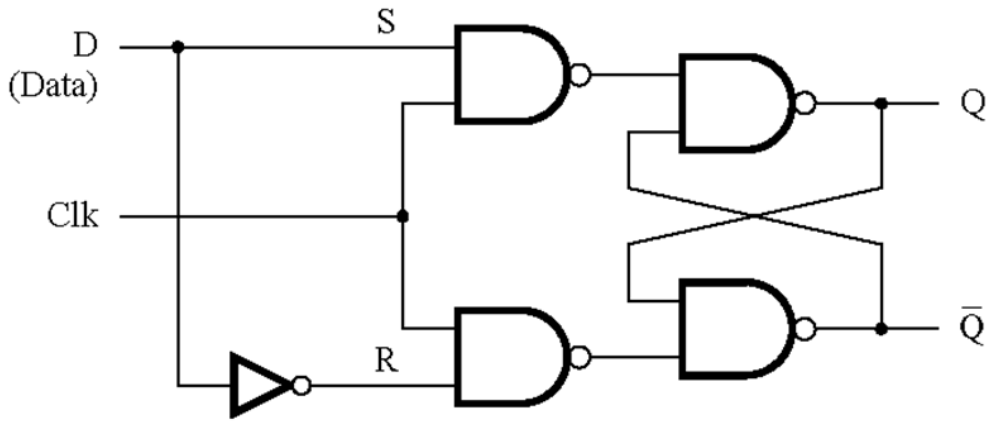
- Bloque muy común y mucho más útil en la práctica, que un latch SR
- Sólo tiene 1 input D en lugar de SR
  - Cuando Clk = 1, **Q** almacena el valor de **D**
  - Cuando Clk = 0, **Q** no cambia
- Con esta implementación, evitamos el problema que ocurre cuando **S = R = 1**

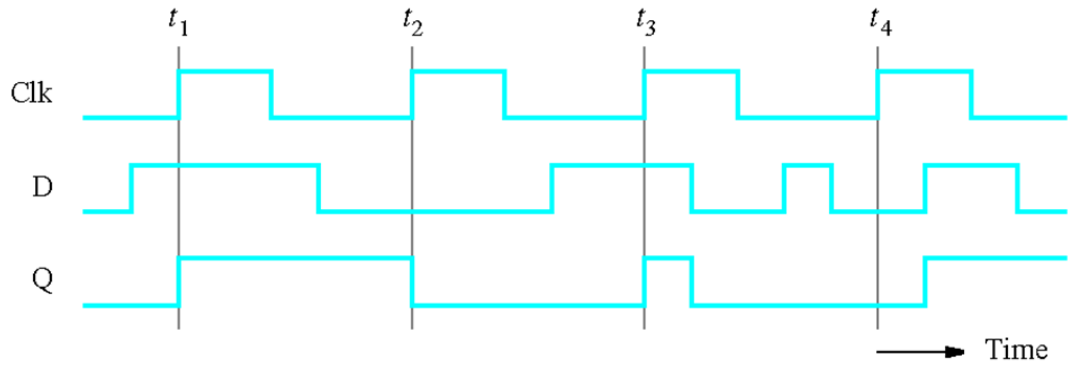
Clk	D	$Q(t+1)$
0	x	$Q(t)$
1	0	0
1	1	1

(b) Characteristic table

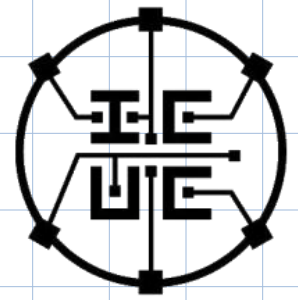


(c) Graphical symbol



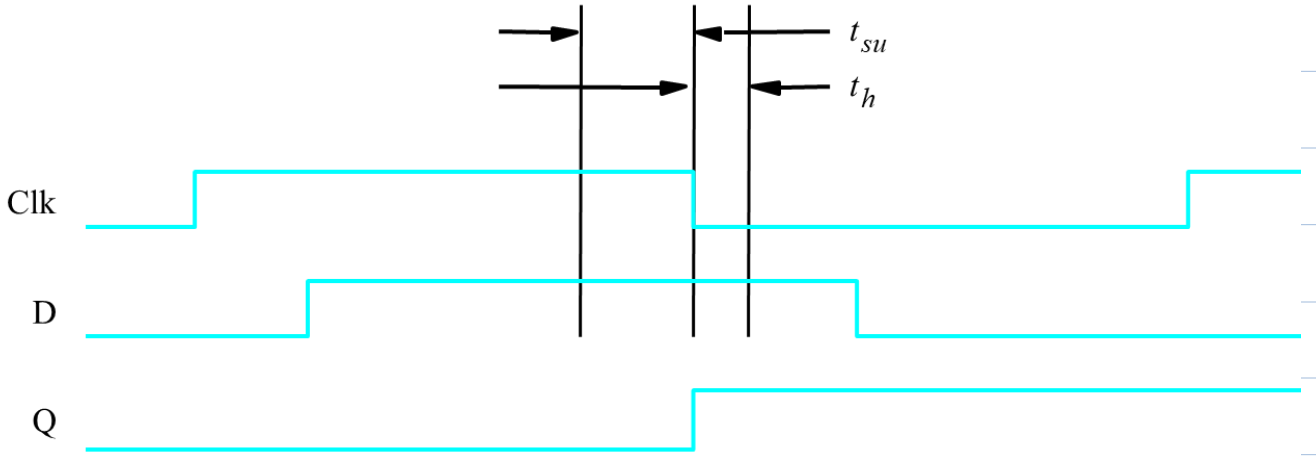


7.3.1 Efectos de retardos de propagación

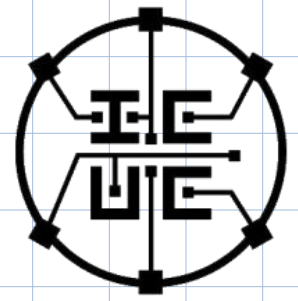


- El cambio de **Clk** no es instantáneo, por lo que es necesario asegurarse que la señal **D** esté estable, para evitar comportamiento errático
  - $\mathbf{t_{su}}$ (set-up time) corresponde al mínimo tiempo que la señal **D** debe estar estable **antes** de que **Clk** se apague
  - $\mathbf{t_h}$ (hold time) corresponde al mínimo tiempo que **D** se debe mantener constante **después** del apagado de **Clk**

- Estos valores dependen de la tecnología utilizada y son especificadas por el fabricante
- Para chips discretos, valores típicos son $\mathbf{t_{su}}$ **= 20ns** y $\mathbf{t_h}$ **= 3ns**
- En tecnologías modernas, pueden ser t_{su} = 10ps y t_h = 2ps



7.4 Flip-flops D maestro-esclavo y gatillados por flanco

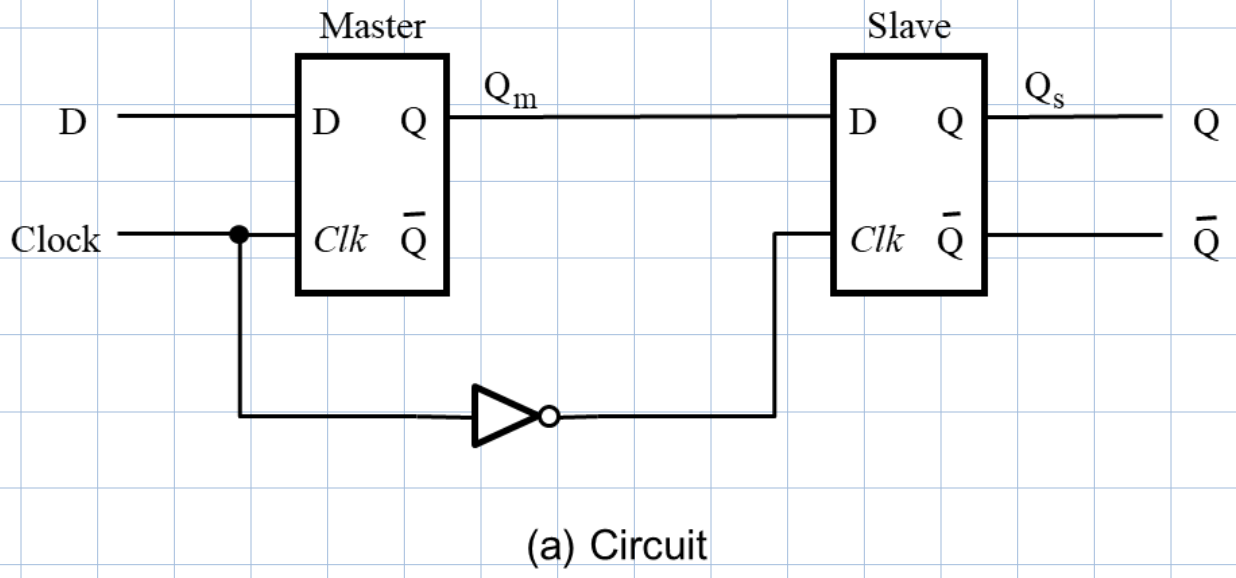


- Los **latch** cambian su estado libremente siempre y cuando **Clk = 1**
  - Sería ideal habilitar la actualización de estado **solo una vez por ciclo** de clock
- El bloque circuital **flip-flop** se comporta de esta manera
  - Sólo actualizan su valor al momento del cambio del clock
  - Algunos flip-flops operan con flancos positivos de reloj, otros con flanco negativo
    - Cuando Clk cambia de 0 a 1 se llama flanco de subida o flanco positivo
    - Cuando Clk cambia de 1 a 0 se llama flanco de bajada o negativo

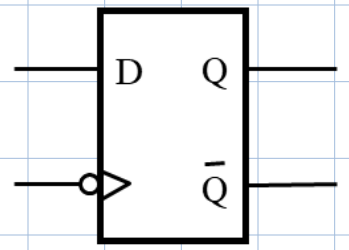
7.4.1 Flip-flop D maestro-esclavo

INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

- Una posible implementación de flip-flop D está compuesta por dos latches
  - El **latch maestro** cambia mientras **Clk = 1**
  - El **latch esclavo** cambia mientras **Clk = 0**



(a) Circuit



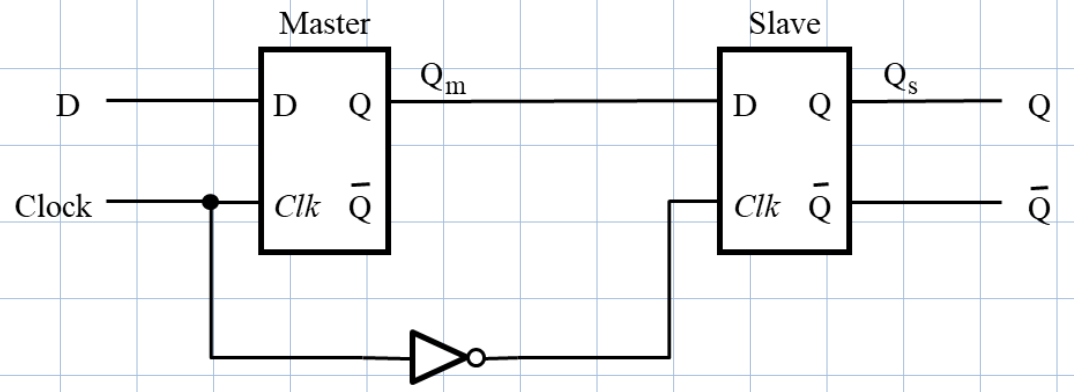
(c) Graphical symbol

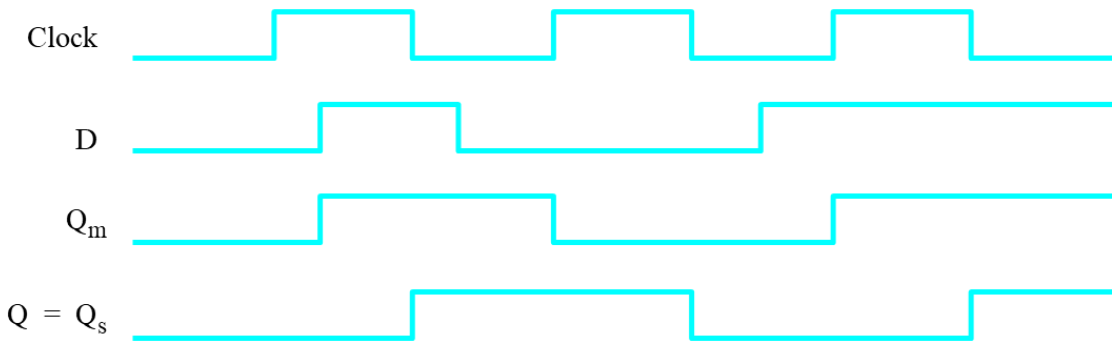
¿Cómo funciona?

7.4.2 Flip-flop D gatillado por flanco

INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

- El circuito consiste en dos latches tipo D llamados **maestro** y **esclavo**
  - Si **Clk = 1**, el latch **maestro (Qm)** sigue al input **D** mientras que el latch **esclavo** no cambia
  - Si **Clk = 0**, el latch **esclavo (Qs)** sigue el valor de **Qm**, mientras que l latch **maestro** no cambia
- Como **Qm** se mantiene constante mientras **Clk = 0**, entonces el valor el **latch escalvo** sigue a un valor que no cambia
- **Qs** se queda con el valor que tenía **Qm** justo cuando el clock cambia de 1 a 0
- A este circuito se le llama **flip-flop tipo D gatillado por flanco negativo**

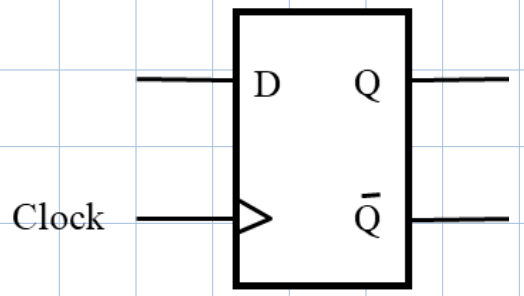




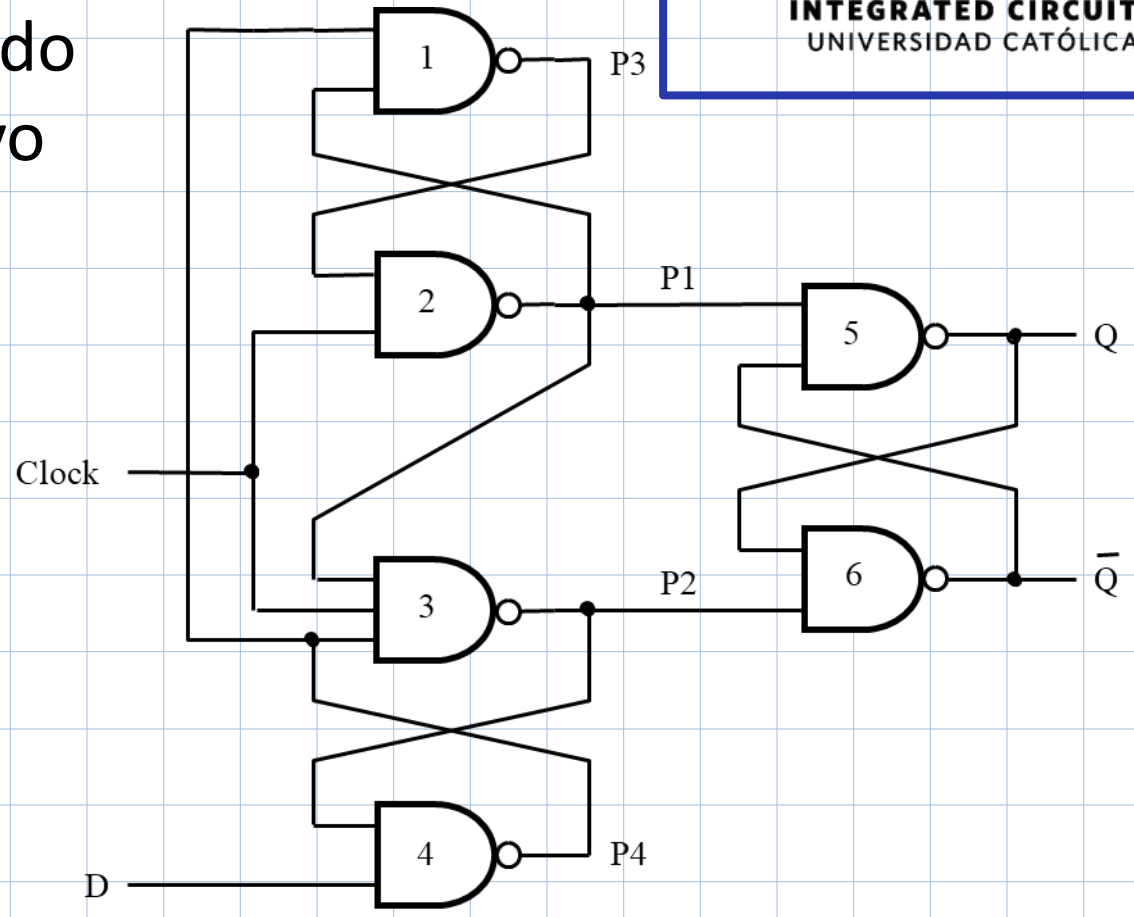
7.4.2 Flip-flop D gatillado por flanco positivo

INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

Esta es otra forma de implementar un flip-flop gatillado por flanco, en este caso, positivo



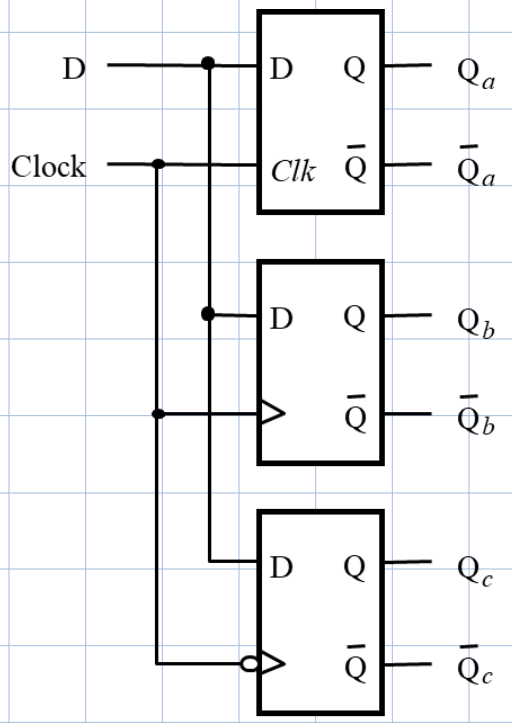
(b) Graphical symbol

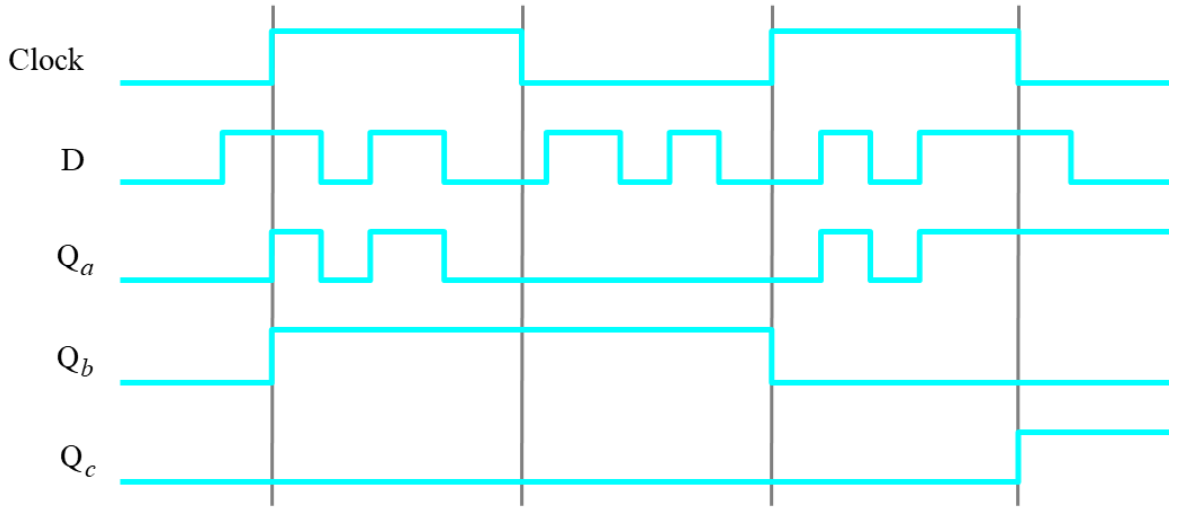


¿Qué pasa si cambiamos las compuertas NAND por compuertas NOR?

7.4.2 Comparación entre Latch y Flip-Flops

- El primero corresponde a un latch (cambia **mientras** clk = 1)
- El segundo es un flip-flop D gatillado por **flanco positivo**
- El tercero es un flip-flop D gatillado por **flanco negativo**





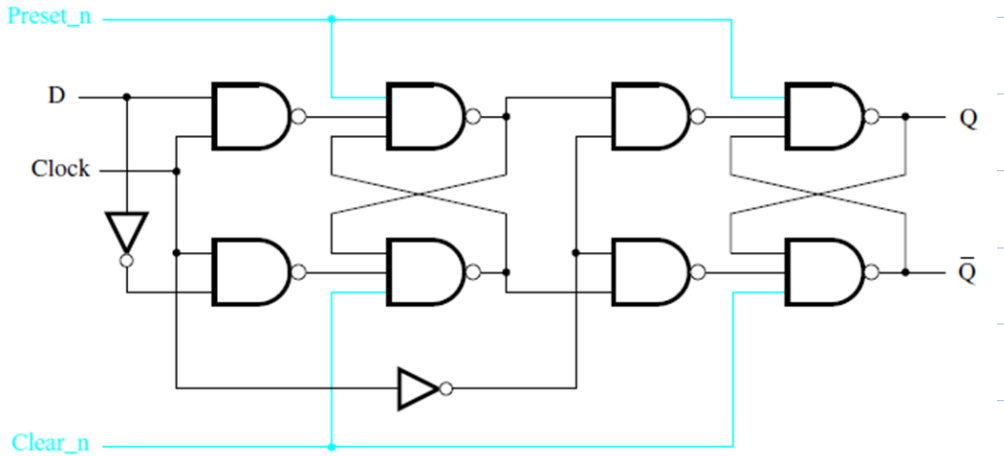
(b) Timing diagram

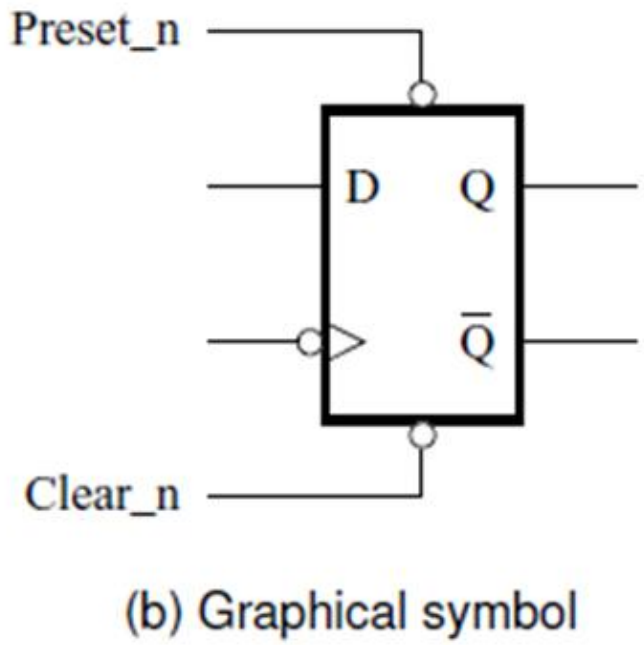
7.4.3 Flip-flops D con clear y preset

INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

En algunos circuitos es útil establecer los valores de sus flip-flops de manera externa y **asíncrona** mediante entradas **preset** y **clear**

- Cuando **Preset_n = 0** entonces **Q = 1**
  - En cambio, **Preset_n = 1** no tiene efecto
- Cuando **Clear_n = 0** entonces **Q = 0**
  - En cambio, **Clear_n = 1** no tiene efecto.

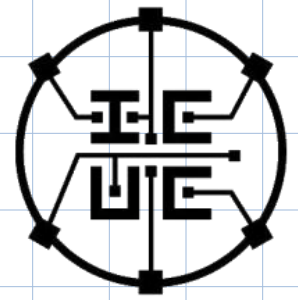




(b) Graphical symbol

¿Qué significa "de manera asíncrona"?

Código para flip-flop con reset



- Notar que la magia del flip-flop ocurre dentro de un bloque always
- La asignación <= se llama nonblocking assignment
  - Consideremos sólo asignaciones dentro de bloques always
    - La asignación = (blocking assignment) se ejecuta en serie
    - La asignación <= (nonblocking assignment) se ejecuta en paralelo
    - No accesar fuera de un bloque always registros asignados con blocking assignment
    - No mezclar ambos tipos de asignación en un bloque always

```
module flipflop (D, Clock, Resetn, Q);
    input D, Clock, Resetn;
    output reg Q;

    always @(posedge Clock)
        if (!Resetn)
            Q <= 0;
        else
            Q <= D;

endmodule
```

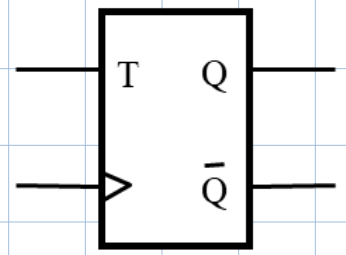
7.5 Flip-flop T

- Cuando **T = 0**, **Q** mantiene su estado
- Cuando **T = 1**, el estado siguiente será el complemento del estado actual
- Se dice que este flip-flop realiza un “toggle” cuando **T = 1**
  - Muy útil para hacer contadores

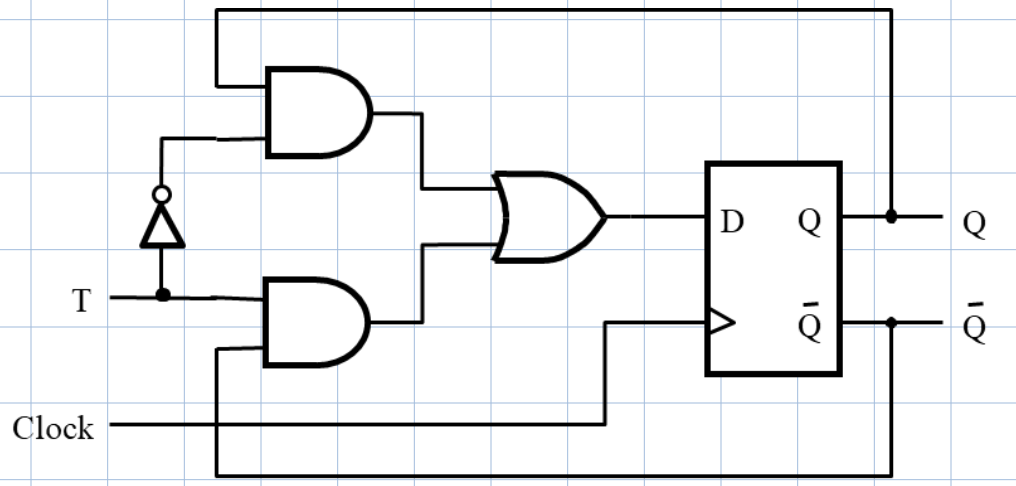
INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

T	$Q(t+1)$
0	$Q(t)$
1	$\bar{Q}(t)$

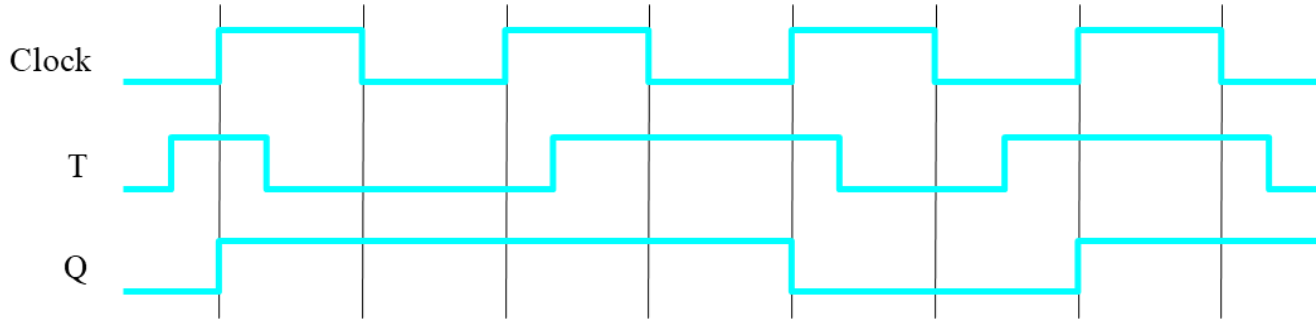
(b) Truth table



(c) Graphical symbol



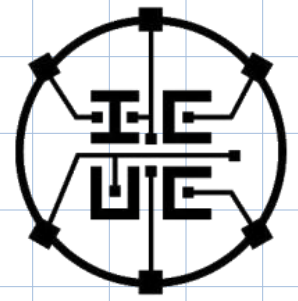
(a) Circuit



(d) Timing diagram

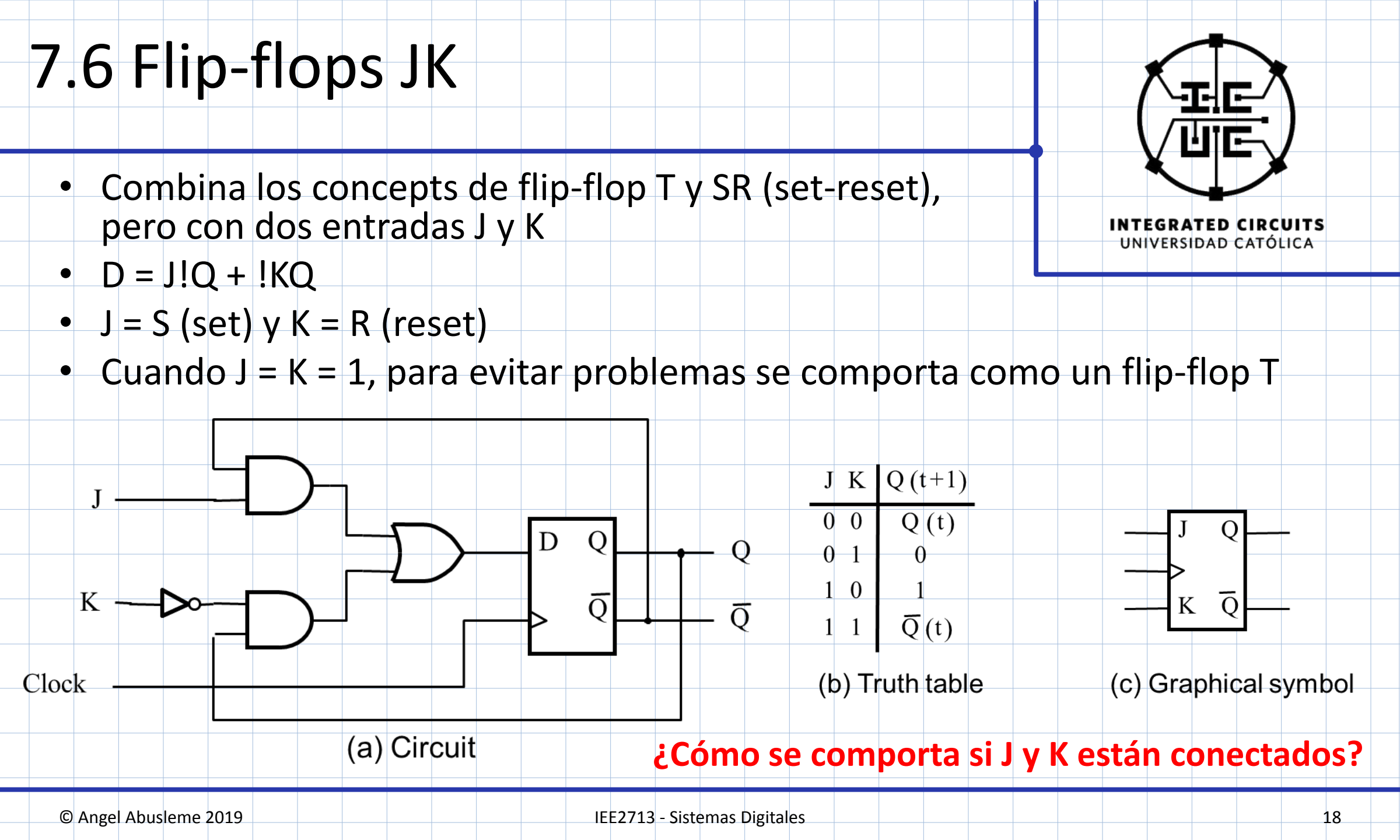
¿Es síncrono o asíncrono?

7.5.1 Flip-flops configurables



- Es posible que, dependiendo del circuito, prefiramos usar diferentes tipos de flip-flops
- En algunos PLDs, los flip-flops son configurables
  - Es posible elegir el tipo de flip-flop que queremos usar

7.6 Flip-flops JK



- Combina los concepts de flip-flop T y SR (set-reset), pero con dos entradas J y K
- D = J!Q + !KQ
- J = S (set) y K = R (reset)
- Cuando J = K = 1, para evitar problemas se comporta como un flip-flop T

(a) Circuit

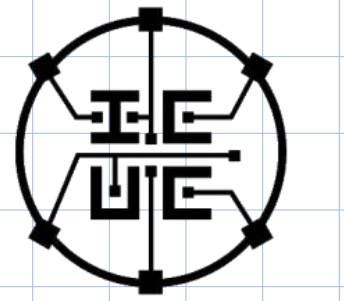
J	K	Q (t+1)
0	0	Q (t)
0	1	0
1	0	1
1	1	$\overline{Q}$ (t)

(b) Truth table

(c) Graphical symbol

¿Cómo se comporta si J y K están conectados?

7.7 Resumen de la terminología



- **Latch básico:** conexión con realimentación entre dos compuertas NOR o dos NAND que guardan 1 bit de información
  - Almacena un 1 cuando S = 1 y almacena un 0 cuando R = 1
- **Latch comandado por compuerta:** no puede cambiar su estado cuando Clk = 0. Sólo puede cambiar cuando el clock está en 1
  - Latch tipo SR: Comandado por Set y Reset
  - Latch tipo D: Guarda el valor de D directamente
- **Flip-Flop:** Su valor almacenado puede ser cambiado solo en un flanco de subida (positivo) o bajada (negativo)

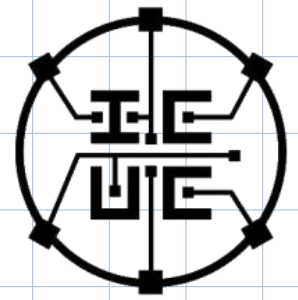
INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

Registros y contadores

angel@uc.cl

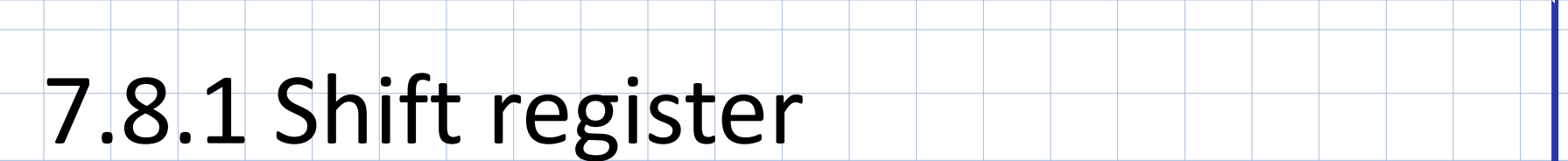
IEE2713 Sistemas Digitales

7.8 Registros

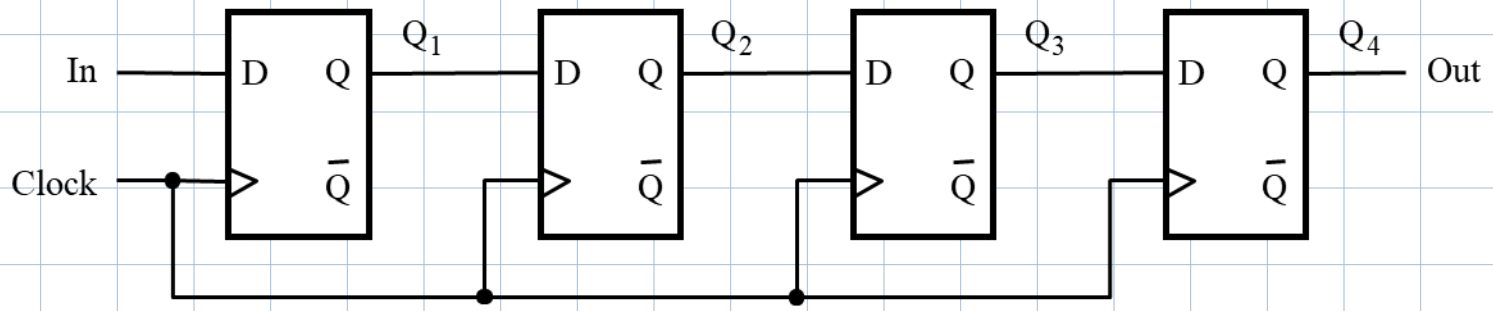


- Un flip-flop almacena un solo bit
- Para almacenar **n** bits, se usan **n** flip-flops bajo un mismo clock
- Nos referimos a estos **n** flip-flops como **un** registro
  - Ejemplo: 8 flip-flops forman un registro que almacena un byte (8 bits)

7.8.1 Shift register

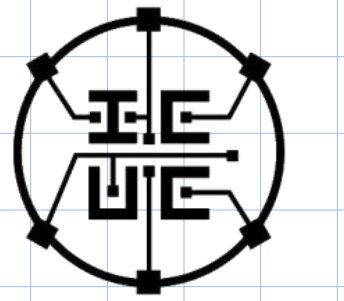


- Un shift register es un conjunto de flip-flops en cascada, que propagan la señal del primero
  - Esta señal avanza en cada flanco del clock
- Para el diagrama, si hago **In = 1**, me demoro **4** ciclos de clock en ver ese mismo 1 en **Q4**
- También puede verse usarse como un retardo de la señal

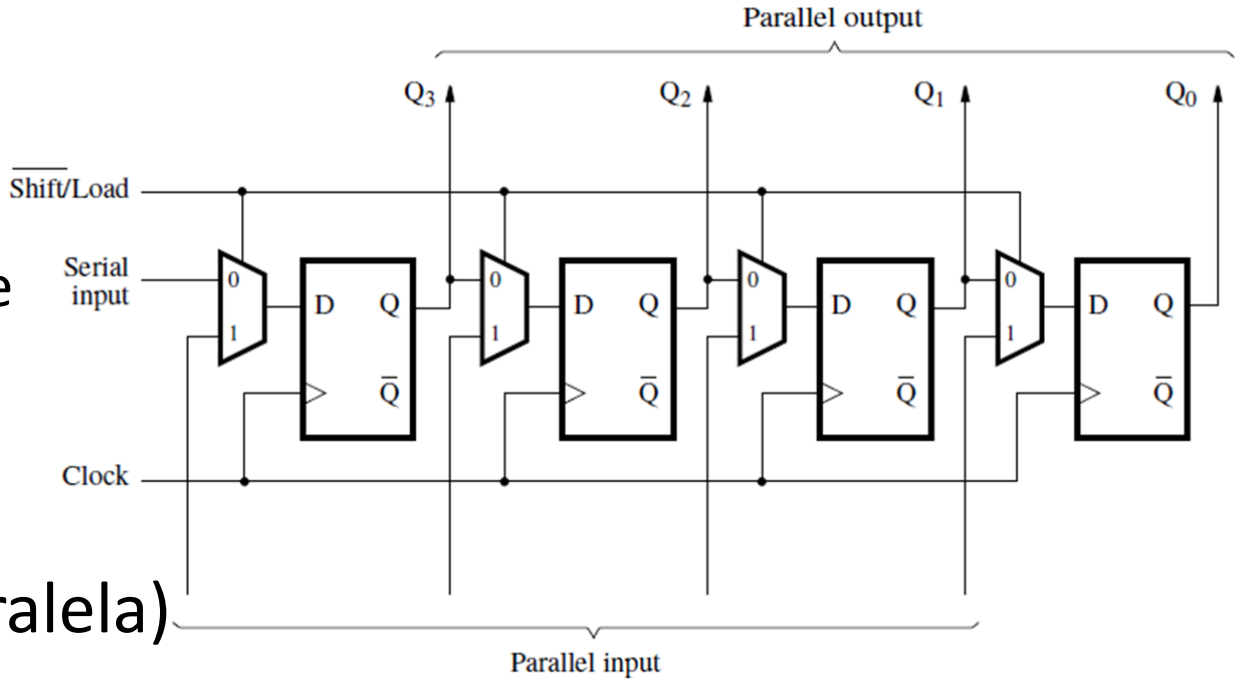


	In	Q_1	Q_2	Q_3	Q_4 = Out
t_0	1	0	0	0	0
t_1	0	1	0	0	0
t_2	1	0	1	0	0
t_3	1	1	0	1	0
t_4	1	1	1	0	1
t_5	0	1	1	1	0
t_6	0	0	1	1	1
t_7	0	0	0	1	1

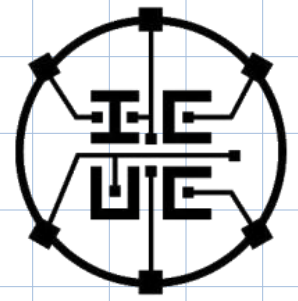
7.8.2 Shift register de acceso paralelo



- Para transferir **n** bits desde y hacia un shift register, se requiere **n** ciclos de clock **(transmisión serial)**
- Usando **n** cables lo podemos hacerlo en un solo ciclo **(transmisión paralela)**
  - Si Shift/Load = 0, el circuito opera como un shift register (serial)
  - Si Shift/Load = 1, los bits se guardan de forma **paralela** en un solo ciclo de clock.
- Para acceder a los valores guardados, podemos leer **Q0** (serial) o todos los valores **Qn** (paralela)

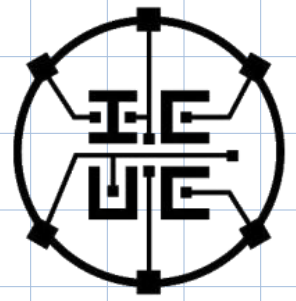


7.9 Contadores

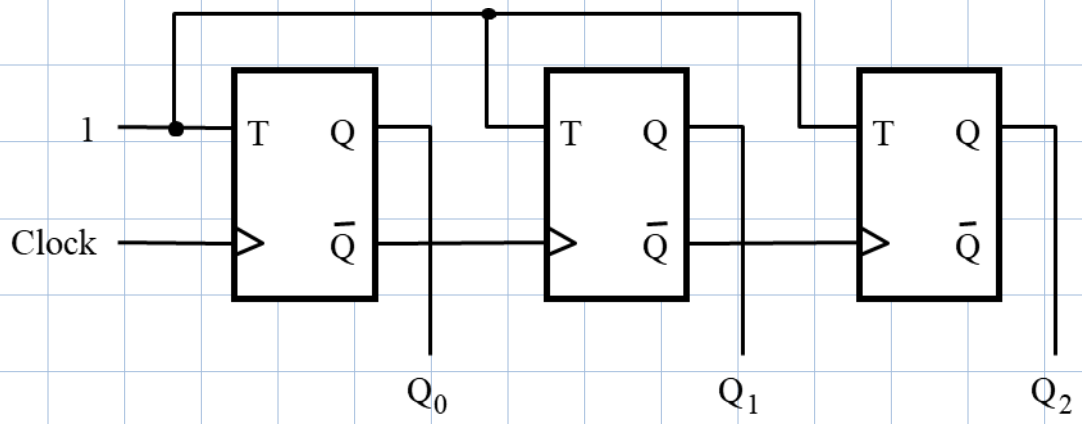


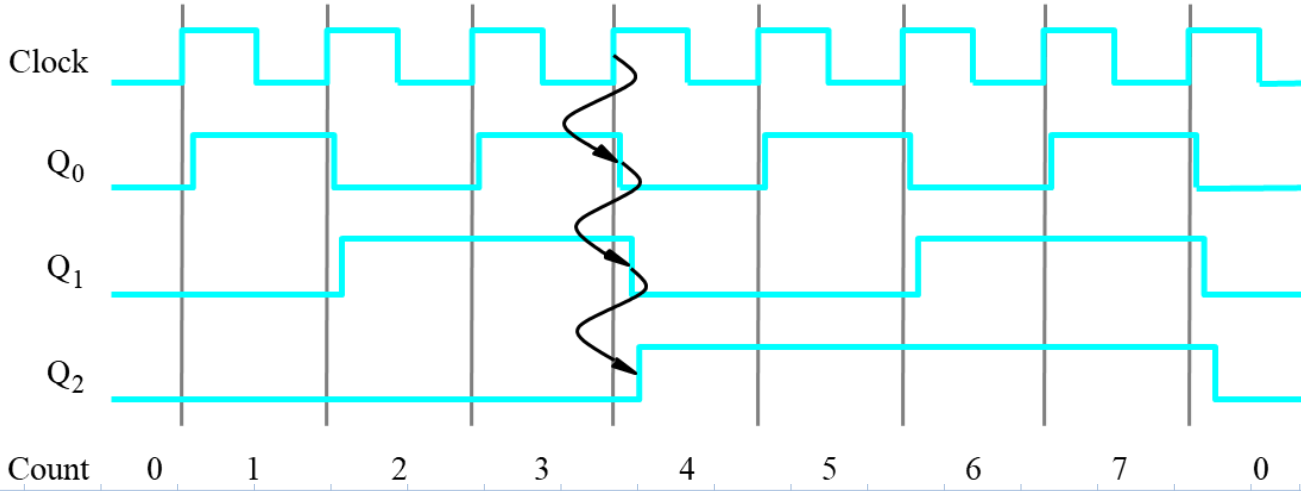
- Para contar, es necesario saber el estado de la cuenta, por lo que es necesario un circuito que tenga memoria
- Contar (incrementar en 1) tiene muchas aplicaciones:
  - Contar el número de ocurrencias de ciertos eventos
  - Generar intervalos de tiempo para controlar tareas
  - Contar el tiempo entre distintos eventos, etc.
- Su implementación es simple utilizando flip-flops tipo T y D

7.9.1 Contadores asíncronos



- Con 3 flip-flops T, podemos contar de 0 a 7 la cantidad de veces que el clock cambia de 0 a 1. Los clocks de cada flip-flop están conectados en cascada
- Como están conectados a un 1, el estado de los flip-flops cambiará para cada flanco positivo del clock
- **Q0** cambia en cada ciclo del clock, **Q1** cada 2 y **Q2** cada 4 ciclos
- Notar que hay un delay de propagación ocasionado por los flip-flops

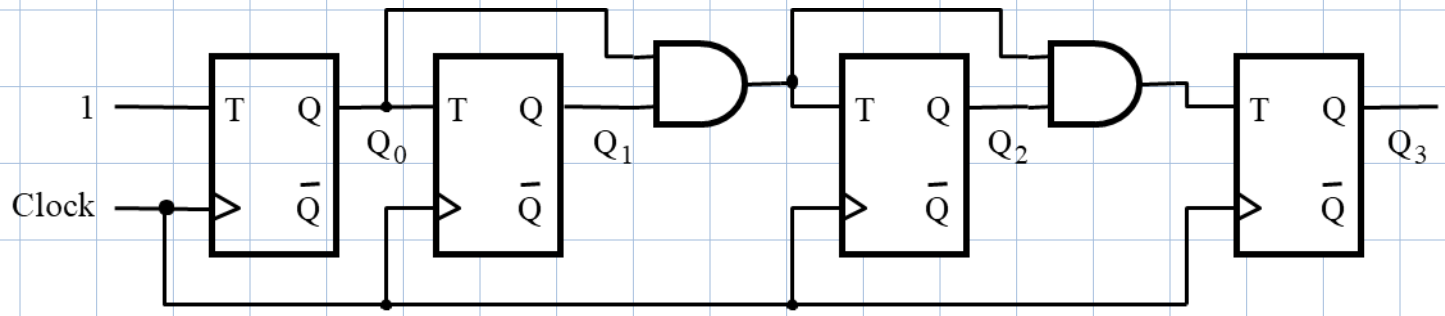




7.9.2 Contadores síncronos

INTEGRATED CIRCUITS
UNIVERSIDAD CATÓLICA

- Es mucho más práctico usar el mismo clock para todos los flip-flops (evitamos la acumulación de delays)
- Es síncrono ya que todos los cambios se producen según un único reloj
- La idea se basa en que **Q2** cambia de estado solo cuando **Q1 = Q0 = 1**, y **Q3** cambia cuando todos flip-flops anteriores valen 1



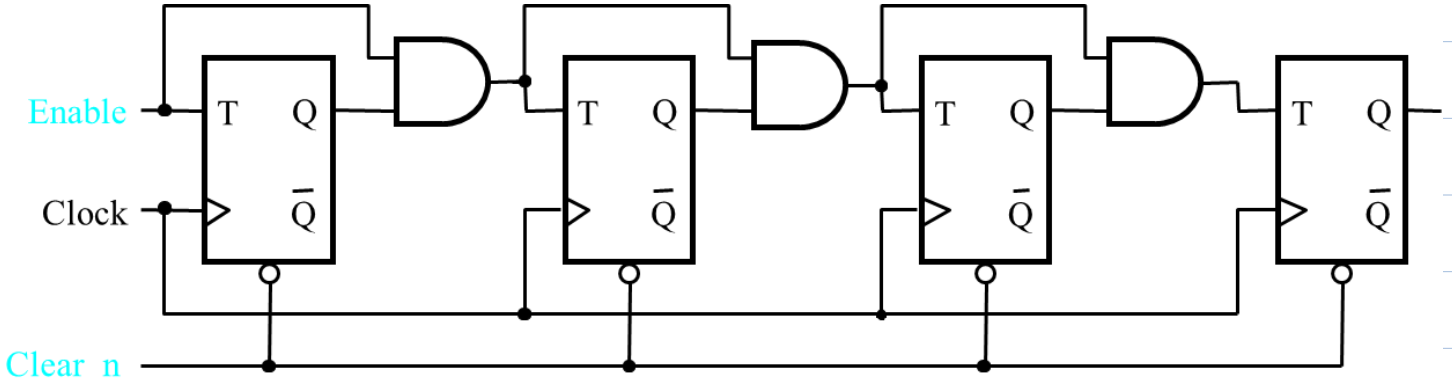
Clock cycle	Q_2	Q_1	Q_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

Q_1 changes

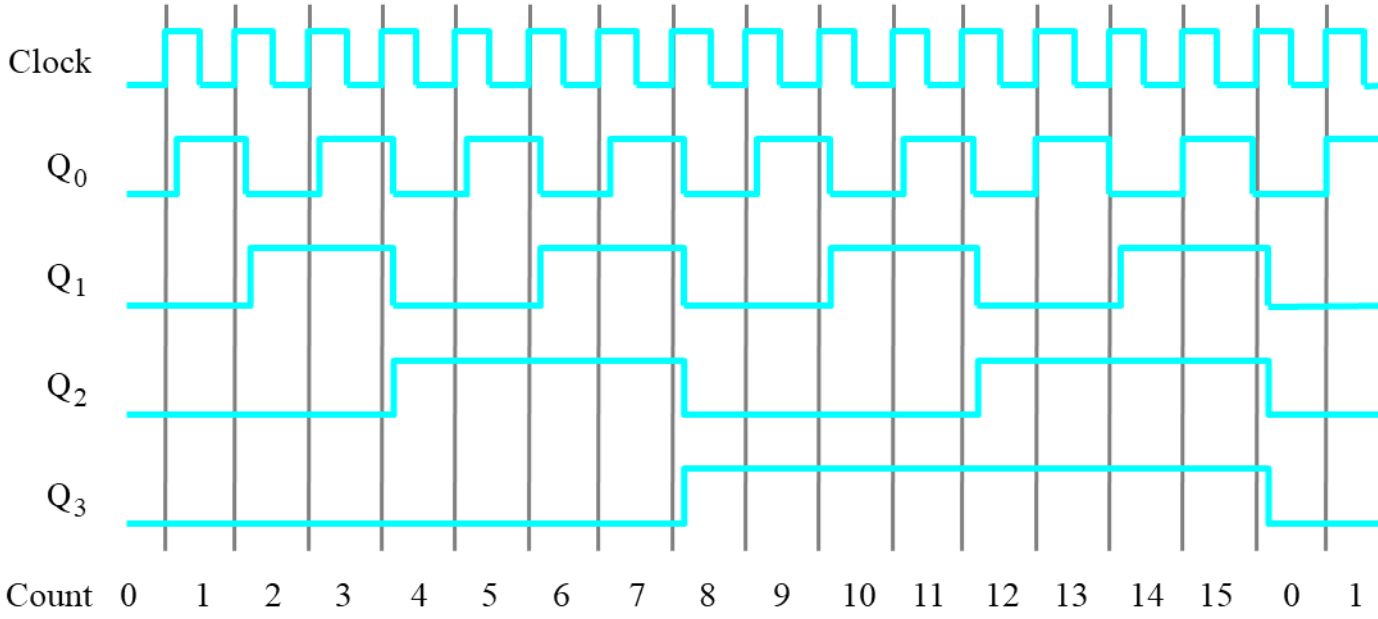
Q_2 changes

- Siendo **Tn** la entrada al flip-flop **n**:
- T0 = 1
- T1 = Q_0
- T2 = Q_1Q_0
- T3 = $Q_2Q_1Q_0$

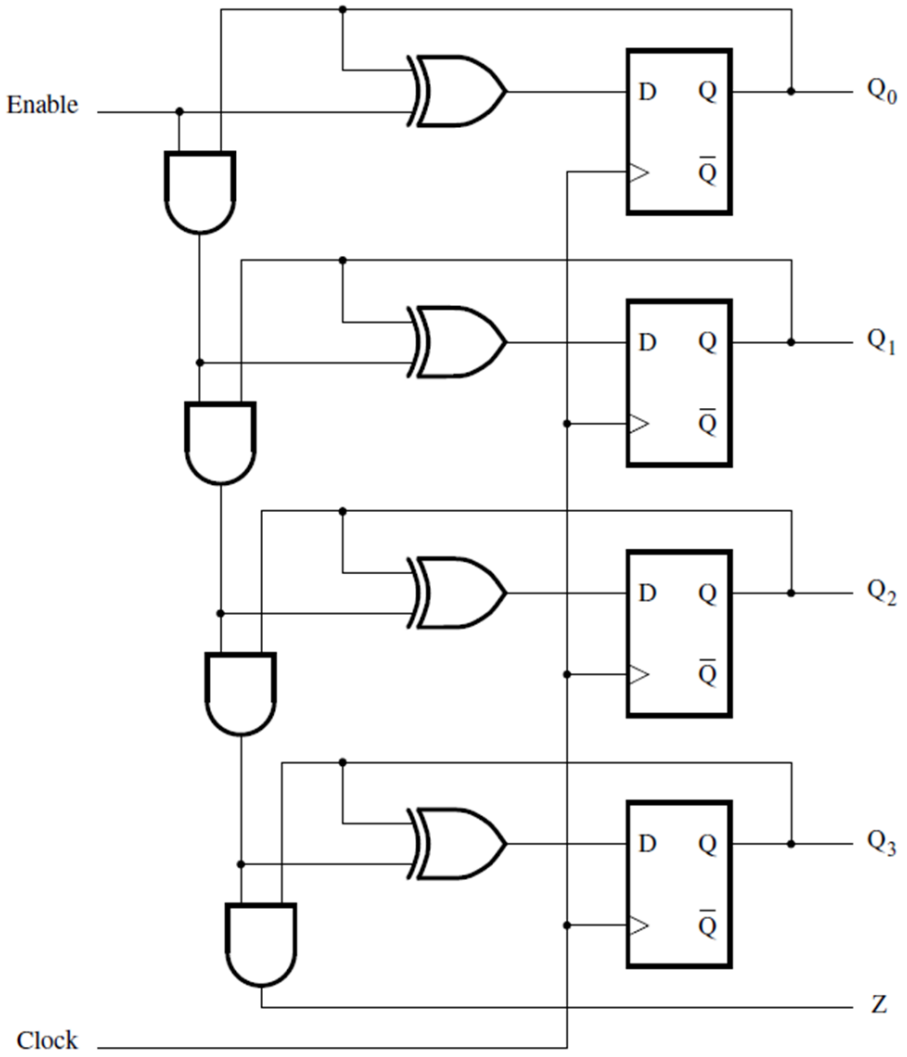
7.9.2 Contadores síncronos



- Se pueden agregar entradas que realizan funciones prácticas:
  - Enable: Pausa la cuenta (mantiene su valor)
  - Clear: Borra la cuenta (lleva todos los flip-flops a cero)

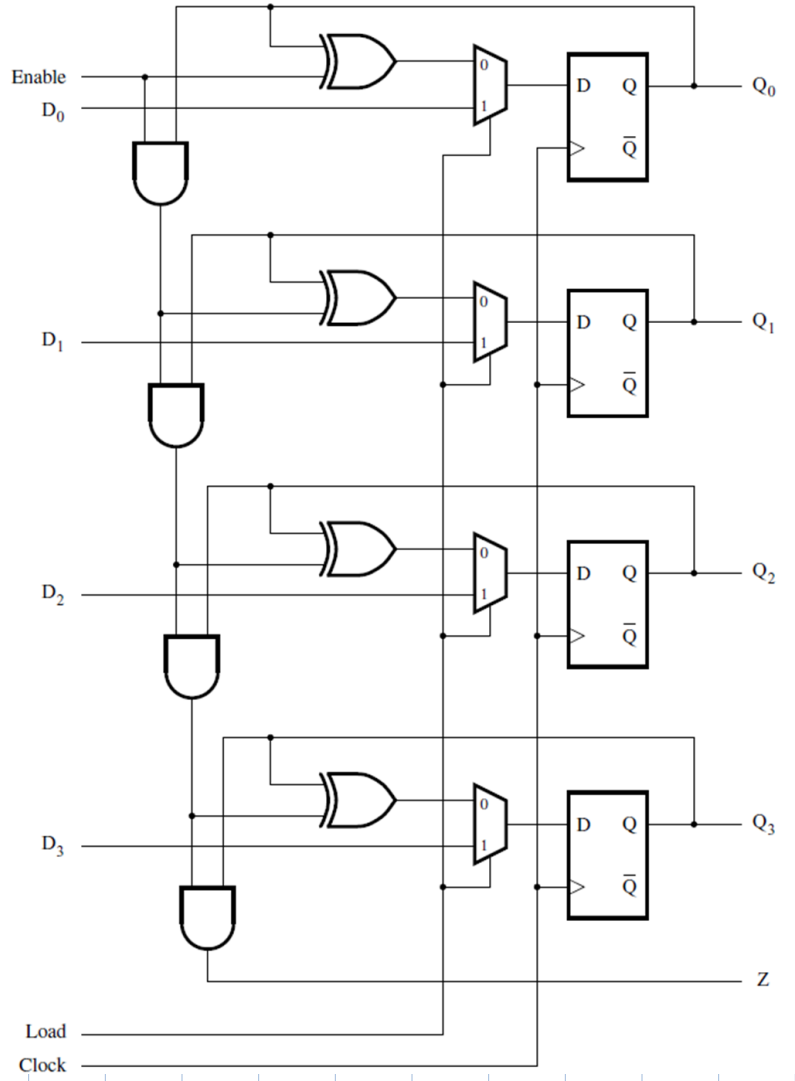


7.9.2 Contador síncrono con flip-flops tipo D de 4-bits



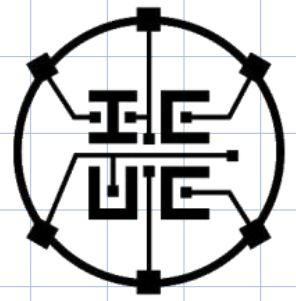
- Más adelante aprenderemos a diseñar este contador (no es tan obvio todavía)
- Por mientras, podemos notar que:
  - $D_0 = Q_0 \oplus 1 = !Q_0$
  - $D_1 = Q_1 \oplus Q_0$
  - $D_2 = Q_2 \oplus (Q_1\ Q_0)$
  - $D_3 = Q_3 \oplus (Q_2\ Q_1\ Q_0)$
- Esto funciona ya que esencialmente es lo mismo que el contador anterior
- Estamos transformando un flip-flop D en uno tipo T debido a que:
  - D = Q!T + !QT
  - Entonces D = Q ⊕ T

7.9.3 Contadores con carga paralela



- Usando multiplexores, es possible partir contando de un numero distinto de 0
- Para esto usamos multiplexores y la señal **Load**
  - Cuando **Load = 0**, el circuito cuenta normalmente
  - Cuando **Load = 1**, se carga un nuevo valor inicial $D_3D_2D_1D_0$

7.10 Sincronización del reset



- ¿Qué hacemos para contar hasta un número distinto de 2^n?
  - Una forma es reseteando el contador
- Supongamos que queremos contar hasta 5
  - La compuerta AND del circuito permite identificar cuando llegamos a 5, activa la señal Load y carga los valores $D_2D_1D_0$
  - Como se carga un cero, el contador reinicia efectivamente la cuenta en 5 y no en 7
- Como el reinicio ocurre en un flanco del clock, se dice que este contador tiene un reset **síncrono**

