Ayudantía

Almacenamiento de Datos

Resumen
Flancos
Diseño de Circuitos

Resumen

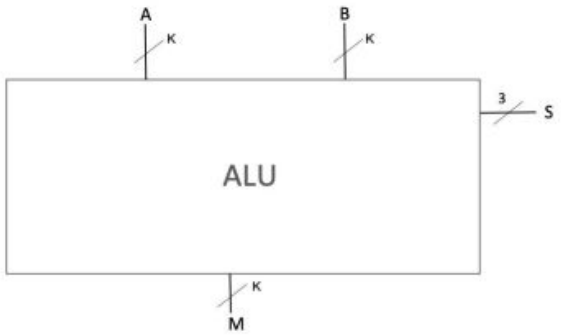
Resumen
Flancos
Diseño de Circuitos

Objetivos

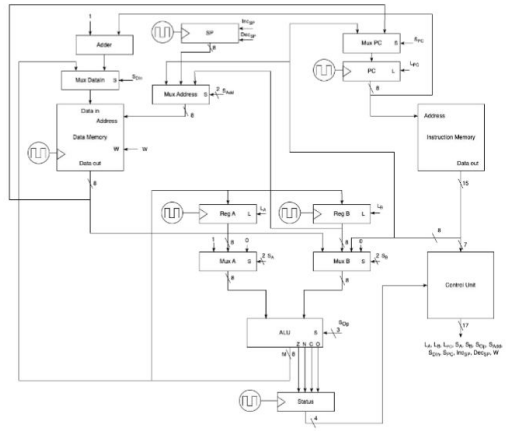
- **Comprender** por qué es necesario almacenar datos en una máquina programable.
- **Conocer** componentes digitales (latches, flip-flops) que permiten el almacenamiento.
- **Realizar** ejercicios que consoliden los conceptos aprendidos.

Almacenamiento de Datos

- Hasta ahora tenemos una ALU, pero ¿Qué pasa con los resultados de la ALU? *No se guardan :(*
- **Necesitamos almacenar información** para utilizarla y seguir operando
- ¡Hoy veremos **componentes digitales** que permiten guardar datos y comenzaremos a construir nuestro primer computador!



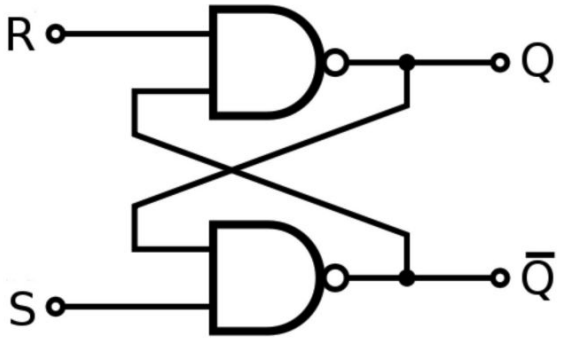
Lo que sabemos



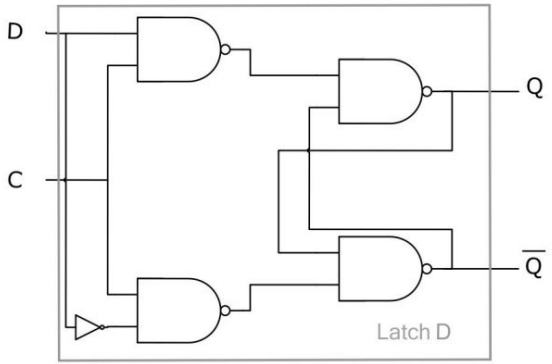
Lo que sabremos más hacia el final del curso

Latches: Retención simple de estado

- **Permiten guardar un bit en un circuito**
- ¡Su estado depende de las entradas y del estado anterior!
- Tipos principales:
  - **RS**: Reset-Set, *presenta un estado inválido*
  - **D**: Más seguro, *evita combinaciones inválidas*
- Se activan mientras la señal de control está activa



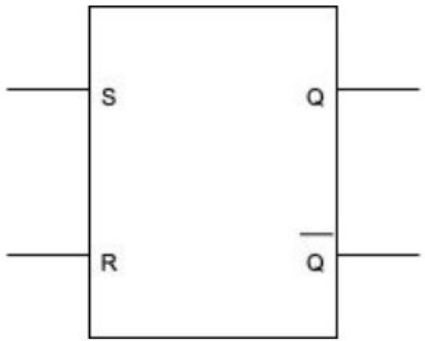
Latch RS



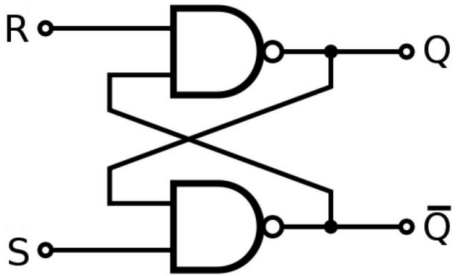
Latch D

Latch RS

Componente que realiza **R**eset (Q=0), **S**et (Q = 1) ó mantiene un estado previo: Q(t+1) = Q(t). A continuación, veremos cómo funciona



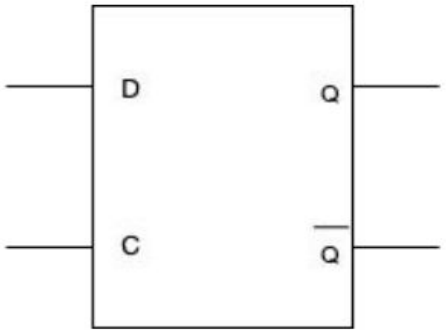
Abstracción



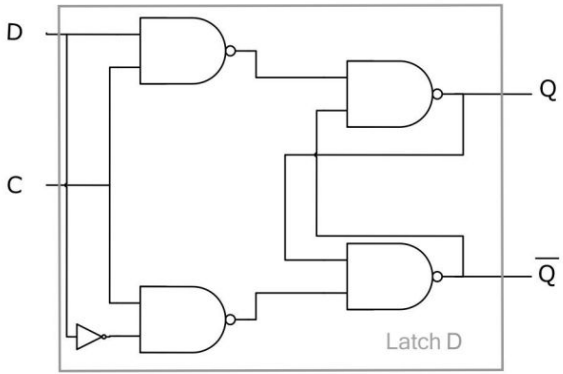
S	R	Q(t+1)
0	0	-
0	1	0
1	0	1
1	1	Q(t)

Latch D

Componente que actualiza el valor de un estado **Q** con una señal **D** si, y solo si la señal de control está activa (C = 1). D = Data.



Abstracción



C	D	Q(t+1)
0	0	Q(t)
0	1	Q(t)
1	0	0
1	1	1

Flancos

Suponemos una **señal *clock*** llamada **CLK**, la cual cambia de 1 a 0 y de 1 a 0 en una **frecuencia constante**.

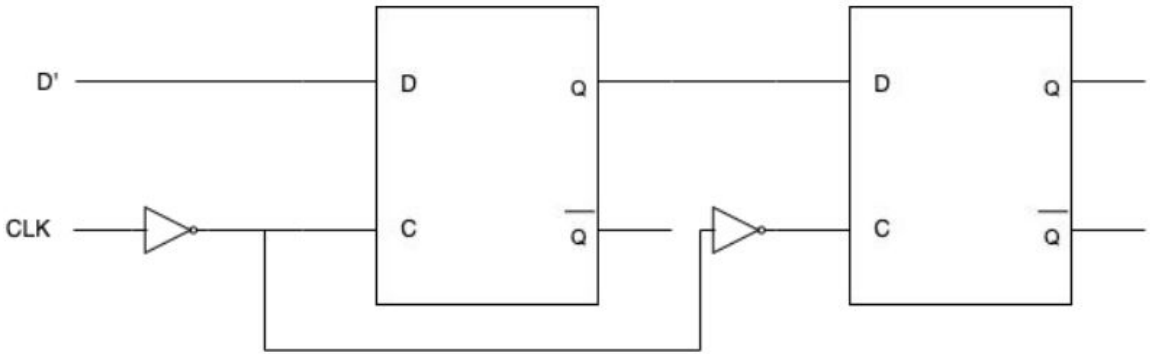
- Flanco de SUBIDA: 0 → 1
- Flanco de BAJADA: 1 → 0

Flanco de subida

Flanco de bajada

Flip-Flops

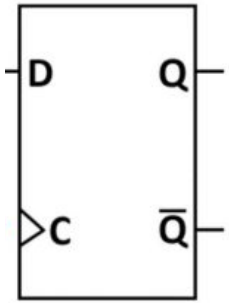
- Son construidos con latches
- **Por defecto, almacenan datos sólo en flanco de subida del clock (señal CLK), pero son modificables**
- Ideales para **sincronizar** operaciones
- Tipos:
  - D: Almacena un valor dado
  - JK y T: más flexibles, pero menos usados en este curso



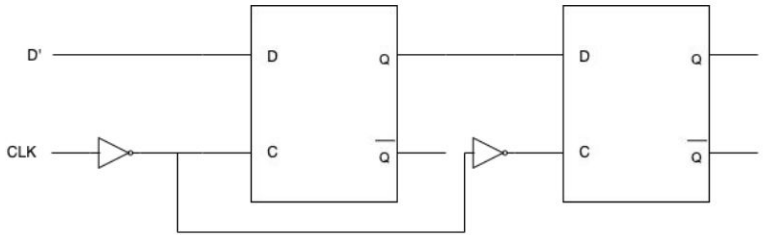
Flip-Flop D

Flip-Flop D

Componente que permite guardar el estado anterior de una señal en un instante dado. *Por defecto, la acción ocurre en flanco de subida*



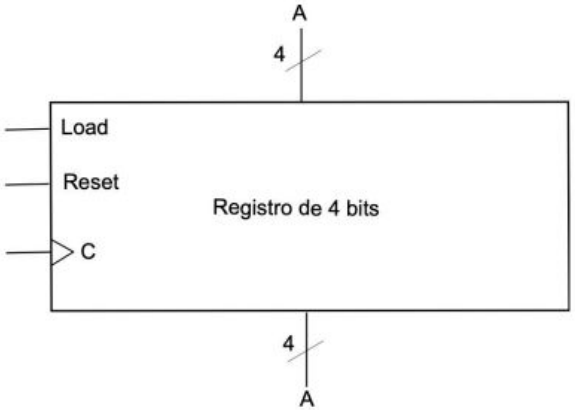
Abstracción



CLK	D	Q(t+1)
0/1/↓	0/1	Q(t)
↑	1	Q(t)
↑	0	0

Registros

- Un registro es **un conjunto de flip-flops** que guarda varios bits
- Se puede controlar con señales:
  - **Load (L)**: Autoriza carga de un nuevo dato
  - **Reset (R)**: Borra el contenido (el registro guarda un 0)



Abstracción: Registro de 4 bits

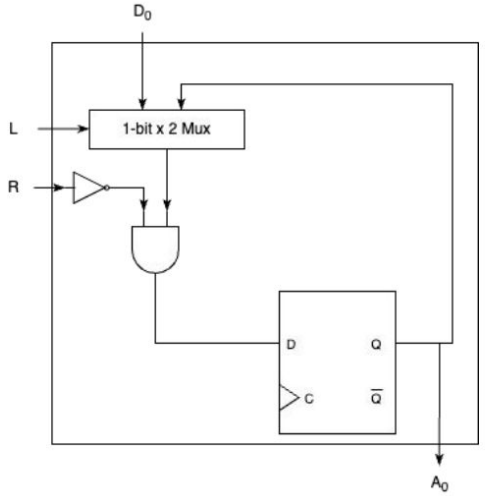
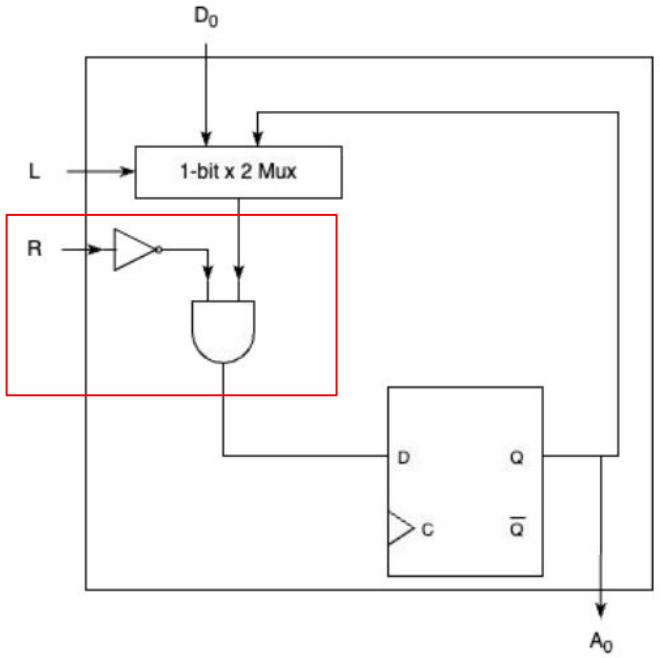


Diagrama Interno Registro A0

Registros: RESET

Esta señal actúa directamente sobre la entrada D del Flip-Flop D donde se almacena A0.

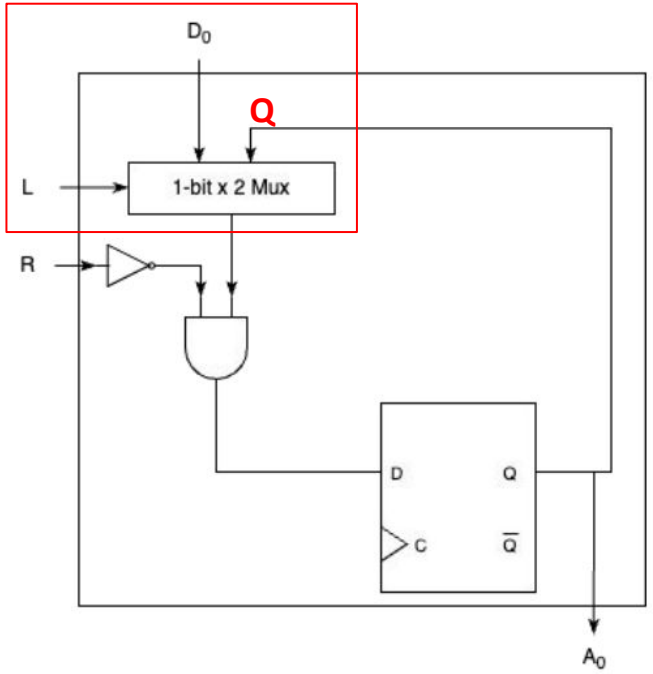
- **Casos**
  - **R = 1** : el Flip-Flop guardará un 0, dado que AND resulta en 0
  - **R = 0**, el valor almacenado A0 dependerá de la salida del Mux



Registros: LOAD

Esta señal es el **selector del Mux** que escoge entre D0 y Q(t)

- **Casos**
  - **L = 1** : Se escoge la señal de entrada D0 y se manda al AND.
  - **OJO**
    - **si R = 0**, entonces se guardará D0 en el Flip-Flop.
    - **Si R = 1**, ocurre un reset.
  - **L = 0** : Se escoge Q(t) y se almacena, por ende Q(t+1) = Q(t)



Tipos de Memoria: RAM & ROM

- **RAM** (Random Access Memory)
  - Es la memoria temporal donde se cargan los programas en ejecución
  - Se pierden sus datos al finalizar el programa

- **ROM** (Read-Only Memory)
  - Acorde a la arquitectura Harvard, la ROM almacena las instrucciones del programa a ejecutar (el código assembly en nuestro caso)

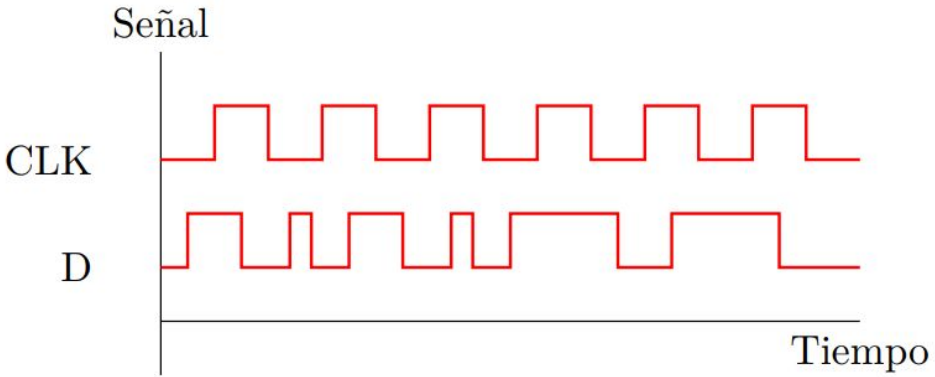
Flancos

Resumen
Flancos
Diseño de Circuitos

Flancos: ¿Para qué sirven?

El desafío de este ejercicio es **entender la relación entre Latches, Flancos y Flip-Flops**. Para esto, se tiene una entrada CLK y una señal D de un latch D.
Se asumen que todas las salidas comienzan en 0 (nada ocurrió antes)

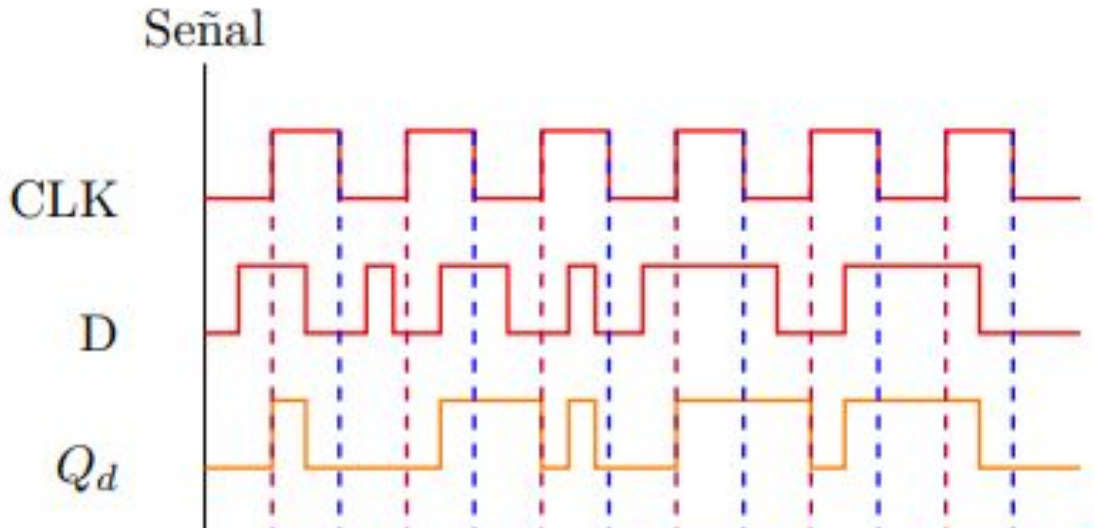
- **Objetivo**: Dibujar las señales de salida de:
  - Latch D
  - Flip-Flop D en flanco de subida
  - Flip-Flop D en flanco de bajada



Flancos: ¿Para qué sirven?

Objetivo: Dibujar las señales de salida de:

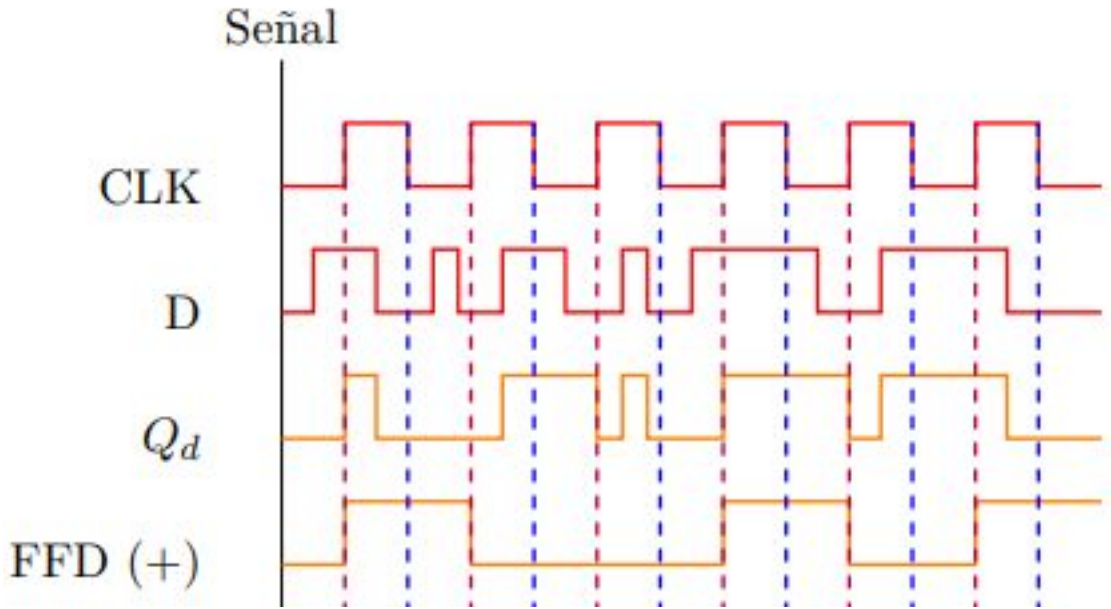
- **Latch D**
- Flip-Flop D en flanco de subida
- Flip-Flop D en flanco de bajada



Flancos: ¿Para qué sirven?

Objetivo: Dibujar las señales de salida de:

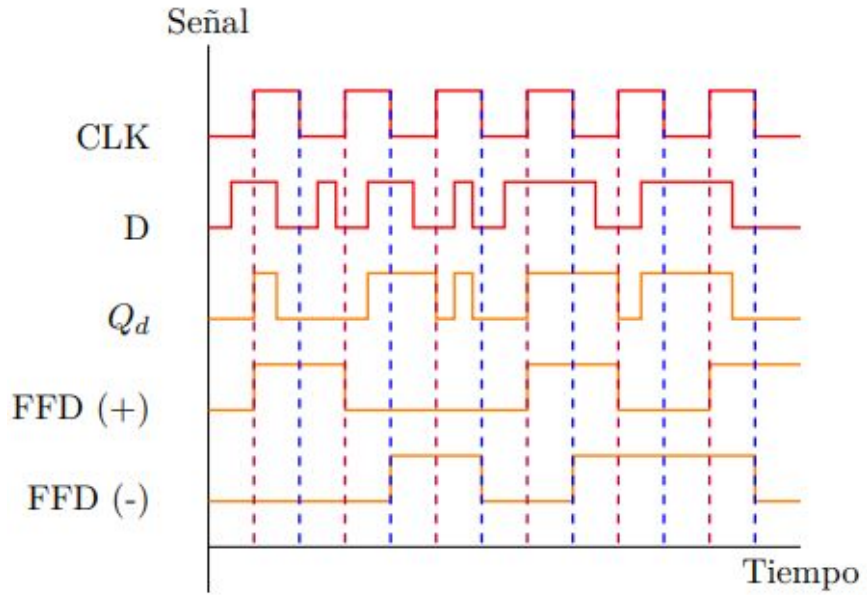
- Latch D
- **Flip-Flop D en flanco de subida**
- Flip-Flop D en flanco de bajada

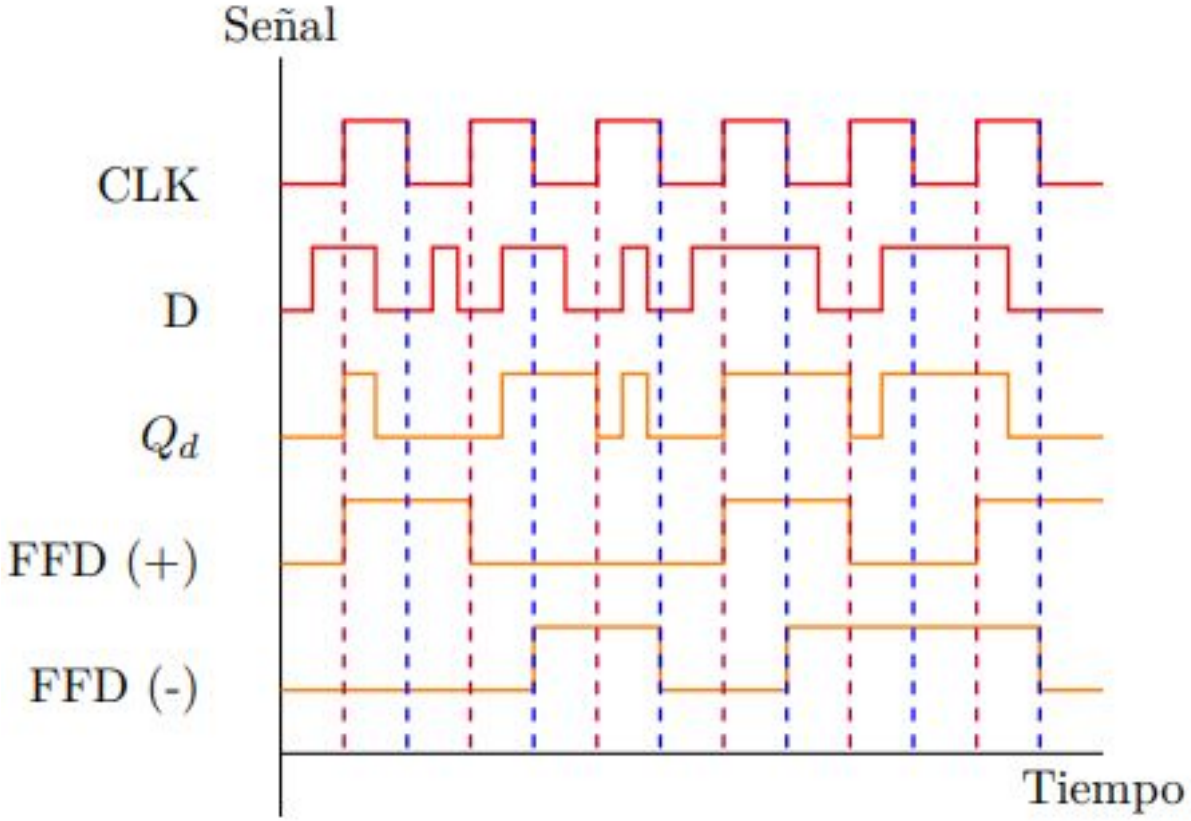


Flancos: ¿Para qué sirven?

Objetivo: Dibujar las señales de salida de:

- Latch D
- Flip-Flop D en flanco de subida
- **Flip-Flop D en flanco de bajada**




Señal
CLK
D
Q_d
FFD (+)
FFD (-)
Tiempo

Diseño de Circuitos

Resumen
Flancos
Diseño de Circuitos

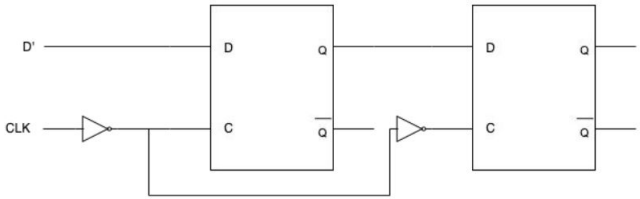
Diseño: a)

Diseñe, utilizando todos los elementos de circuitos lógicos vistos en clase, un **contador secuencial de 2 bits** que se incrementa con cada *flanco de bajada* de la señal de control. Este contador deberá partir en 00 y volver a 00 por *overflow* una vez llega a su valor máximo.

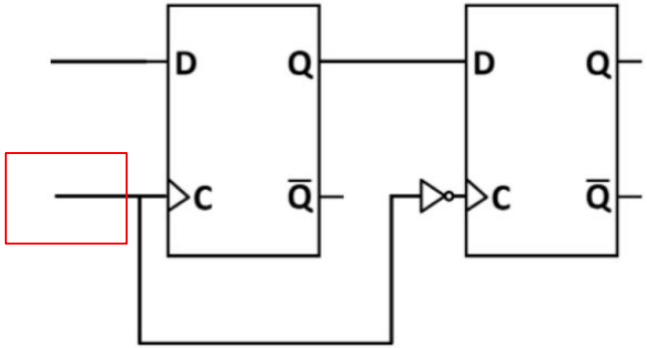
Diseño: a) SOLUCIÓN

Diseñe, utilizando todos los elementos de circuitos lógicos vistos en clase, un **contador secuencial de 2 bits** que se incrementa con cada *flanco de bajada* de la señal de control. Este contador deberá partir en 00 y volver a 00 por overflow una vez llega a su valor máximo.

Modificación: En la modificación el Latch de entrada ya no tiene la entrada de control C negada



Flip-Flop D

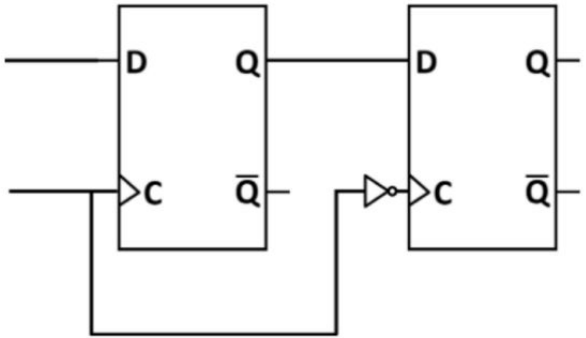


Flip-Flop D modificado

Diseño: a) SOLUCIÓN

Diseñe, utilizando todos los elementos de circuitos lógicos vistos en clase, un **contador secuencial de 2 bits** que se incrementa con cada *flanco de bajada* de la señal de control. Este contador deberá partir en 00 y volver a 00 por overflow una vez llega a su valor máximo.

Consejo: Crear una tabla para representar cada estado posible



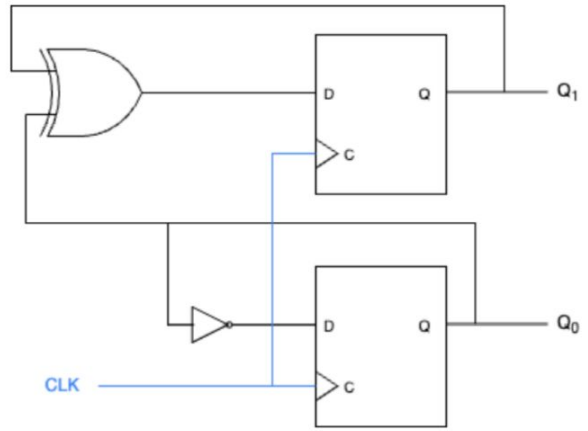
Q^t_1	Q^t_0	Q^{t+1}_1	Q^{t+1}_0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

Diseño: a) SOLUCIÓN

Diseñe, utilizando todos los elementos de circuitos lógicos vistos en clase, un **contador secuencial de 2 bits** que se incrementa con cada *flanco de bajada* de la señal de control. Este contador deberá partir en 00 y volver a 00 por overflow una vez llega a su valor máximo.

Diseño del circuito: Se deduce que $Q^{t+1}_0 = \neg Q^t_0$ y $Q^{t+1}_1 = Q^t_0$ XOR Q^t_1

Q^t_1	Q^t_0	Q^{t+1}_1	Q^{t+1}_0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0



Diseño: b)

Modifique el circuito anterior para que pase a ser un contador secuencial de 3 bits.

Diseño: b) SOLUCIÓN

Modifique el circuito anterior para que pase a ser un contador secuencial de 3 bits.

Respuesta: Se debe agregar un tercer Flip-Flop D (modificado) que representa el bit más significativo. Se debe hacer otra tabla de verdad para representar todos los estados posibles $2^3 = 8$.

Q^t_2	Q^t_1	Q^t_0	Q^{t+1}_2	Q^{t+1}_1	Q^{t+1}_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

Diseño: b) SOLUCIÓN

Modifique el circuito anterior para que pase a ser un contador secuencial de 3 bits.

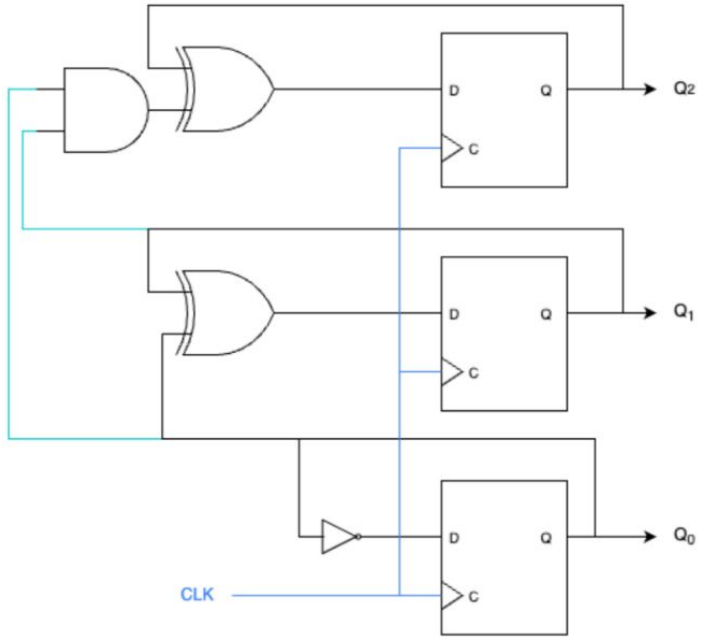
Respuesta: Se debe agregar un tercer Flip-Flop D (modificado) que representa el bit más significativo. Se debe hacer otra tabla de verdad para representar todos los estados posibles $2^3 = 8$.

Diseño: Es importante notar que Q_2 alterna
su valor si y sólo si $Q_1 = Q_2 = 1$. Por ende,
$Q^{t+1}_2 = (Q^t_1 \text{ AND } Q^t_0) \text{ XOR } Q^t_2$

Q^t_2	Q^t_1	Q^t_0	Q^{t+1}_2	Q^{t+1}_1	Q^{t+1}_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

Diseño: b) SOLUCIÓN

Diseño: Es importante notar que Q_2 alterna su valor si y sólo si $Q_1 = Q_2 = 1$. Por ende, $Q^{t+1}_2 = (Q^t_1 \text{ AND } Q^t_0) \text{ XOR } Q^t_2$



Q^t_2	Q^t_1	Q^t_0	Q^{t+1}_2	Q^{t+1}_1	Q^{t+1}_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

Diseño: c)

Este contador debe recibir una señal de entrada R de 1 bit. Si el valor de R es igual a 1 **durante el flanco de bajada** de la señal de control, entonces el contador debe actualizar su valor a 0 en vez de incrementarse en una unidad.

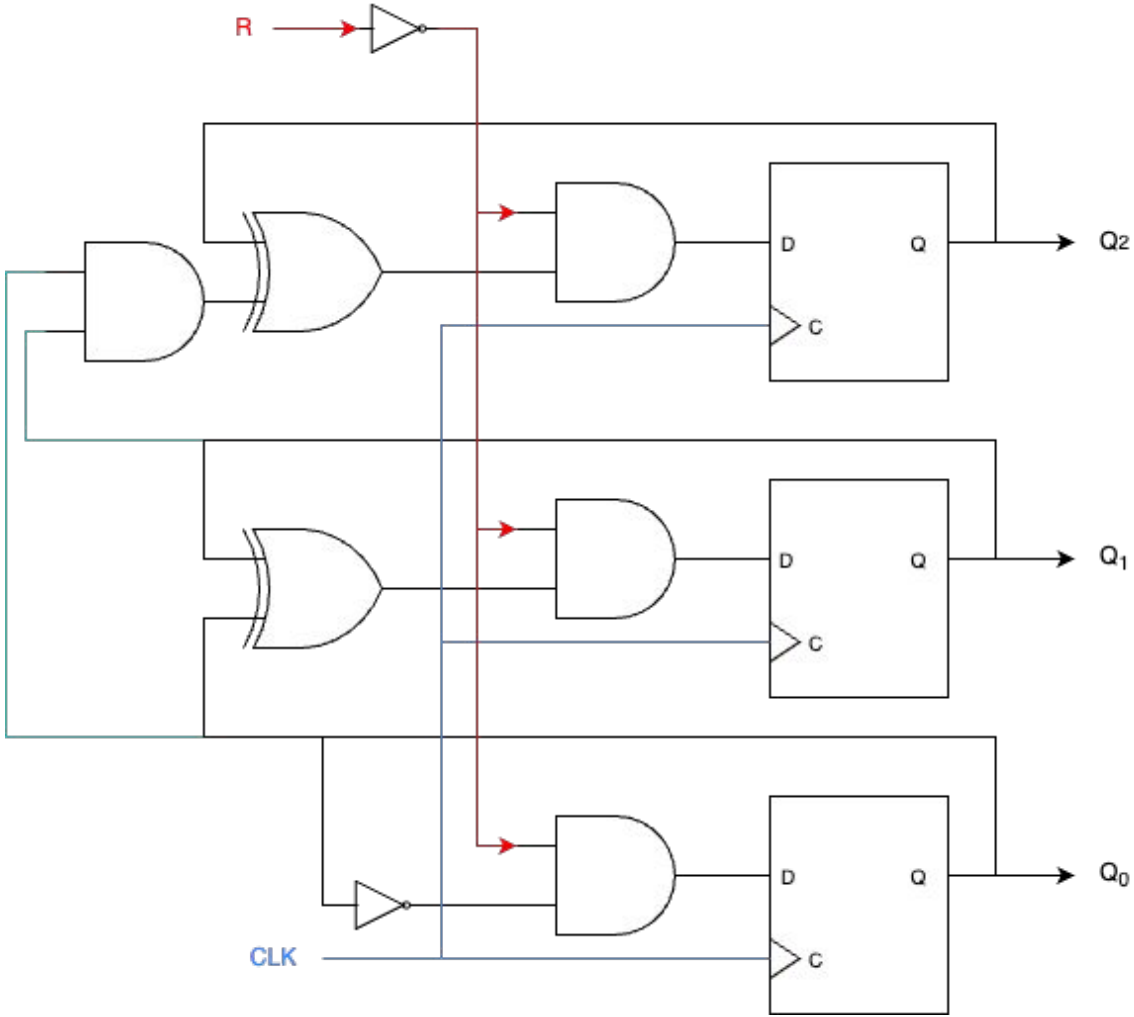
Diseño: c) SOLUCIÓN

Este contador debe recibir una señal de entrada R de 1 bit. Si el valor de R es igual a 1 **durante el flanco de bajada** de la señal de control, entonces el contador debe actualizar su valor a 0 en vez de incrementarse en una unidad.

Respuesta: Para implementar la señal de reset R hay que considerar que si R= 1, entonces $Q^{t+1}_2\, Q^{t+1}_1\, Q^{t+1}_0 = 000$. Por lo tanto, haciendo uso de la ley de dominación, se pueden ajustar las expresiones de los estados del contador de la siguiente forma:

$$Q^{t+1}_0 = R^t \wedge Q^t_0$$
$$Q^{t+1}_1 = R^t \wedge (Q^t_0 \oplus Q^t_1)$$
$$Q^{t+1}_2 = R^t \wedge ((Q^t_1 \wedge Q^t_0) \oplus Q^t_2).$$

Finalmente, el siguiente diagrama representa el contador modificado conectado a la señal R:


R
D
Q
C
Q_2
D
Q
C
Q_1
D
Q
C
Q_0
CLK

Feedback

Resumen
Flancos
Diseño de Circuitos

Ayudantía

Almacenamiento de Datos

Resumen
Flancos
Diseño de Circuitos