Tecnologia y Organizacion de Computadores

Asignatura: 10GIIN

Actividad: #4

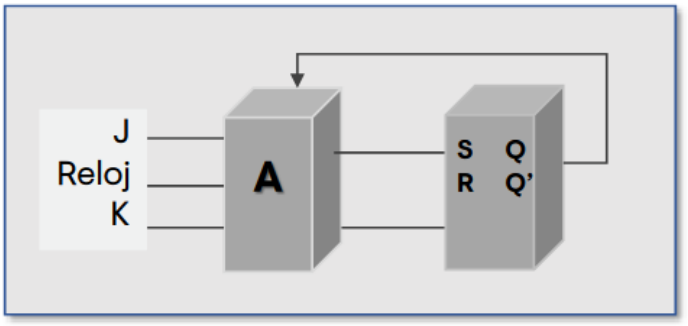
Alumno: Gagliardo Miguel Angel

Fecha: 01/05/2022

Objetivo de la actividad

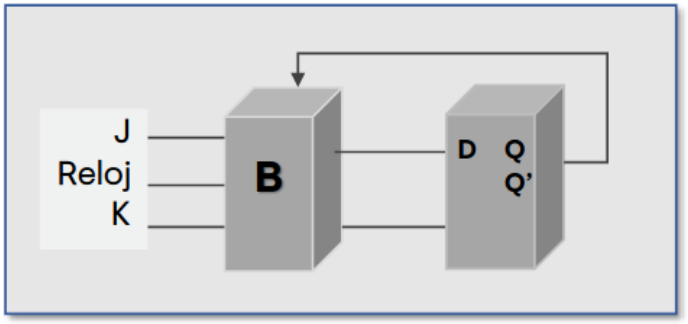
1. Implementar un *flip-flop* tipo JK en base a uno tipo SR.

Lo que se busca es la lógica combinatoria del **bloque A**. Se calculan las expresiones de S y R a partir de la tabla de verdad y mapas.



2. Implementar un *flip-flop* tipo JK a partir de uno tipo D.

Lo que se busca es la lógica combinatoria del **bloque B**. Se calculan las expresiones de D a partir de la tabla de verdad y mapas.



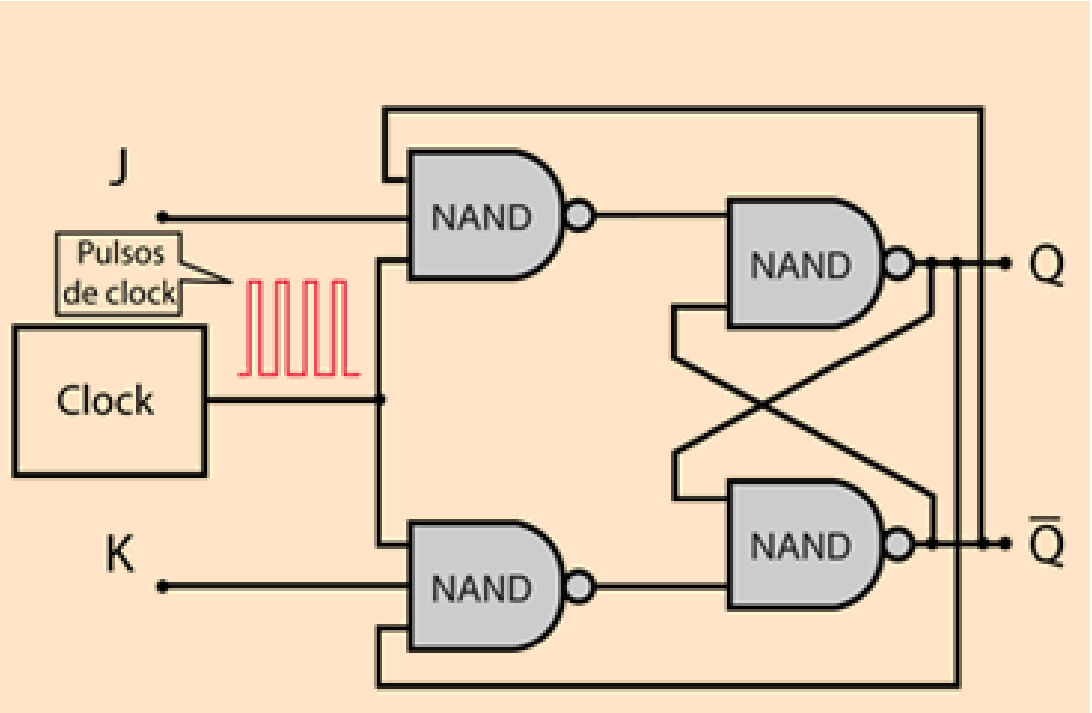
Cada una de las decisiones de diseño (producto de esta actividad) deben estar justificadas indicando: uso de postulados, propiedades, leyes, tablas de verdad, mapas de Karnaugh, compuertas, etc.

Descripción del Flip-Flop JK

El Flip-Flop J-K tiene el carácter de seguimiento de entrada del flip-flop D sincronizado, pero tiene dos entradas, denominadas tradicionalmente J y K. Si J y K son diferentes, la salida Q toma el valor de J durante la subida del siguiente pulso de sincronismo.

Si J y K son ambos low (bajo), entonces no se produce cambio alguno. Si J y K son ambos high (alto), entonces en la siguiente subida de clock la salida cambiará de estado. Puede realizar las funciones del flip-flop set/reset y tiene la ventaja de que **no hay estados ambiguos**.

En la siguiente figura, podemos ver una versión equivalente de este Flip-Flop:



La tabla de transición muestra las características de un flip-flop J-K disparado por flanco ascendente.

Inputs			Outputs		
J	**K**	**CLK**	**Q**	**Q'**	
0	0	↑	Q_0	Q_0'	Sin cambios
1	0	↑	1	0	Reset
0	1	↑	0	1	Set
1	1	↑	Q_0	Q_0'	Cambios

Implementación de un Flip-Flop JK en base a uno RS

Para esta implementación es necesario obtener un Flip-Flop JK en base a un SR, con lo cual lo primero que necesitamos es visualizar la Tabla de excitación del Flip-Flop SR a partir de sus estados actuales (Q_t) y sus estados futuros (Q_{n+1}):

Q_t	Q_{t+1}	S	R
0	0	0	X
0	0	0	X
0	1	1	0
0	1	1	0
1	1	X	0
1	0	0	1
1	1	X	0
1	0	0	1

Luego, tomamos la tabla de verdad del Flip-Flop JK, obtenemos los valores para la conversión de JK a SR a partir de su tabla de estados actuales (Q_t) y sus estados futuros (Q_{n+1}), siendo las salidas S y R:

Q_t	J	K	Q_{t+1}	S	R
0	0	0	0	0	X
0	0	1	0	0	X
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	1	X	0
1	0	1	0	0	1
1	1	0	1	X	0
1	1	1	0	0	1

Con el resultado de esta tabla, vamos a armar los mapas de Karnaugh correspondientes

Para S:

Qn \ J K	00	01	11	10
0	0	0	1	1
1	X	0	0	X

Luego: S = $\neg Q$. J

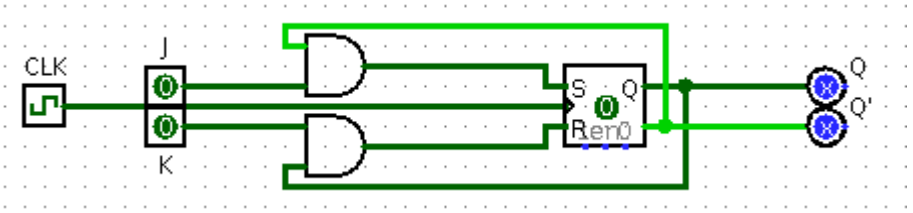
Para R:

Qn \ J K	00	01	11	10
0	X	X	0	0
1	0	1	1	0

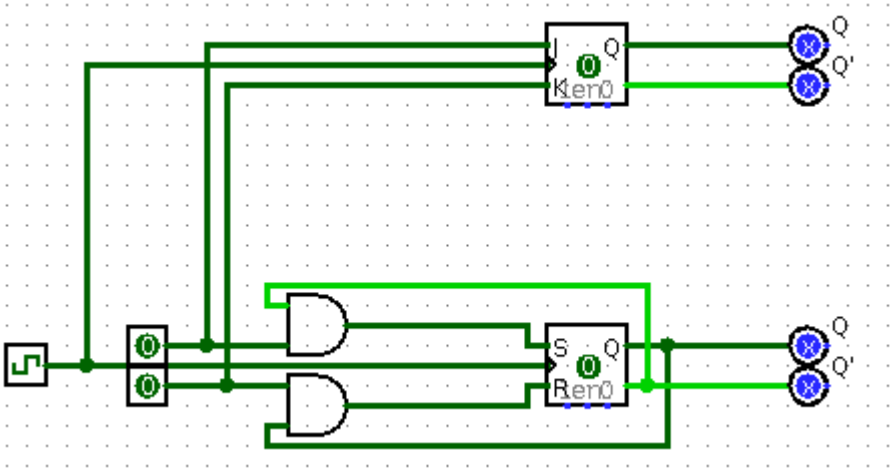
Luego: R = Q_n . K

Diseño Obtenido

Con el resultado de los mapas de Karnaugh armamos el equivalente al flip-flop JK utilizando compuertas AND, junto a nuestro Flip-Flop SR.



Si tuviéramos que conectarlo al lado de un Flip-Flop JK para ver cómo funcionan ambos de la misma manera y sus conexiones, se vería de la siguiente manera:



Implementación de un Flip-Flop tipo JK en base a uno D

Para esta obtener un Flip-Flop JK en base a uno D lo primero es visualizar la Tabla de excitación del Flip-Flop D a partir de sus estados actuales (Q_t) y sus estados futuros (Q_{n+1}):

Q_t	Q_{t+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

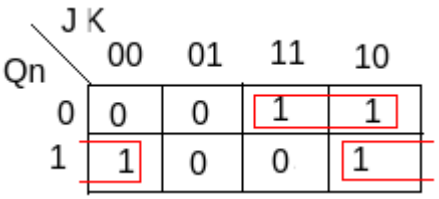
Ya vemos una complejidad de antemano aquí, al no tener 2 “outputs” como antes (SR), tendremos que reemplazar esa funcionalidad.

Luego, tomamos la tabla de verdad del Flip-Flop JK, obtenemos los valores para la conversión de JK a D a partir de su tabla de estados actuales (Q_t) y sus estados futuros (Q_{n+1}), siendo D la única salida:

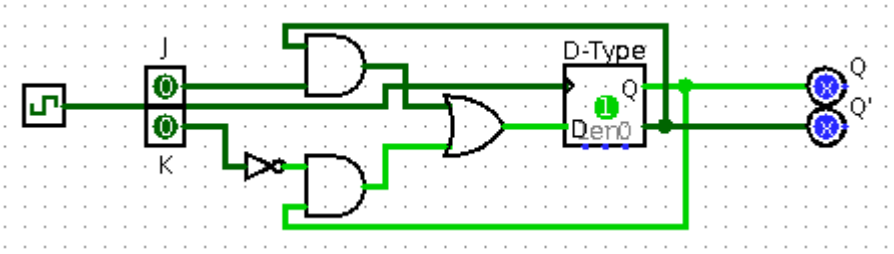
Q_t	J	K	Q_{t+1}	D
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	1	1
1	0	0	1	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	1

Con el resultado de esta tabla, vamos a armar el mapa de Karnaugh correspondiente para **D**:

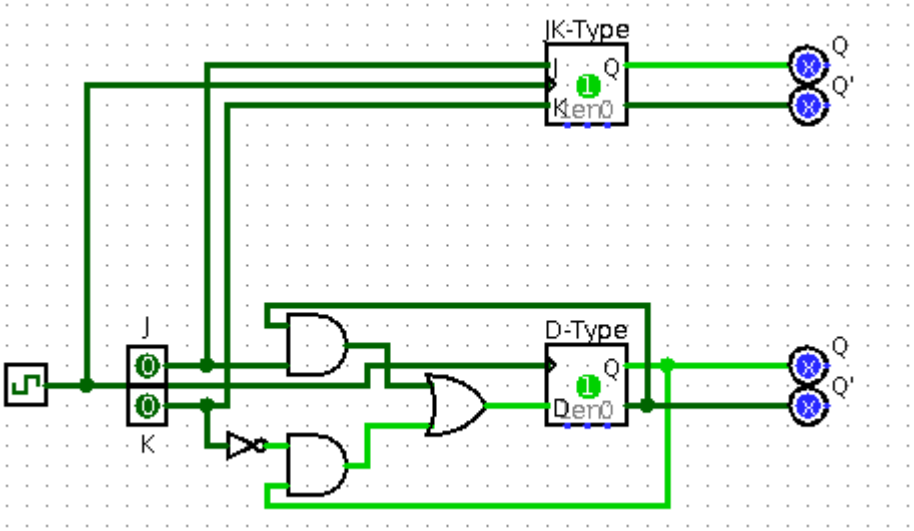
Luego: D = $\neg K.Q \ + J.\neg Q$



Con el resultado de este mapa, armamos el circuito equivalente, utilizando compuertas AND, OR y NOT



Si tuviéramos que conectarlo al lado de un Flip-Flop JK para ver cómo funcionan ambos de la misma manera y sus conexiones, se vería de la siguiente manera:



Conclusión

Por un lado, podemos ver que el flip-flop J-K se comporta como el flip-flop S-R, a excepción de que el JK **resuelve el problema** de tener una salida indeterminada cuando las entradas se encuentran activas a la vez. Podríamos decir entonces que el flip-flop J-K es idéntico al S-R, pero sin entradas restringidas.

Por el lado del flip-flop D, como podemos ver que inicialmente su tabla de verdad es más sencilla, pero al mismo tiempo no es tan flexible como el JK. Esto es principalmente porque el JK tiene las funciones de SET, RESET, HOLD (mantener restado) y TOGGLE (estado contrario). Por tanto el flip-flop JK puede hacer más funciones al tener 2 inputs (“J” y “K”), en vez de la única entrada (“D”) del D-Type, que sólo habilita las funciones de SET y RESET.

Esto quiere decir en última instancia que, tal como se ve en los circuitos, en términos de componentes la implementación del flip flop D es más compleja y costosa al tener que agregar más compuertas para compensar la falta de entradas (“J” y “K”), en cambio, el flip-flop SR al ser muy similar al JK, su implementación es mucho más sencilla y menos costosa, donde lo único que necesitamos es evitar las entradas restringidas.