Organización y Arquitectura de las Computadoras

Práctica 05: Lógica Secuencial

Bonilla Reyes Dafne
319089660

Medina Guzmán Sergio
314332428

1. Objetivo

El alumno aprenderá a analizar, diseñar y simular circuitos secuenciales.

2. Procedimiento

Deberás entregar un solo archivo de Logisim con las soluciones de los ejercicios y un documento con las respuestas a las preguntas planteadas. Solamente puedes hacer uso de compuertas lógicas AND, OR y NOT; flip-flops tipo D, JK y SR; multiplexores, demultiplexores; separadores; y pines de entrada y salida. Recuerda etiquetar las entradas y salidas de cada uno de los subcircuitos.

3. Ejercicios

1. Diseña un circuito secuencial en Logisim que simule el autómata descrito arriba, donde tendrás que minimizar las tablas para cada Source y Reset usando mapas de Karnaugh o álgebra booleana, recuerda que los resultados de los 2 estados de memoria deben ir conectados al cable 0 y 1, de un separador de 4 bits conectado a un LED Hexadecimal (Hex Display, en la parte de Input/Output de Logisim), de tal forma que despliegue la succión de números 0,1,2,3 o la sucesión de letras C, D, E, F en el LED. Para guardar el estado, deberás usar flip-flops tipo SR.

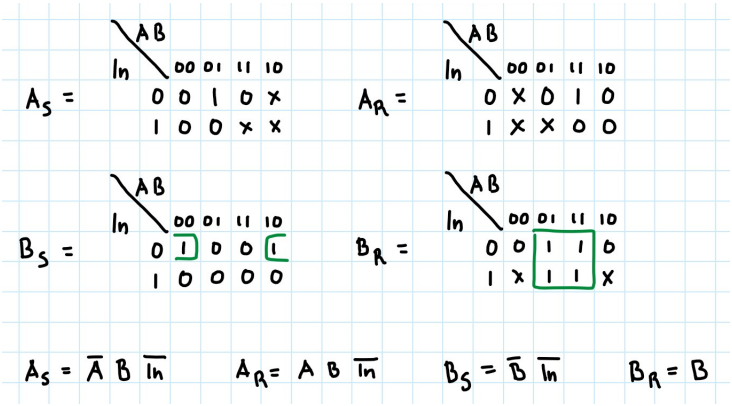


Figura 1: Mapas de Karnaugh del ejercicio y fórmulas a representar en el circuito

   Este circuito se encuentra en `Circuitos/Ejercicio-01`.

2. Diseña un circuito para simular un registro, solo puedes usar flip-flops tipo D.

   Este circuito se encuentra en `Circuitos/Ejercicio-02`. Fue diseñado usando separadores de 8 bits para poder realizar de manera más eficiente el ejercicio 3.

3. Diseña un circuito para simular el banco de registros descrito en el planteamiento. Utiliza el registro diseñado en el ejercicio anterior.

   Este circuito se encuentra en `Circuitos/Ejercicio-03`. Fue diseñado usando los registros del ejercicio 2 y puertas de tipo AND. Su función es activar la entrada `Enable` del registro usando el demultiplexor y las puertas AND. El resto de entradas del registro se mantienen iguales, usando un `Clock` y un `Pin 0`. Si el `Enable` recibe 1, se escribe el registro y finalmente los 4 registros obtenidos se convierten en los datos de entrada para el multiplexor que terminará por leer los datos.

4. Interconecta el sumador/restador de 8 bits elaborado en la práctica anterior con el banco de registros del ejercicio anterior y de acuerdo a las especificaciones de la sección de banco de datos.
   Este circuito se encuentra en `Circuitos/Ejercicio-04`.

4. Preguntas

1. ¿En qué difieren los distintos tipos de flip-flops?

   Un circuito que tiene dos estados estables se trata como un flip-flop. Estos estados estables se utilizan para almacenar datos binarios que se pueden cambiar aplicando distintas entradas.[1] Los flip-flops de tipo D, JK y SR son tres tipos comunes de flip-flops cuya principal diferencia es su funcionamiento:[2]

   1. **Flip-flop D:** El flip-flop tipo D, también conocido como flip-flop de datos, tiene una entrada D que se utiliza para introducir datos. Cuando se aplica un pulso a la entrada de reloj, el valor presente en la entrada D se copia en la salida del flip-flop. El flip-flop tipo D se utiliza comúnmente como una memoria de una sola unidad de bits, ya que el valor que se introduce en la entrada D se mantiene en la salida hasta que se introduce un nuevo valor.[3]
   2. **Flip-flop JK:** El flip-flop tipo JK tiene dos entradas principales, llamadas J y K. Cuando se aplica un pulso a la entrada de reloj, el valor presente en las entradas J y K se utiliza para determinar el valor de salida del flip-flop. Si J y K son ambos 0, el valor de salida no cambia. Si J y K son ambos 1, el valor de salida cambia de estado. Si J es 1 y K es 0, el valor de salida se establece en 1. Si J es 0 y K es 1, el valor de salida se establece en 0. El flip-flop tipo JK se utiliza comúnmente para implementar contadores y otros circuitos secuenciales.[4]
   3. **Flip-flop SR:** El flip-flop tipo SR tiene dos entradas principales, llamadas S y R. Cuando se aplica un pulso a la entrada de reloj, el valor presente en las entradas S y R se utiliza para determinar el valor de salida del flip-flop. Si S y R son ambos 0, el valor de salida no cambia. Si S y R son ambos 1, el valor de salida es indeterminado. Si S es 1 y R es 0, el valor de salida se establece en 1. Si S es 0 y R es 1, el valor de salida se establece en 0. El flip-flop tipo SR se utiliza comúnmente como un elemento de memoria y como una herramienta para la eliminación de rebotes en circuitos digitales.[5]

      Notemos que el flip-flop JK funciona de la misma manera que el flip-flop SR. La única diferencia entre el flip flop JK y el flip flop SR es que cuando ambas entradas del flip flop SR se establecen en 1, el circuito produce estados no válidos como salidas, pero en el caso del flip flop JK, no hay estados no válidos incluso si ambos flip-flops J y K están configuradas en 1.

   De esta manera, podemos concluir que las principales diferencias entre los flip-flops de tipo D, JK y SR se encuentran en la forma en que se controlan y en su comportamiento cuando se aplican diferentes combinaciones de señales de entrada.

2. ¿Cómo se decide que tipo se usará en el circuito?

   La elección del tipo de flip-flop a utilizar en un circuito dependerá de los requisitos específicos de ese circuito. En general, se deben considerar los siguientes factores al decidir qué tipo de flip-flop utilizar:[6]

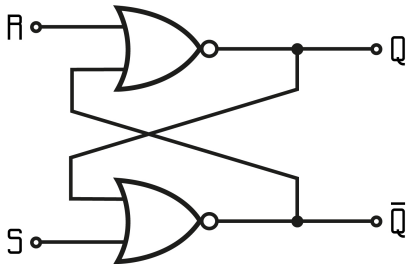
   1. **Requisitos de temporización:** La elección del tipo de flip-flop puede depender de los requisitos de temporización del circuito. Por ejemplo, si se necesita una alta velocidad de operación, se puede

utilizar un flip-flop tipo D, mientras que si se necesita un circuito secuencial complejo, un flip-flop tipo JK puede ser más adecuado.

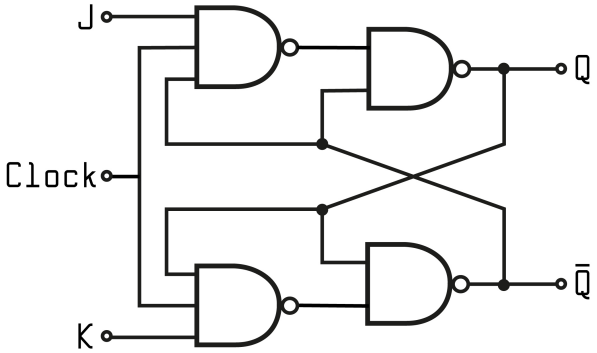
2. **Necesidades de memoria:** Si se necesita un circuito de memoria, el flip-flop tipo D es una buena opción, ya que puede almacenar un valor de entrada y retenerlo hasta que se introduce un nuevo valor. Si se necesita un circuito de memoria que pueda cambiar de estado dependiendo de otras entradas, entonces se puede usar un flip-flop tipo JK o SR.
3. **Compatibilidad del circuito:** La elección del tipo de flip-flop también puede depender de la compatibilidad con otros componentes del circuito. Por ejemplo, si se están utilizando otros componentes que requieren entradas de reloj separadas, entonces se puede utilizar un flip-flop tipo JK o SR que tenga entradas de reloj separadas para cada flip-flop.
4. **Complejidad del circuito:** Si el circuito es simple y solo se necesita un elemento de memoria, un flip-flop tipo D puede ser suficiente. Si el circuito es más complejo y requiere múltiples elementos de memoria interconectados, entonces se puede utilizar un flip-flop tipo JK o SR.

3. ¿Cómo se ve un flip-flop SR usando solo compuertas lógicas básicas? ¿Y un flip-flop JK?

   - Flip-flop SR:[7]



   - Flip-flop JK:[8]



4. Existen 2 tipos de máquinas de estado finito que usan flip-flops. ¿Cuáles son? ¿En qué se diferencian?

   Los 2 tipos de máquinas de estado finito que usan flip-flops son las máquinas de estado finito síncronas y asíncronas.

   1. **Síncronas.** Las máquinas de estado finito síncronas son aquellas que utilizan una señal de reloj para sincronizar la transición de estados y actualizar los flip-flops. Estas máquinas se basan en la detección del flanco de subida o bajada del reloj para realizar la transición de estado.[9]
   2. **Asíncronas.** Las máquinas de estado finito asíncronas no utilizan una señal de reloj para sincronizar la transición de estados. En su lugar, se basan en la detección de las condiciones de entrada para realizar la transición de estado. Estas máquinas utilizan flip-flops asíncronos o latches para almacenar información sobre su estado actual.[10]

5. ¿Por qué es mejor utilizar flip-flops de tipo JK en vez de tipo SR para realizar máquinas de estado finito?

   Existen varias razones para preferir flip-flops de tipo JK. En primer lugar, el flip-flop JK puede ser utilizado para implementar cualquier tipo de flip-flop, como SR, D o T, y, por lo tanto, se puede usar para implementar cualquier tipo de máquina de estado finito.

   En segundo lugar, el flip-flop JK es menos propenso a errores de estado indeterminado en comparación con el flip-flop SR. Esto se debe a que el flip-flop JK tiene una entrada de reloj, lo que evita la posibilidad de que ambas entradas S y R sean iguales a 1 al mismo tiempo, lo que puede resultar en una situación de estado indeterminado. Por otro lado, en el flip-flop SR, si ambas entradas S y R son iguales a 1 al mismo tiempo, puede resultar en una situación de estado indeterminado.

   Además, el flip-flop JK permite la implementación de operaciones de `Set` y `Reset` separadas. En comparación, el flip-flop SR no permite la implementación de operaciones de `Set` y `Reset` separadas, ya que cuando ambas entradas S y R son iguales a 1, el resultado no es determinado.

   En general, el flip-flop JK es una opción más flexible y segura para la implementación de máquinas de estado finito, ya que evita los errores de estado indeterminado y permite la implementación de operaciones de `Set` y `Reset` separadas.[11]

6. Un registro de desplazamiento es un circuito secuencial que desplaza a la izquierda o a la derecha la información contenida en él. Considerando el desplazamiento de 1 bit a la izquierda, ¿cómo se implementa dicho circuito? ¿Cómo podríamos simular su funcionamiento con las operaciones que se tienen en la ALU de 8 bits?

   Para implementar un registro de desplazamiento de 1 bit a la izquierda, podemos utilizar una cadena de flip-flops conectados en cascada, de tal manera que la salida del flip-flop de la derecha se conecte a la entrada del flip-flop de la izquierda. [12]

   Para el desplazamiento de 1 bit a la izquierda, se aplica un pulso de reloj, lo que hace que cada bit se desplace a la izquierda y el bit más significativo (el que estaba en el primer flip-flop) se pierda. El nuevo bit que entra al registro es un cero.

   Para simular el funcionamiento del registro de desplazamiento de 1 bit a la izquierda con las operaciones que se tienen en la ALU de 8 bits, podemos realizar la siguiente secuencia de operaciones:

   1. Cargar el valor del registro de desplazamiento en un registro de propósito general de 8 bits en la ALU.
   2. Realizar una operación de desplazamiento a la izquierda en el registro de propósito general de 8 bits en la ALU.
   3. Almacenar el resultado de la operación de desplazamiento de vuelta en el registro de desplazamiento.

   Cada operación de desplazamiento a la izquierda en la ALU de 8 bits equivale a un desplazamiento de 1 bit a la izquierda. De esta manera, podemos simular el funcionamiento del registro de desplazamiento utilizando las operaciones de la ALU de 8 bits.

Referencias

[1] Basics of Flip-Flops

[2] Logisim Documentation: D/T/J-K/S-R Flip-Flop

[3] D Flip Flop

[4] JK Flip Flop

[5] SR Flip Flop

[6] Mano, M. (4a Ed.) *Logic And Computer Design Fundamentals*, Pearson, 2013.

[7] Galaviz, C. J. (n.d) *Lógica digital y diseño de circuitos digitales* p.32

[8] JK flip-flop using NAND gate

[9] Synchronous State Machines

[10] Asynchronous State Machines

[11] Hennessy, J. y D. Patterson (6a Ed.) *Computer Architecture: A Quantitative Approach*, 2017.

[12] Registros de Desplazamiento