Diseño de circuitos combinacionales

Estudiantes:

David Andrés Torres Betancour (CC 1017251689)

Juliana Cadavid Ramírez (CC 1000415078)

Arquitectura de computadores

Universidad de Antioquia

Ingeniería de sistemas

2020

Descripción:
El circuito combinacional desarrollado se basa en una red de ordenamiento de ocho números de ocho bits cada uno que tiene como salida los números ordenados de manera ascendente, en la cual se usa un comparador de dos números de 8 bits con signo, si el número es negativo se representa en complemento a 2, y este a su vez se construye con un comparador de dos números de un bit. En el diseño solo se implementaron operadores lógicos básicos como: AND, OR y NOT. El desarrollo se realizó en la herramienta Logisim Evolution.

Objetivos:

- Reconocer y aplicar los conceptos vistos en clase como representación de complemento a 2, tablas de verdad, simplificación de funciones booleanas, mapas de Karnaugh, esquemas de circuitos combinacionales y su aplicación en Logisim Evolution.
- Desarrollar circuitos combinacionales de manera estructurada y jerárquica.
- Realizar una red de ordenamiento con comparadores básicos elaborados en base a conceptos de tablas de verdad, simplificación de funciones booleanas con mapas de Karnaugh.

Desarrollo

1. **Elaboración del comparador de un bit**

   Para la elaboración del comparador de un bit se aplicaron tablas de verdad con 4 entradas, una para el primer número a comparar (A), otra para el otro número (B) y los otros dos son considerados como acarreos que se deben tener en cuenta para la comparación de los dos números de un bit, el cual será conveniente para la ejecución del comparador de dos números de 8 bits. Estos dos acarreos son E y N. Las combinaciones elegidas en el desarrollo para la comparación son 00 para reconocer la igualdad entre los números, 01 para A<B y 10 para A>B. En la tabla hay comparaciones de don´t care en las salidas ya que no se tuvo en cuenta la combinación 11. Además en situaciones de A=0 : B= 1 y A=1 : B=0, las entradas de E y N no importan ya que no influyen en la salida.Las demás salidas son resultados de comparaciones entre 0 y 1, teniendo en cuenta la metodología mencionada anteriormente. (Figura 1, Tabla de verdad).

   Luego de la elaboración de la tabla de verdad, es necesario reconocer los mintérminos que se adquirieron en la salida de S e I con los 1 de la columna respectiva que según la tabla serían:

   S(A,B,E,N)= A’B’EN’ + AB’+ ABEN’
   I(A,B,E,N) = A’B’E’N + A’B + ABE’N

   Para obtener una función más reducida se pasa al paso número dos.

A	B	E	N	S	I
0	0	0	0	0	0
0	0	0	0	0	1
0	0	1	0	1	0
0	0	1	1	X	X
0	1	X	X	0	1
0	1	X	X	0	1
0	1	X	X	0	1
0	1	X	X	0	1
1	0	X	X	1	0
1	0	X	X	1	0
1	0	X	X	1	0
1	0	X	X	1	0
1	1	0	0	0	0
1	1	0	1	0	1
1	1	1	0	1	0
1	1	1	1	X	X

(Figura 1, tabla de verdad)

2. Simplificación de funciones booleanas

Para un diseño más preciso y sencillo en el comparador de un bit es necesario reducir las funciones booleanas obtenidas de la tabla de verdad de S e I, para esto se implementan los mapas de Karnaugh en cada una de las funciones, con la notación de Grey. Además se tendrá en cuenta las salidas de don´t care obtenidas para una simplificación mejor. (Figura 2.1, mapa de Karnaugh de S) (Figura 2.2, mapa de Karnaugh de I)

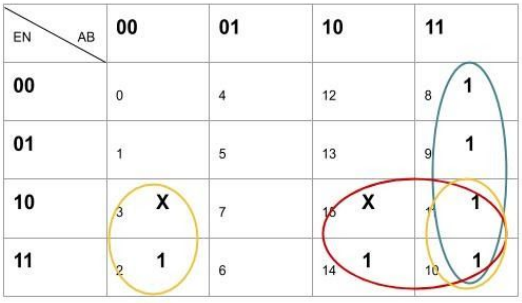


Figura 2.1, mapa de Karnaugh de S

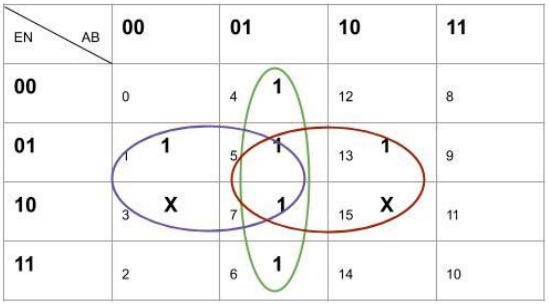


Figura 2.2, mapa de Karnaugh de I

Luego de agrupar cada uno de los subconjuntos de 4 variables obtenidos, se pasa a reducir las funciones de S y de I. Para S obtuvimos 3 subconjuntos, el primero de color azul de la última columna con los números 8,9,10 y 11 representados como 1000, 1001, 1010 y 1011 respectivamente, en el que las variables A y B no cambian, por lo que se agregan a la expresión reducida para obtener AB', el siguiente subconjunto de color rojo contiene los números 10, 11, 14 y 15 que equivalen a 1010, 1011, 1110 y 1111 respectivamente, en el que las variables A y E no cambian y se obtiene una expresión final de AE y el último grupo amarillo que tiene los números 2, 3, 10 y 11 que representan 0010, 0011, 1010 y 1011 respectivamente, en donde la variable B y E no cambian, finalmente se obtiene la expresion B'E. Cada uno de los subconjuntos contienen un implicante primo esencial, lo que implica que la función de S se expresa como una suma de mintérminos así: S(A,B,E,N)= B'E + AB' + AE, el resultado es una reducción de manera eficaz y correcta de la función principal.
Para I se obtuvieron 3 subconjuntos, el primero de color verde de la columna con los números 4,5,6 y 7 representados como 0100, 0101, 0110 y 0111 respectivamente, en el que las variables A y B no cambian, por lo que se agregan a la expresión reducida para obtener A'B, el siguiente subconjunto de color rojo contiene los números 5,7,13 y 15 que equivalen a 0101, 0111, 1101 y 1111 respectivamente, en el que las variables B y N no cambian y se obtiene una expresión final de BN y el último grupo morado que tiene los números 1, 3, 5 y 7 que representan 0001, 0011, 0101 y 0111 respectivamente, en donde la variable A y N no cambian, finalmente se obtiene la expresion A'N. Cada grupo obtenido contiene un implicante primo esencial entonces la función de I se reduce a I(A,B,E,N) = A'B + A'N + BN.

El siguiente paso sería aplicar las funciones obtenidas en la herramienta Logisim Evolution con sus compuertas de AND, OR y NOT requeridas para finalmente obtener el comparador de dos números de un bit.

3. **Circuito del comparador de dos números de un bit**

Con las funciones obtenidas de S e I, se construye un circuito combinacional de 4 entradas (A,B,E y N), en donde se obtuvieron 3 mintérminos. En el caso de S, se usan 3 compuertas AND, una compuerta OR y dos NOT, y lo mismo para el caso de I. Estas simplificaciones S(A,B,E,N)= B'E + AB' + AE y I(A,B,E,N) = A'B + A'N + BN se representan en Logisim Evolution, como en la figura 3.

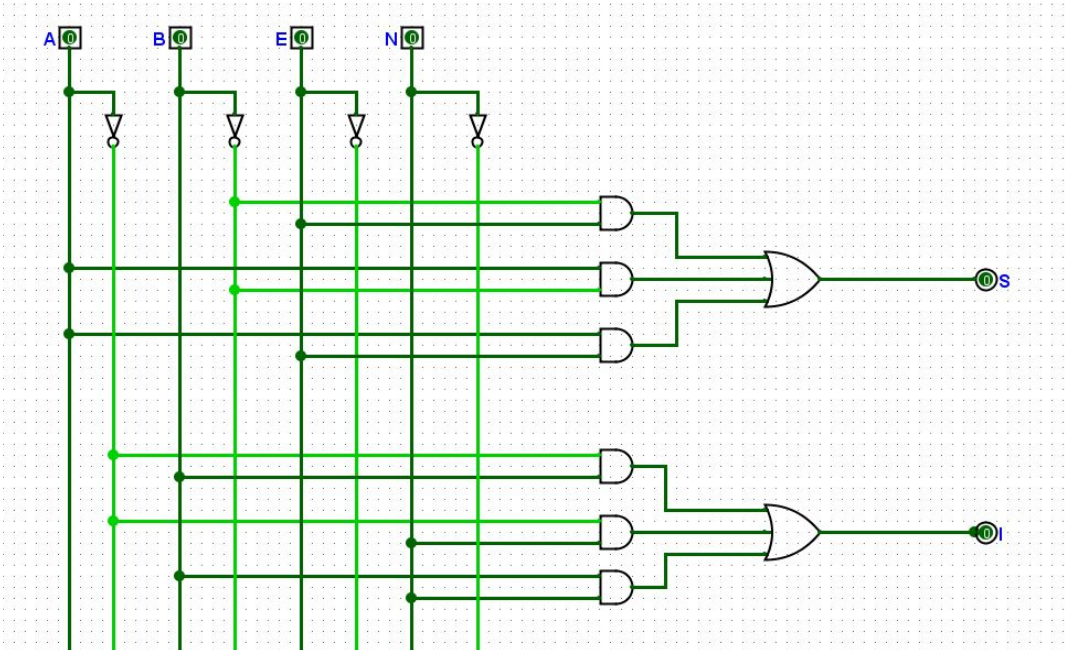


Figura 3, comparador de un bit

Luego de tener el comparador de un bit, se procede a crear el comparador de ocho bits con el comparador obtenido anteriormente.

4. **Circuito comparador de dos números de ocho bits**

Para la elaboración de este comparador se usaron 8 comparadores de un bit con un diseño jerárquico que permite la obtención del número mayor de la comparación de dos números de ocho bits con signo. Para el manejo del signo negativo en complemento a 2, fue necesario el intercambio de las entradas del último comparador de un bit, en donde la entrada de A fue B y en la de B fue A, en este caso el octavo bit del número, esto para comprobar la comparación de los números. Figura 4.1, comparador de ocho bits y Figura 4.2, continuación del comparador.

Figura 4 , comparador de ocho bits.

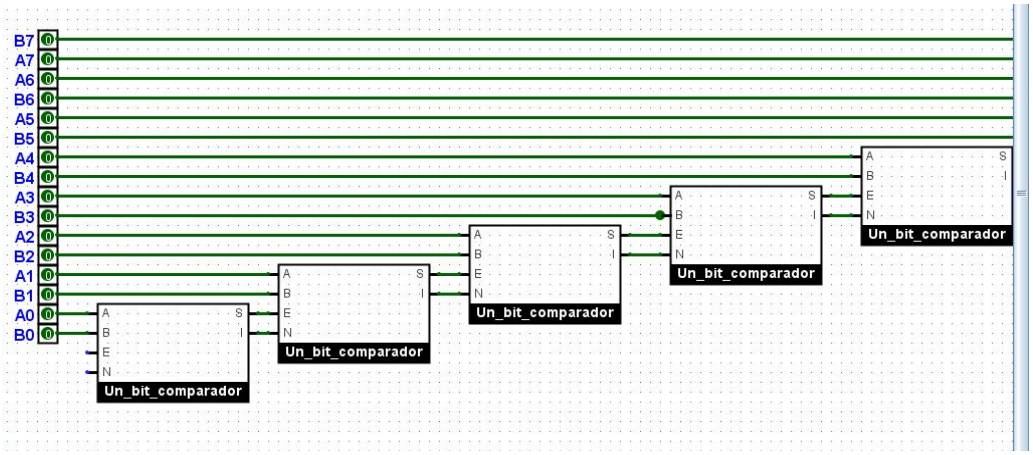
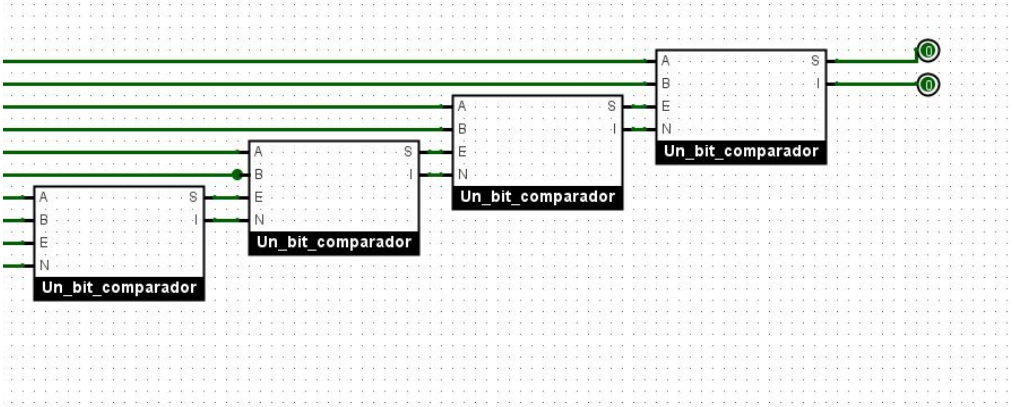


Figura 4, continuación del comparador.



5. Diseño multiplexor:

Como ya se tiene un circuito que compara dos números, lo que prosigue es buscar la forma de que dada la salida del comparador de ocho bits, indicar en el resultado el valor de los números que se están comparando. Esto se puede lograr con un multiplexor 4: 1, que tendrá dos entradas de selección cada una de 1 bit y cada canal será de 8 bits, ya que se busca que a la salida tenga esa longitud. Para el diseño del circuito es importante plantear primero la correspondiente tabla de verdad.

5.1 Tabla de verdad

Cuando se tiene en el Selector S0 y S1 los valores de (0,0) respectivamente el canal D0 podrá pasar a la salida, en este caso los números van a ser iguales así que por ese canal puede pasar cualquier de los dos números. Para el caso de (0,1) pasará el canal D1, donde se encuentra el número menor. Para los valores de (1,0) pasará el canal D2 , donde se va a encontrar el valor del número mayor y finalmente para (1, 1) se presenta una situación don't care ya que en el código que se definió no tiene significado alguno, todo lo anterior se evidencia en la Figura 5.1.

Select		Data Lines				Y
S0	S1	D0	D1	D2	D3	
0	0	D0	X	X	X	1
0	1	X	D1	X	X	1
1	0	X	X	D2	X	1
1	1	X	X	X	X	X

Figura 5.1, Tabla de verdad multiplexor

Luego de la elaboración de la tabla de verdad, es necesario reconocer los mintérminos que se adquirieron en la salida Y.

Y= S0'S1'D0 + S0'S1D1 + S0S1'D2 + S0S1D3

Ya con esta expresión es posible iniciar el diseño del circuito.

5.2 Diseño del circuito:

Con la función obtenida Y, se construye un circuito de seis entradas(D0,D1,D2,D3, S0,S1), donde las entradas D serán los canales, que para este caso base cada canal tendrá un solo bit y las entradas S serán los selectores. Fueron necesarias 4 compuertas AND, una compuerta OR y dos NOT, como se puede ver en la Figura 5.2

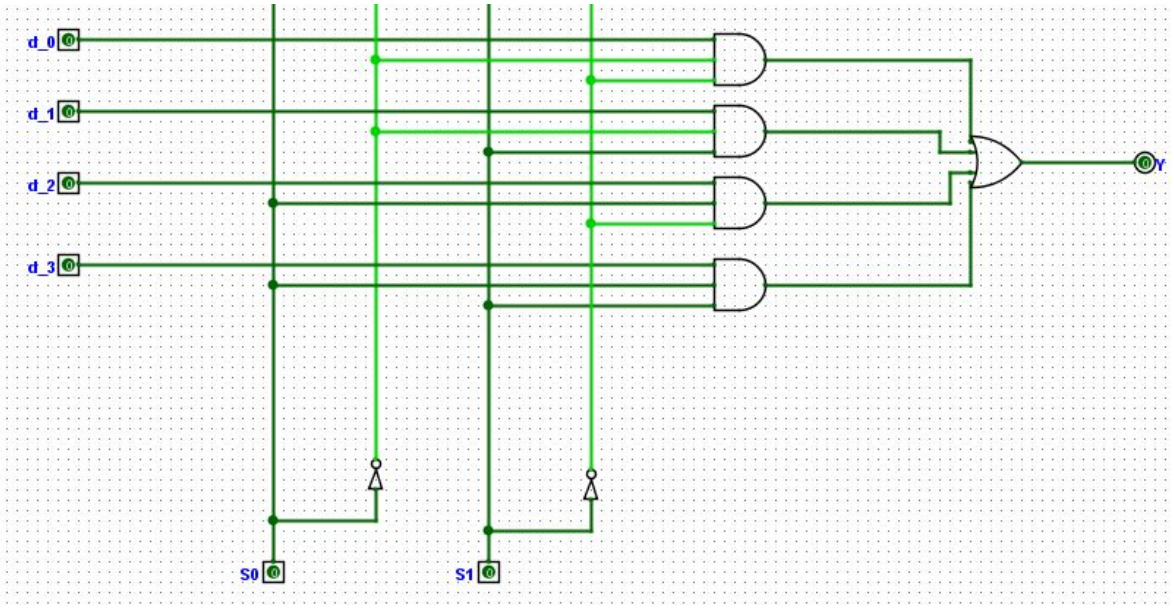


Figura 5.2, Circuito multiplexor 4:1 base

Teniendo en cuenta que es necesario que cada canal sea de 8 bits y ya se tiene un multiplexor de 1 bit en cada canal, se prosigue a crear un circuito combinacional que utilice el circuito multiplexor 4:1 base, que se obtuvo anteriormente para que sea capaz de recibir 8 bits por canal. Esto se logra simplemente usando 8 instancias del multiplexor para que cada canal reciba 1 bit del número de 8 bits que se está ingresando, así en el caso de que se active el canal D0, todos los bits de los 8 multiplexores del canal D0, se activaran, así los 8 bits del número pasaran por el canal D0. Lo anteriormente se dicho se logra visualizar en la figura 5.3

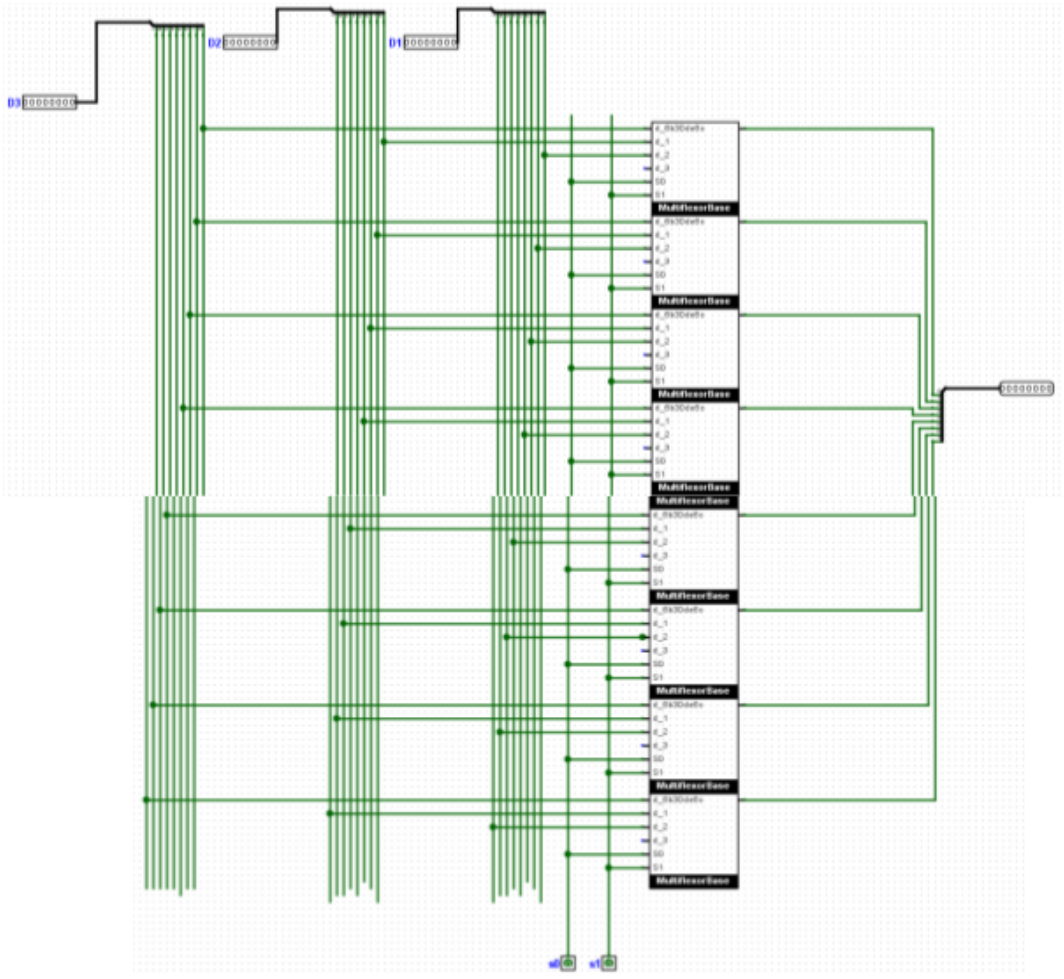


Figura 5.3, Circuito multiplexor 4:1

Es importante resaltar que hay tres entradas, ya que como se evidenció anteriormente el canal D4, es una situación de don't care.

6. **Circuito comparador de dos números y ordena de manera descendente**

Ya teniendo el comparador de 8 bits y el multiplexor 4: 1, ya será posible construir un circuito combinacional de dos entradas y dos salidas, que sea capaz de indicar en la salidas el número mayor y el menor respectivamente, en el caso de ser igual claramente en ambas salidas será el mismo número.

Para lograr el objetivo anterior solo basta con unir la salida del comparador de 8 bits con las entradas de selección de los dos multiplexores, un multiplexor se va a encargar de dar salida al numero mayor y el otro de indicar el número menor, este último se logra simplemente intercambiando las salidas del comparador, Figura 6.

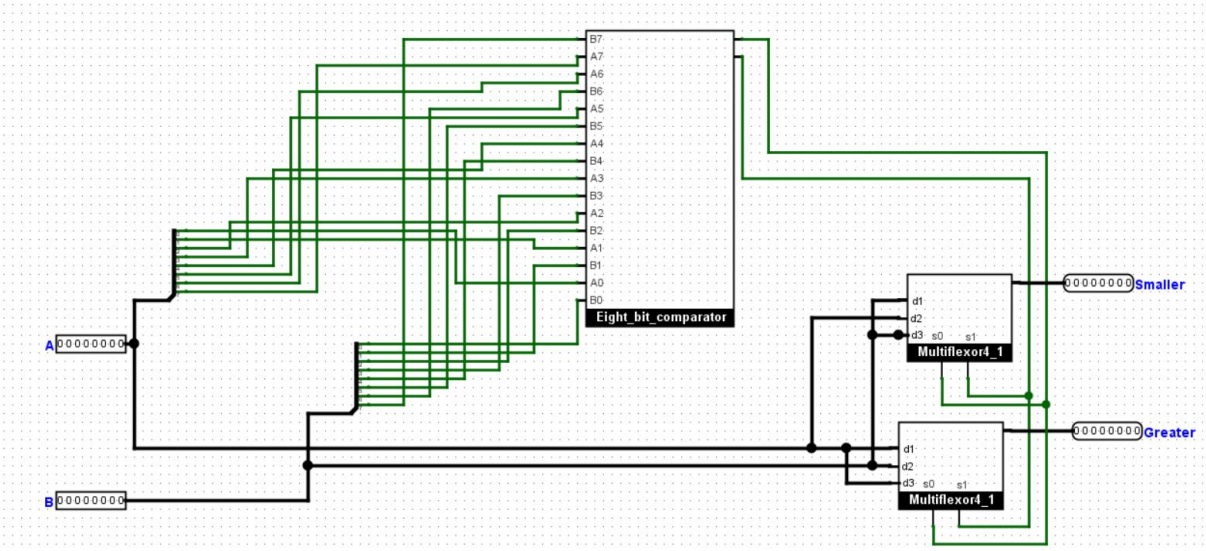


Figura 6, Circuito comparador de dos números y ordena de manera descendente

7. **Red de ordenamiento**

Una vez realizado el circuito anterior, el cual será la base para construir la red de ordenamiento de 8 números de 8 bits, se procede a buscar un algoritmo que sea capaz de lograr este fin.

Después de las investigaciones se llegó al acuerdo de aplicar el algoritmo de ordenamiento llamado **Biotonic sorter[1],** donde su principal premisa es que una estructura bitónica es una secuencia que cambia como máximo de dirección una sola vez.

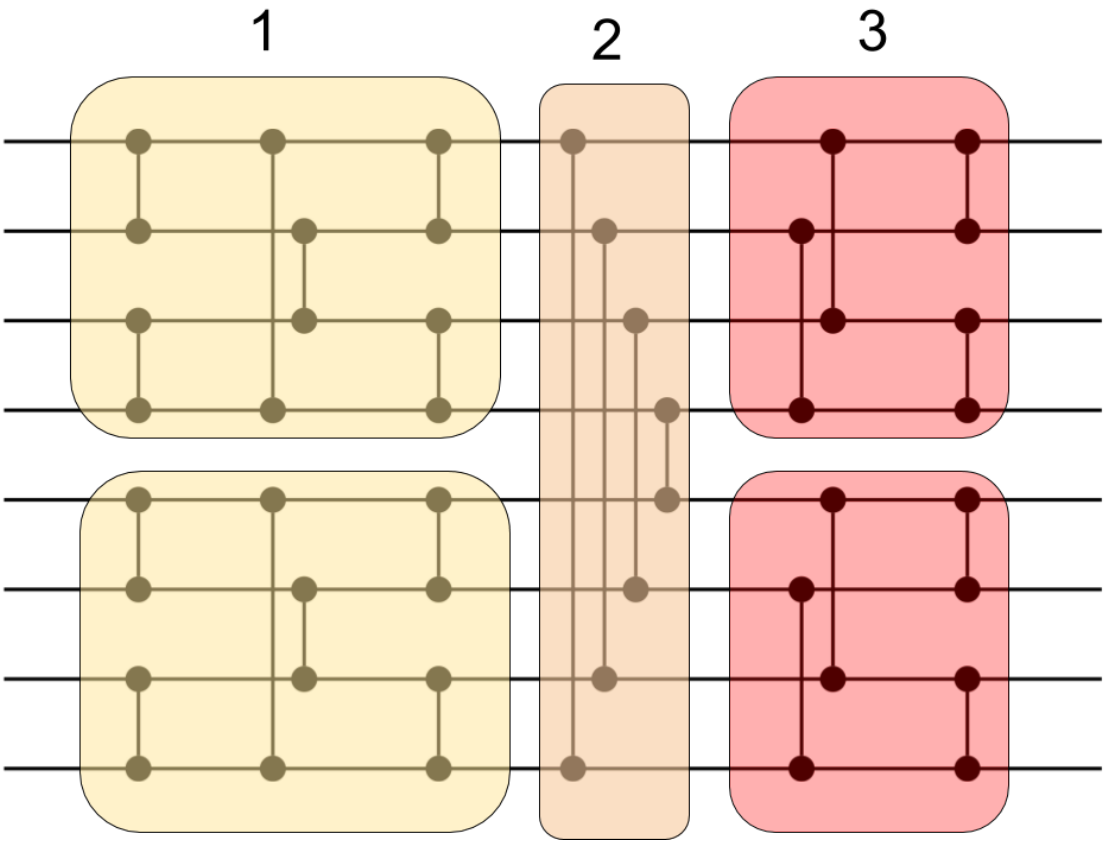


Figura 7, Ordenamiento bitónico de 8 números

El funcionamiento del algoritmo se puede dividir en 3 partes, Figura 7:

1. Los bloques amarillos tanto inferior como superior, cada uno se encarga de ordenar secuencias de 4 elementos.
2. El bloque naranja se encarga de dividir las dos secuencias ordenadas,en una mitad más grande y otra mitad más pequeña.
3. Finalmente, los dos bloques rojos se encargan de ordenar las 2 secuencias bitónicas.

7.1 Diseño del circuito:

Por lo que se evidencia en el numeral anterior la red de ordenamiento se resume a crear solo tres circuitos combinacionales, donde su elemento principal será el circuito comparador de dos números.

Siguiendo la estructura que maneja el ordenamiento bitónico, los circuitos que se obtienen son los siguientes:

7.1.1 Circuito ordenador descendente de 4 números

Este circuito consta de 4 entradas y 4 salidas, donde fue necesario usar 6 comparadores de dos números. Figura 7.1.1

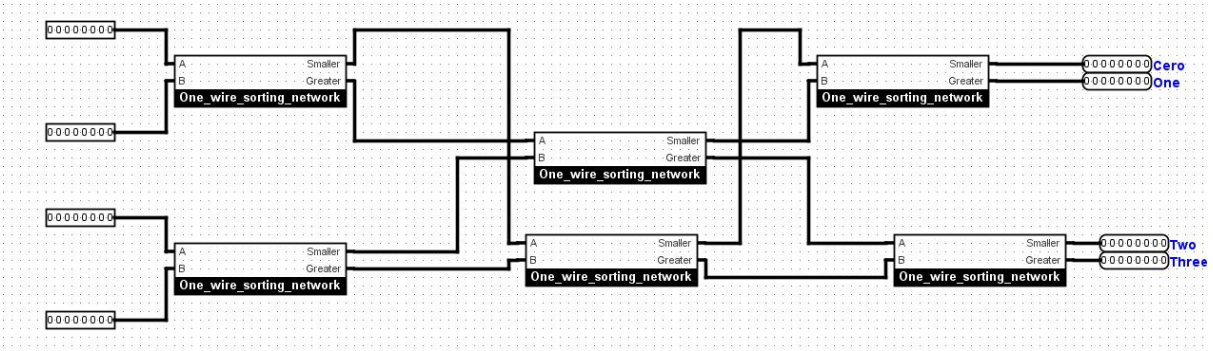


Figura 7.1.1 Circuito ordenador descendente de 4 números

7.1.2 Circuito para dividir dos secuencias ordenadas

Como se mencionó anteriormente, este circuito va a dividir las dos secuencias en una mitad más grande y otra más pequeña. Este circuito consta de 8 entradas y 8 salidas, donde fue necesario usar 4 comparadores de 2 números, uno por cada par de números.

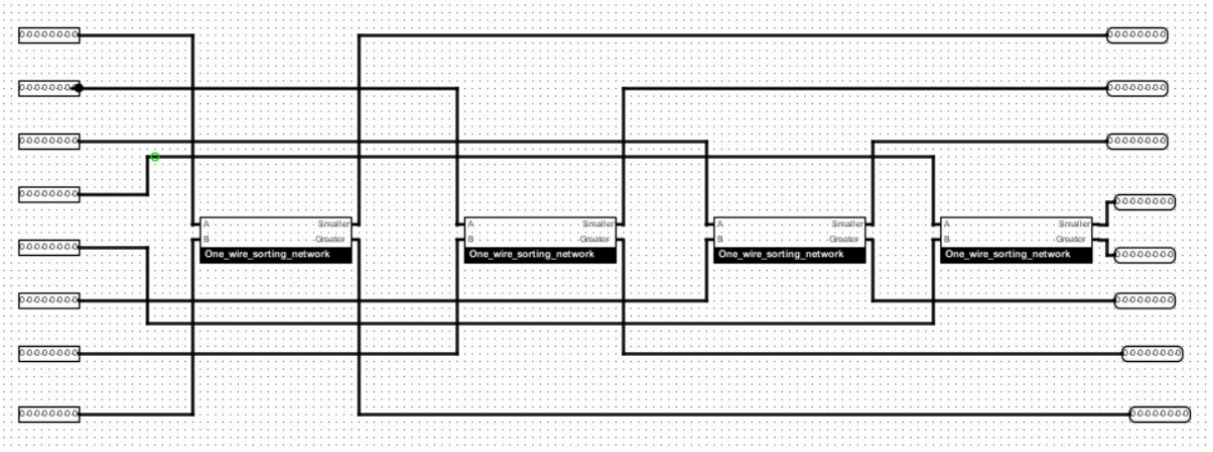
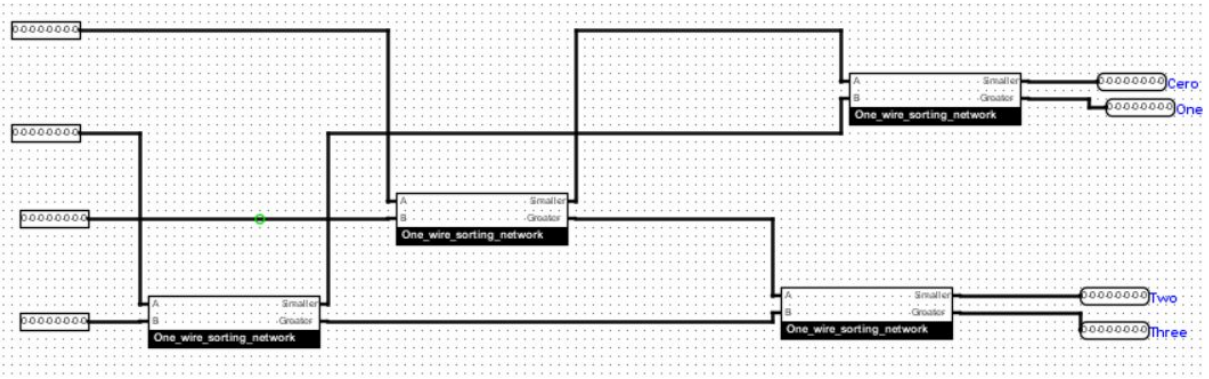


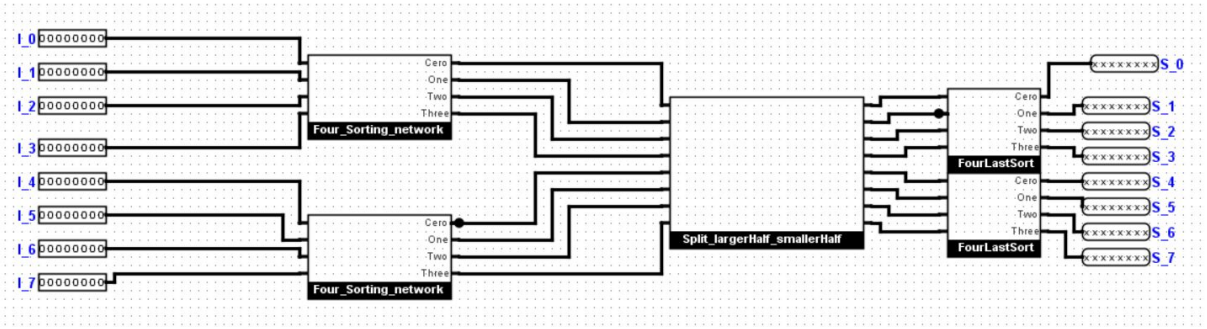
Figura 7.1.2 Circuito ordenador descendente de 4 números

7.1.3 Circuito para ordenar dos secuencias bitónicas

Por último,este circuito se encarga de ordenar las secuencias bitónicas resultantes. Este circuito consta de 4 entradas y 4 salidas, donde fue necesario usar 4 comparadores de 2 números.

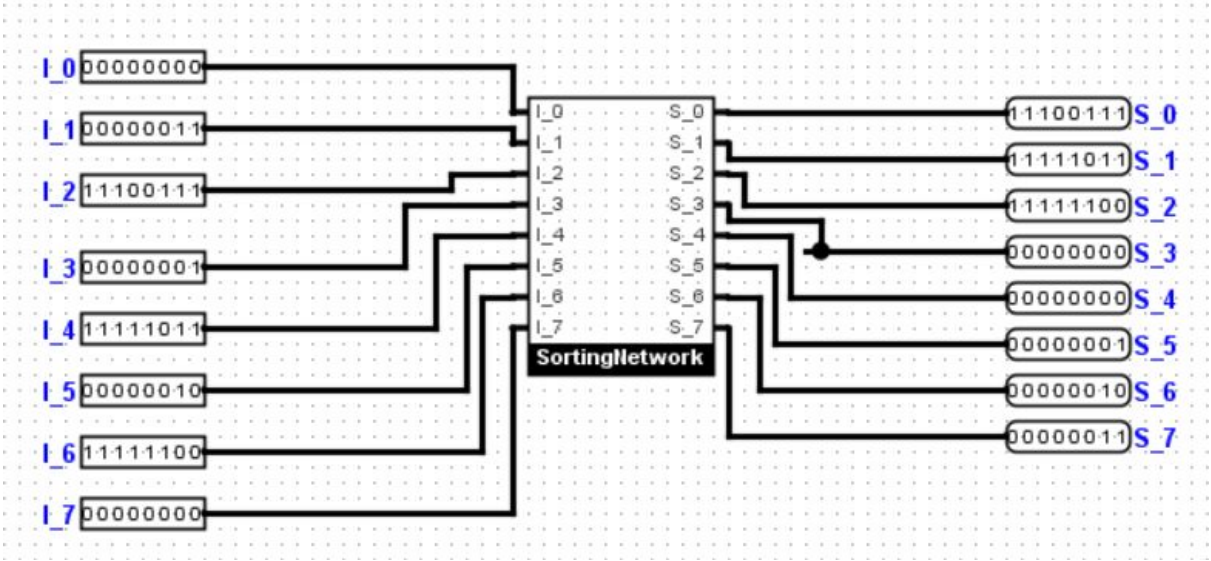


Una vez realizados los circuitos anteriores, se procede a construir el circuito combinacional principal, siguiendo la estructura propuesta por el ordenador bitónico.



Como se describió al inicio de este numeral este circuito consta de 2 circuitos encargados ordenador de manera descendente 4 números, luego el circuito de en medio divide las dos secuencias ordenadas y finalmente los 2 últimos circuitos se encargan de ordenar las dos secuencias bitónicas.

Así finalmente se obtiene el circuito combinacional encargado de ordenar 8 números de 8 bits en forma descendente



Conclusiones:

- Los circuitos combinacionales ideales son desarrollados de manera jerárquica y estructurada, lo que permite un mejor diseño, funcionamiento y manejo del circuito.
- La teoría fundamental de los circuitos combinacionales como las tablas de verdad, mapas de Karnaugh, simplificación de funciones booleanas y demás permiten la simulación de circuitos combinacionales de manera lógica, estructural y funcional.
- Las estructuras más amplias son, generalmente, instanciadas por unas más concisas. Como el comparador de 8 bits compuesto por comparadores de 2 bits utilizados en la red de ordenamiento.
- El algoritmo de red de ordenamiento Biotonic sorter nos permitió el manejo de los comparadores de una manera lógicamente estructurada, lo que da como resultado una red óptima y exitosa.

Bibliografía

[1] Biotonic Sorter. Wikipedia. Obtenido de : https://en.wikipedia.org/wiki/Bitonic_sorter