Informe

Laboratorio No.1

Isaac Jiménez Fernández
C.C. 1003059949

Jose Carlos Ortiz Padilla
C.C. 1001651897

Universidad de Antioquia
Facultad de Ingeniería
Departamento de Sistemas

Medellín

2023

Introducción

El presente informe de laboratorio consiste en la explicación del proceso de diseño que se llevó a cabo para la elaboración del problema dado en el laboratorio 1, teniendo en cuenta las especificaciones dadas del problema. El desarrollo del informe se realizó mediante la herramienta de simulación *Logisim Evolution,* el cual permitió simular y mostrar los resultados del diseño e implementación. Basándonos en los mapas de Karnaugh y las tablas de verdad creadas por nosotros mismos para la elaboración y correcto funcionamiento de los diferentes circuitos necesarios para la elaboración de la red de ordenamiento.

Planteamiento del problema del problema

Se busca crear un sistema de reduccion en varias fases mediante comparadores y un contador para así mostrar los resultados en displays de 7 segmentos con números de 8 bits (negativos implementados mediante complemento a dos), en este caso, se utilizará un sistema de reduccion buscando el mayor y flip flops tipo SR, y un patron de ordenamiento 0&1, 2&3… ya que es el correspondiente al número del equipo, módulo 16.

10	Mayor	SR	0&1, 2&3, ...

Descripción del diseño

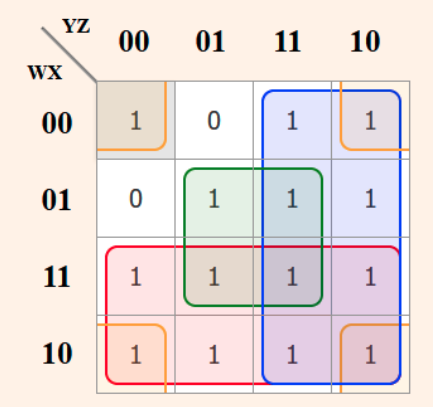
1) **Display 7 segmentos.**

Se creó la tabla de verdad del display de 7 segmentos de 4 bits ya que con estos son suficientes para cubrir los dígitos del 0-9.

W	X	Y	Z	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
1	0	1	0	1	1	0	1	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	0	1	1	1	0	0	1	1
1	1	0	1	1	0	1	1	0	1	1
1	1	1	0	1	0	1	1	1	1	1
1	1	1	1	1	1	1	0	0	1	1

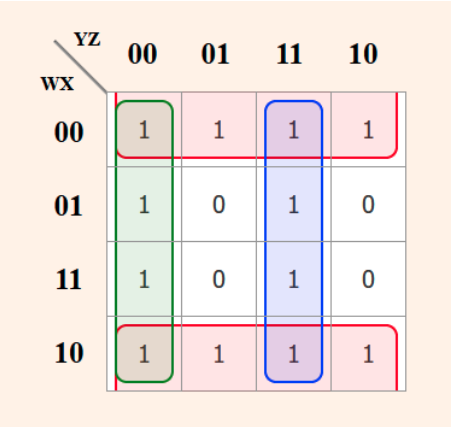
Con la tabla, se realizan los mapas de Karnaugh para determinar las expresiones de cada salida.

Salida A:



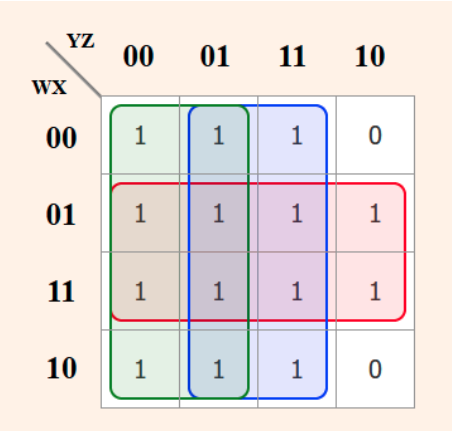
A= X´Z´+Y+XZ+W

Salida B:



B=X´+Y´Z´+YZ

Salida C:



C=Y´+Z+X

Salida D:

WX \ YZ	00	01	11	10
00	1	0	1	1
01	0	1	0	1
11	0	1	0	1
10	1	0	1	1

D=X′Z′+X′Y+YZ′+XY′Z

Salida E:

WX \ YZ	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	0	0	0	1
10	1	0	0	1

E=X′Z′+YZ′

Salida F:

WX \ YZ	00	01	11	10
00	1	0	0	0
01	1	1	0	1
11	1	1	1	1
10	1	1	1	1

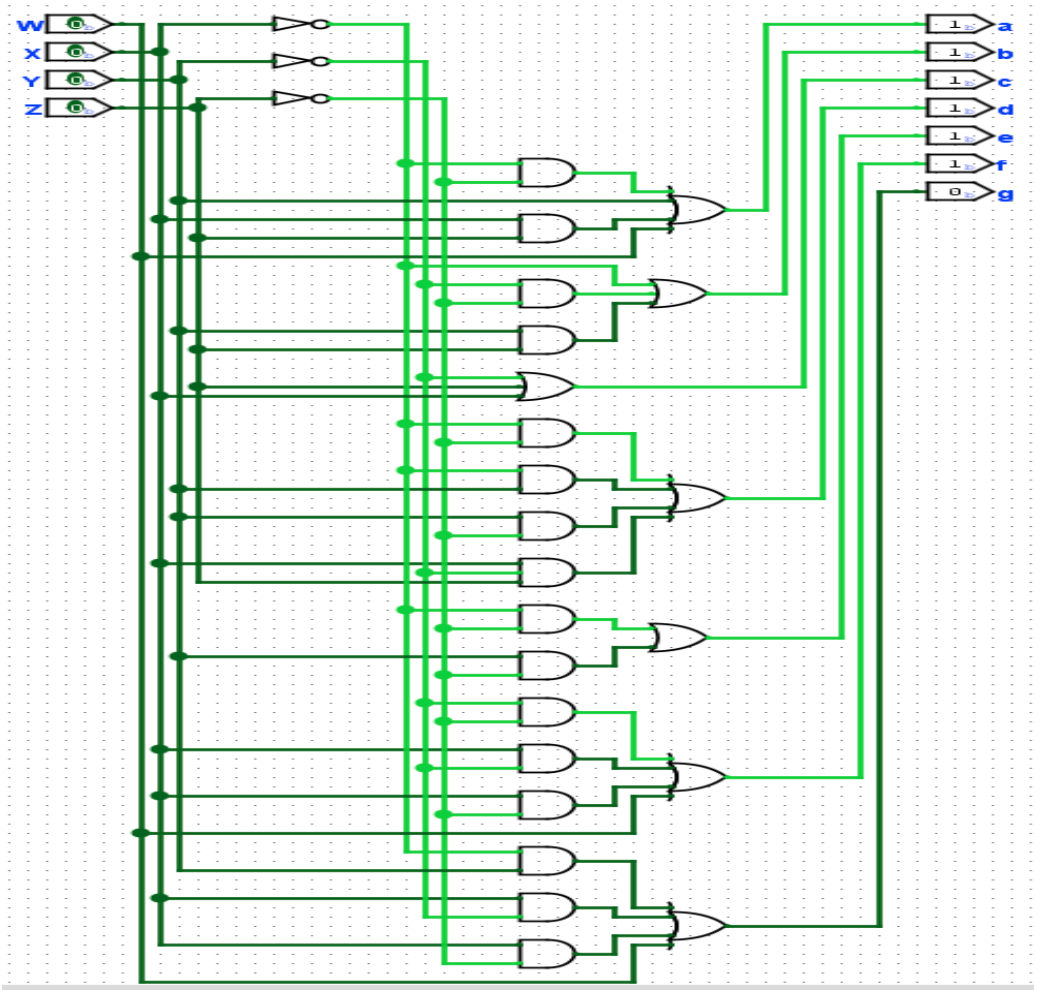
F=Y′Z′+XY′+XZ′+W

Salida G:

WX \ YZ	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	1	1	1	1
10	1	1	1	1

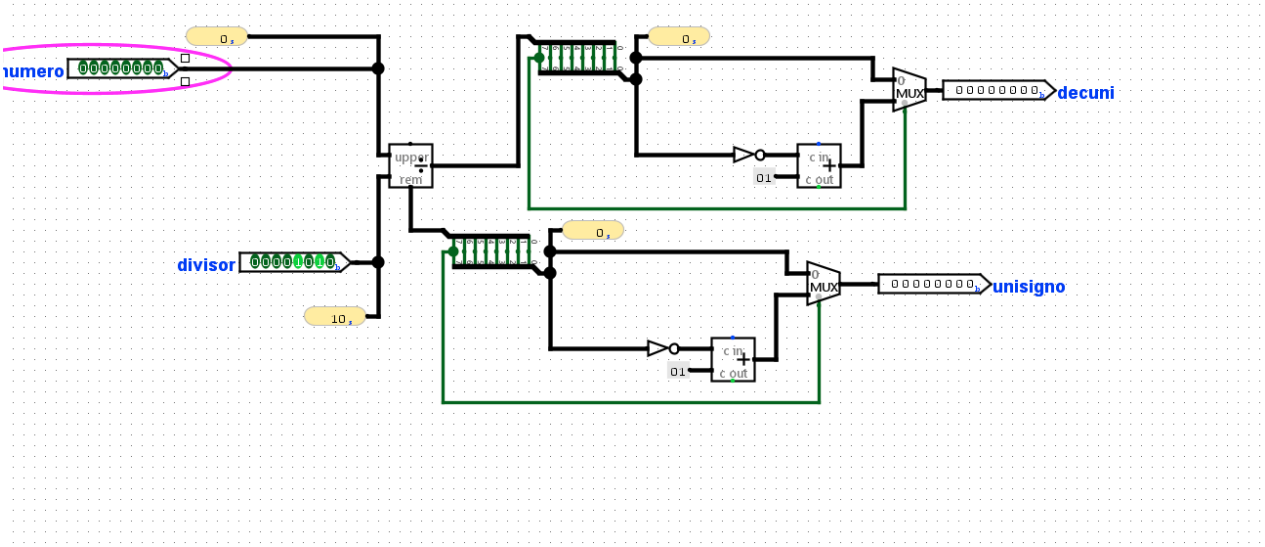
G=X´Y+XY´+XZ´+W

Implementando las expresiones, el circuito para el display de 7 segmentos queda así:

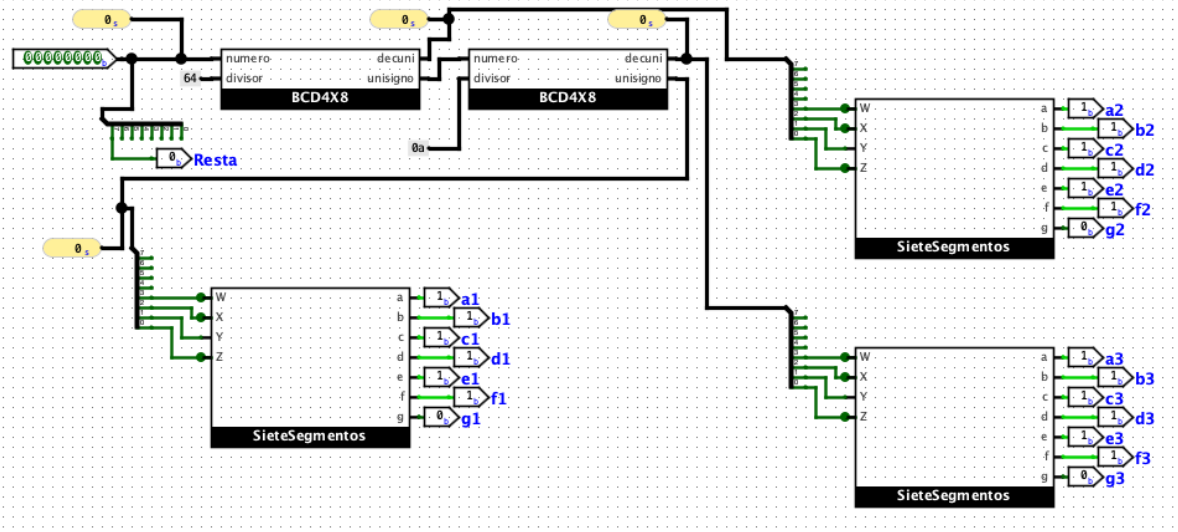


Una vez creado el circuito para el display de 7 segmentos, se debe hacer una modificación ya que este fue realizado para números de 4 bits y el sistema trabaja con números de 8 bits, por lo tanto, se deben unir dos de 4 bits para obtener uno de 8 bits, este circuito fue llamado ***BCD4X8.***

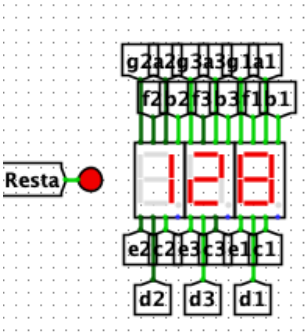
Para este circuito, se utilizó un divisor de 8 bits en complemento a 2 que permite separar los números. Se divide entre diez y cada número separado entra a un sumador, esto se hace más que todo por los números negativos, puesto que juntamos el último bit con el sumador con un multiplexor para que después nos de su valor en complemento a 2.



Con estas modificaciones, se crea el circuito que permite que un número de 8 bits se muestre en 3 displays de 7 segmentos.



Para representar los números negativos se dispuso de un led que se activa demostrando que el número es una cantidad negativa.



2) Comparador con entradas de 8 bits.

Para diseñar el comparador de 8 bits se empleó el diseño jerárquico, realizando primero un comparador de 1 bit y posteriormente, se usa este para crear el de 8 bits.

Para diseñar el comparador de 1 bit, se implementó un circuito que funciona de manera correcta únicamente con números positivos:

Tabla de verdad ***Comparador1Bit***:

P1	P0	D1	D0	C1	C0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	0	0
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	0	1
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	1	0
1	0	1	0	1	0
1	0	1	1	1	0
1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	1	1

Y con la tabla de verdad, se desarrollan los mapas de Karnaugh para las salidas C1 y C0 y así, encontrar las expresiones minimizadas que nos permitirán implementar el circuito.

Mapas de Karnaugh

C0

P1P0 \ D1D0	00	01	11	10
00	0	1	0	0
01	1	1	1	1
11	1	1	1	1
10	0	0	0	0

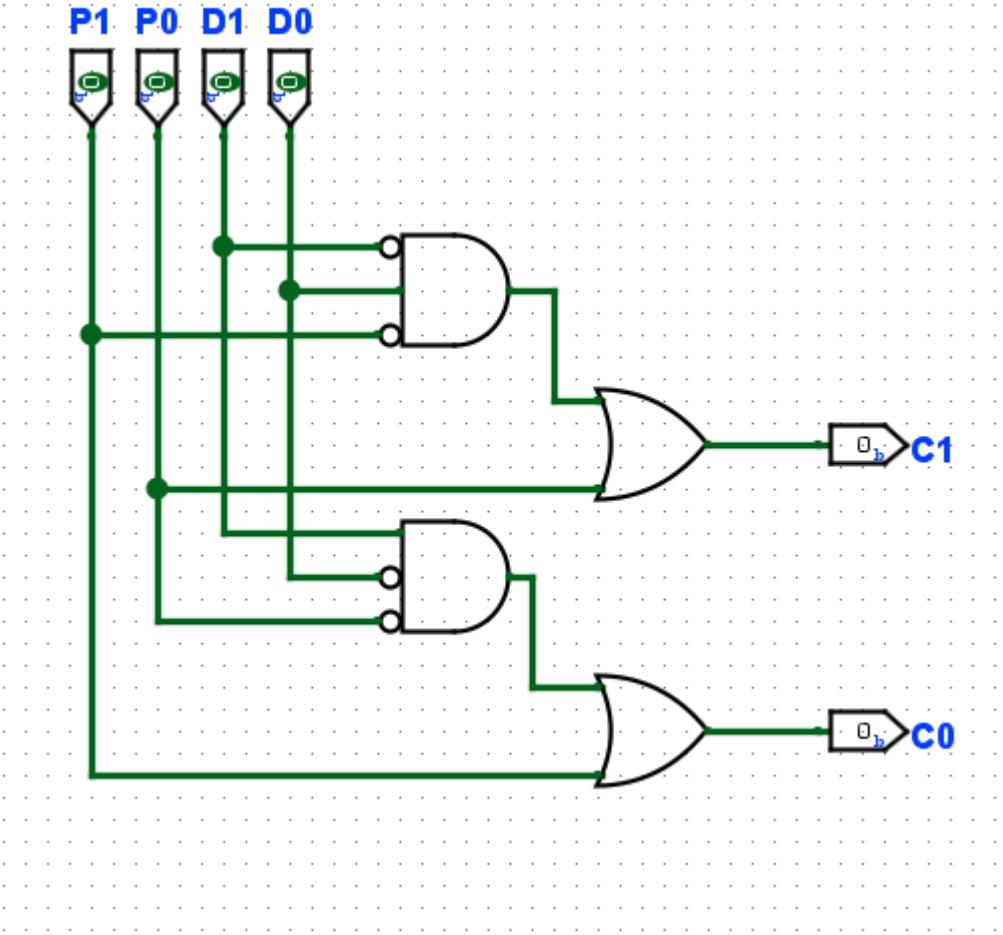
C0=P1´D1´D0+PO

C1

P1PO \ D1D0	00	01	11	10
00	0	0	0	1
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

C1=P0´D1D0´+P1

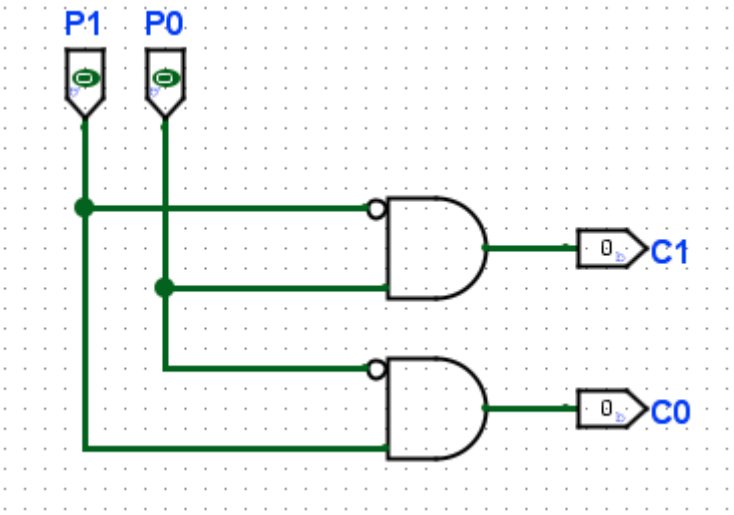
Con estas expresiones, se implementa el circuito ***Comparador1Bit***:



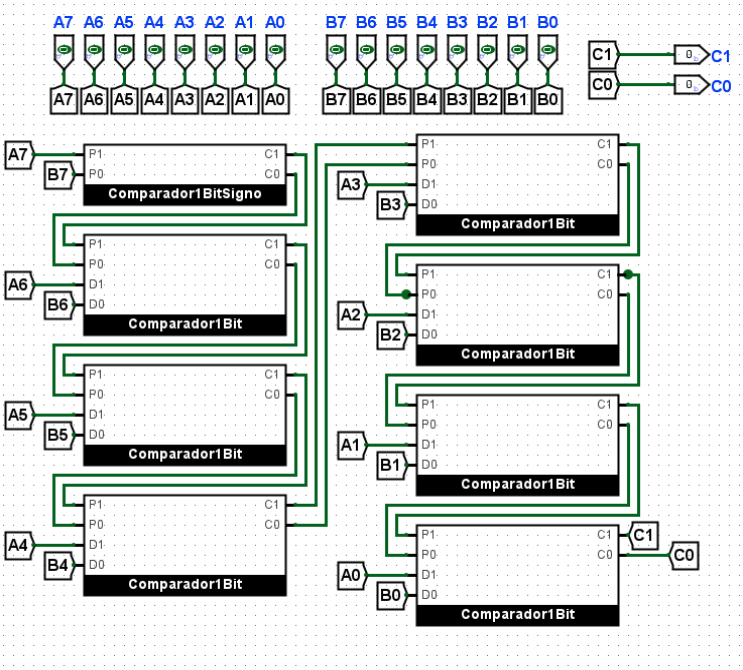
Para el correcto funcionamiento del comparador de 8 bits al ingresar números con signo, se diseñó otro circuito llamado ***Comparador1BitSigno*** mediante la siguiente tabla de verdad:

P1	P0	C1	C0
0	0	0	0
0	1	1	0
1	0	0	1
1	1	0	0

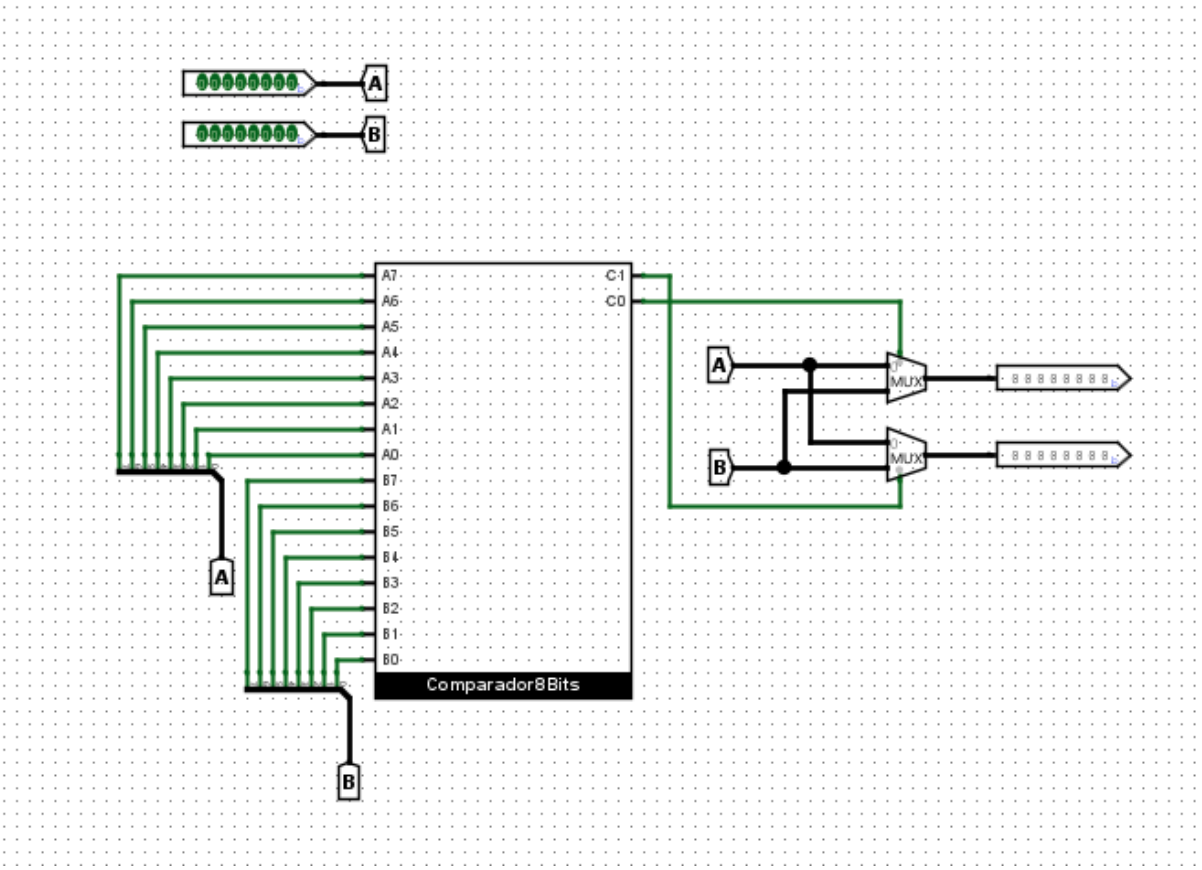
Implementación:



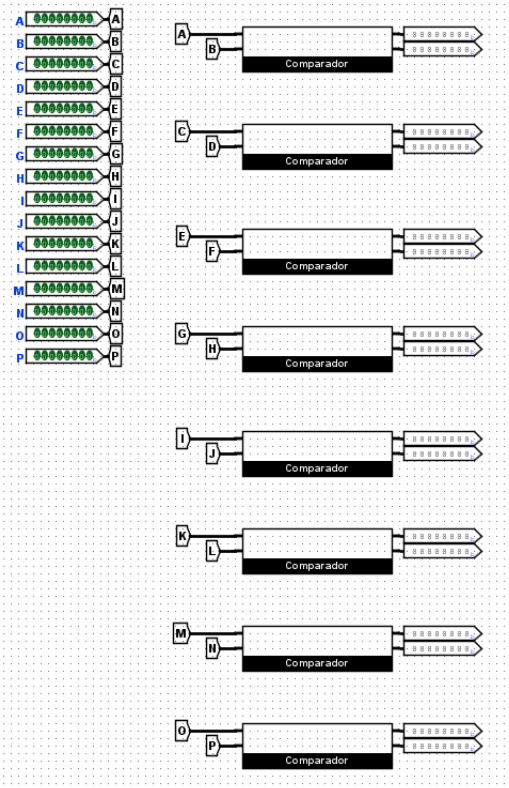
Este circuito permite el ingreso de números con signo sin afectar el comportamiento del comparador, para ello se une junto a otros siete comparadores de 1 bit para lograr un comparador con dos entradas de números de 8 bits:



Este circuito, tiene como salidas C1 y C0 que indican si el número es menor, mayor o igual mediante salidas binarias, ya que este comparador es mayor, para que el circuito entregue el número mayor en su primer salida y el menor en la segunda, se implementaron multiplexores en otro circuito nombrado ***Comparador*** de la siguiente manera:



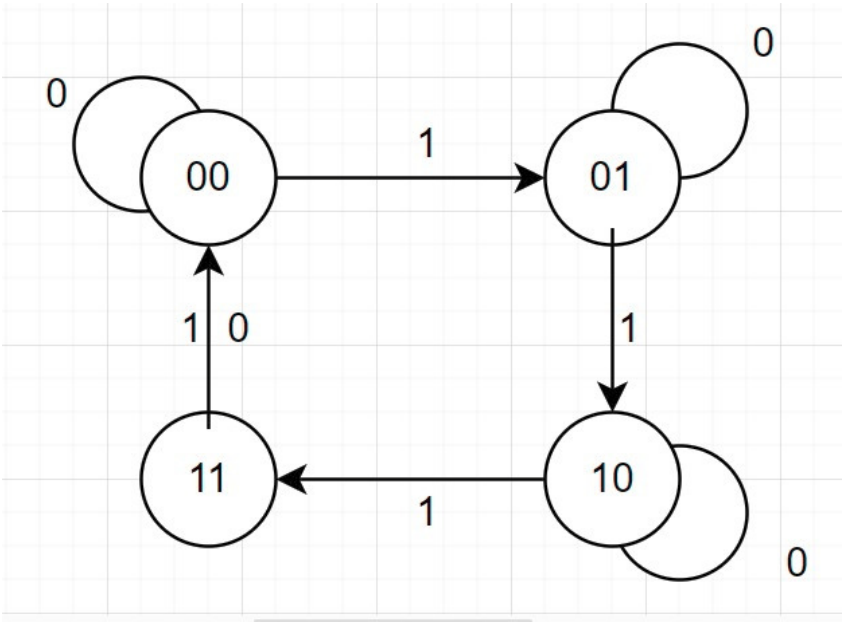
Por último creamos un circuito llamado ***Comparar16Num*** el que nos permite con solo este circuito hacer la comparación de todos los 16 números, lo que hace que sea menos circuitos a implementar y más eficiente.



3) Máquina de Estados Finita

Para la realización del contador se realizó una máquina de estado finito con flip flop tipo SR. Este contador tiene 4 estados

Diagrama de estados:



Con este diagrama de estados, se puede diseñar la tabla de verdad que describa este comportamiento:

X	Q1	Q0	S1	R1	S0	R0
0	0	0	0	-	0	-
0	0	1	0	-	-	0
0	1	0	-	0	0	-
0	1	1	0	1	0	1
1	0	0	0	-	1	0
1	0	1	1	0	0	1
1	1	0	-	0	1	0
1	1	1	0	1	0	1

Mapas de Karnaugh

S1

	Q1/Q0			
X	0	0	0	X
	0	1	0	X

S1=X.Q1.Q0

R1

	Q1/Q0			
X	X	X	1	0
	X	0	1	0

R1=X.Q0

S0

	Q1/Q0			
X	0	X	0	0
	1	0	0	1

S0=Q1´Q0

R0

	Q1/Q0			
X	X	0	1	X
	0	1	1	0

R0=Q1.Q0+X.Q0

4) Fulladder y Halladder

Se crearon estos dos circuitos para incrementarle 1 bit a las fases y mostrar el número de la fase de una forma ordenada.

Halladder

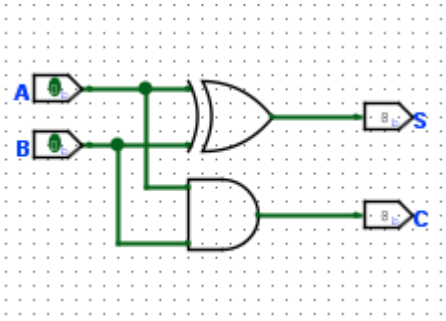


Tabla de verdad:

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Fulladder

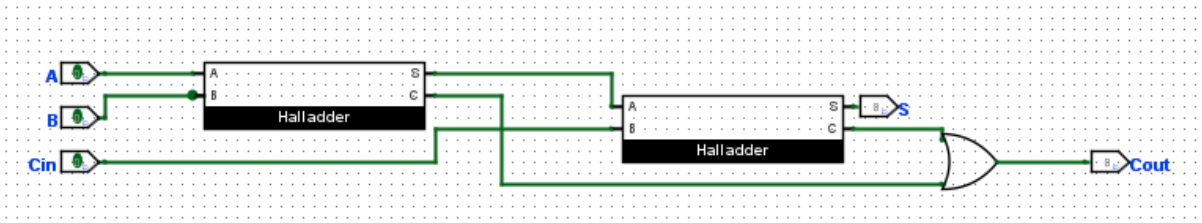


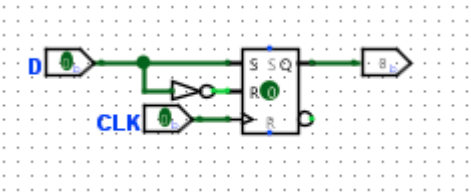
Tabla de verdad:

A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

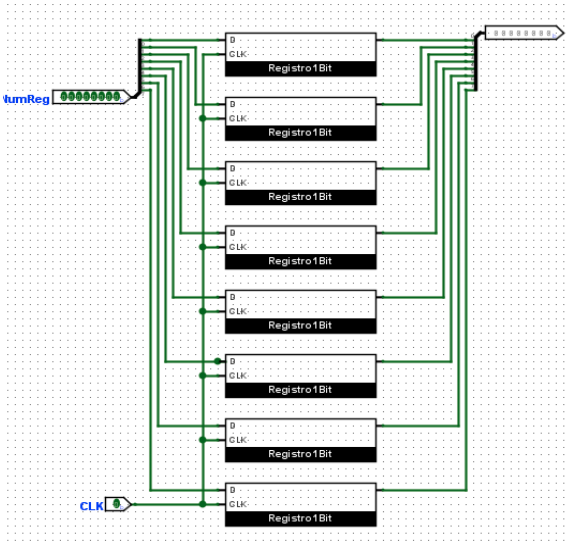
5) Registros

Lo primero que realizamos para el funcionamiento del registro final fue crear 1 circuito llamado ***Registro1Bit*** en el cual como lo dice su nombre se busca guardar el registro de 1 bit.

Registro1Bit

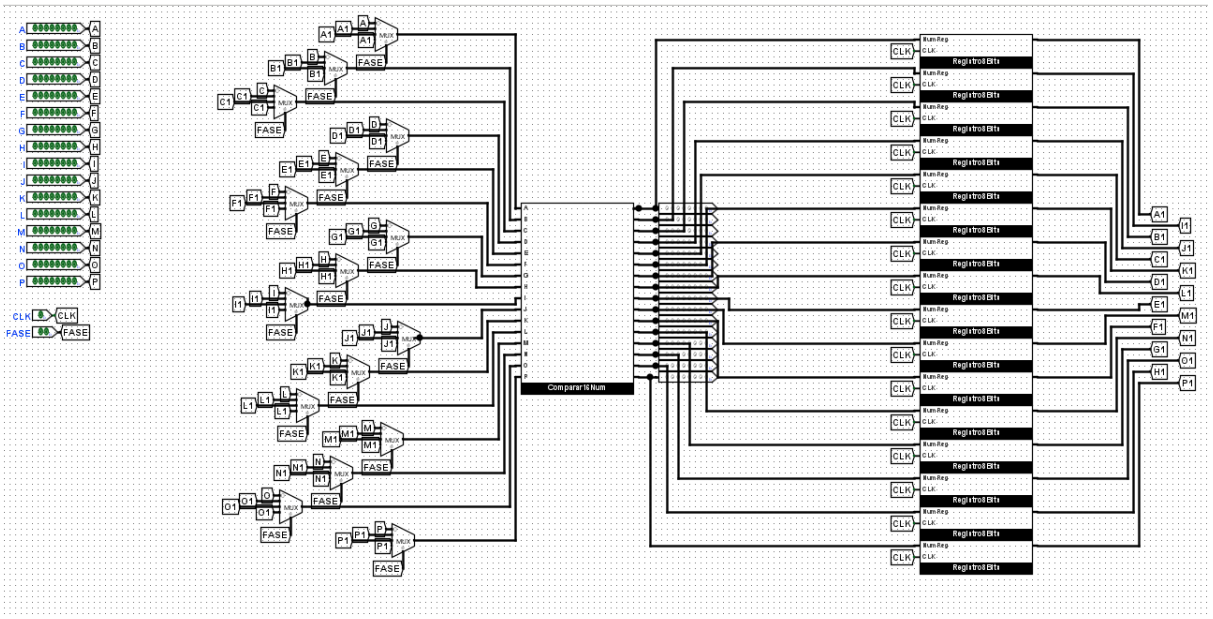


Ya implementado **Registro1Bit** así como hicimos con el comparador juntamos 8 **Registro1Bit** para crear un registro de 8 bits el cual llamamos ***Registro8Bits.***



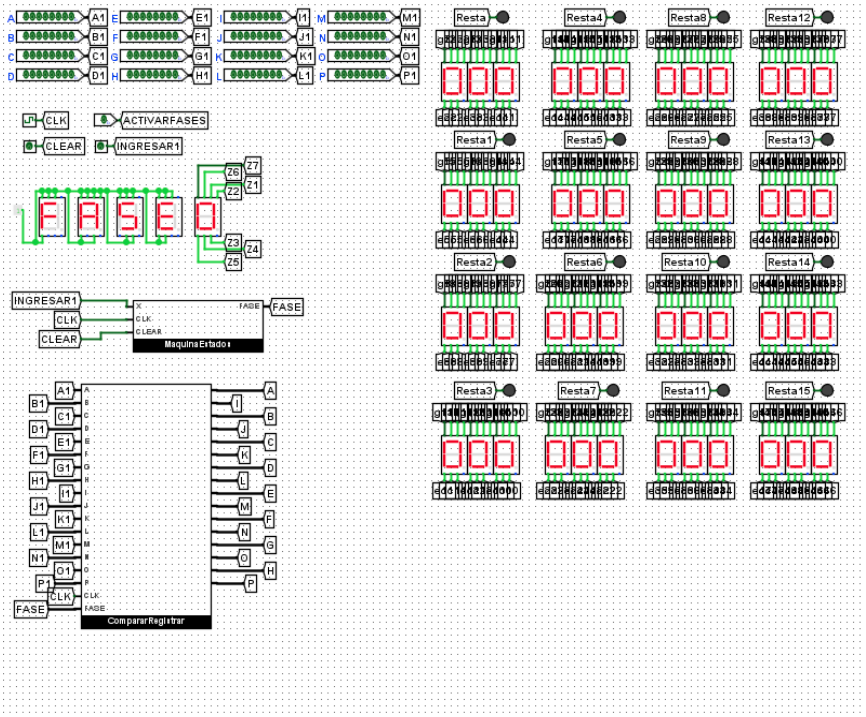
6) Comparar Registrar

Al tener hechos el registro de 8 bits y también tener el comparador creamos un circuito llamado ***CompararRegistrar*** en el que utilizamos 1 comparador de 16 números y 16 registros de 8 bits con los que permiten con unos multiplexores usar los mismos registros de 8 bits para guardar los números ya comparados fase por fase haciendo de esto un sistema jerárquico.



7) Circuito completo:

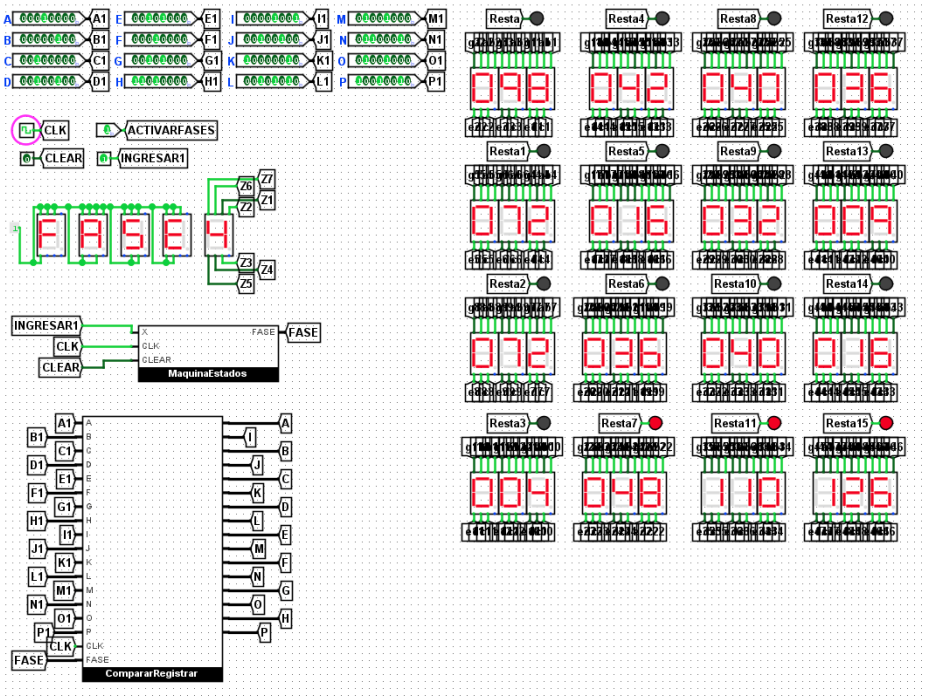
Con los elementos necesarios para sistema de reducción y ordenamiento ya completados, se monta el circuito principal en ***main(***también está los 16 displays de 7 segmentos, pero por estética de la imagen no se pusieron)



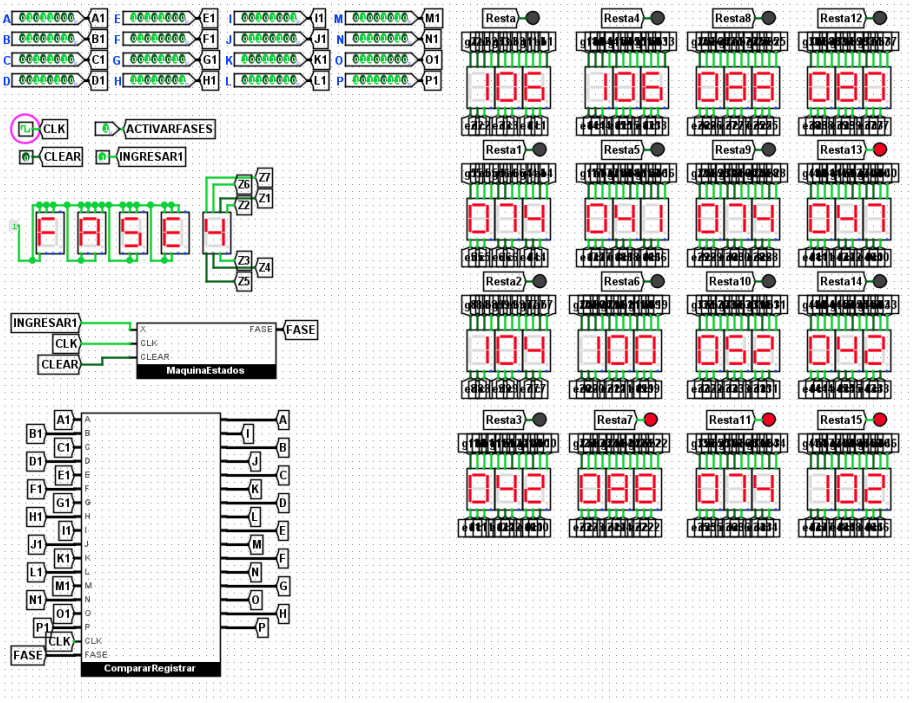
Este circuito se compone de varias partes que fueron montadas con los elementos diseñados y completados anteriormente, los números son mostrados en displays de 7 segmentos como se mencionó anteriormente y además con la máquina de estados, se indica a qué fase corresponde este ordenamiento. Para activar correctamente el programa deben estar en 1 el ActivarFases e ingresar1 para que así comience en la fase 1, Finalmente en la parte inferior derecha se pueden ver los bloques que son los que permiten que los números que son arrojados en cada fase y el de la fase, puedan ser mostrados en sus respectivos displays.

Para verificar su correcto funcionamiento se realizaron tres pruebas con números aleatorios, con signo, sin signo y de igual magnitud:

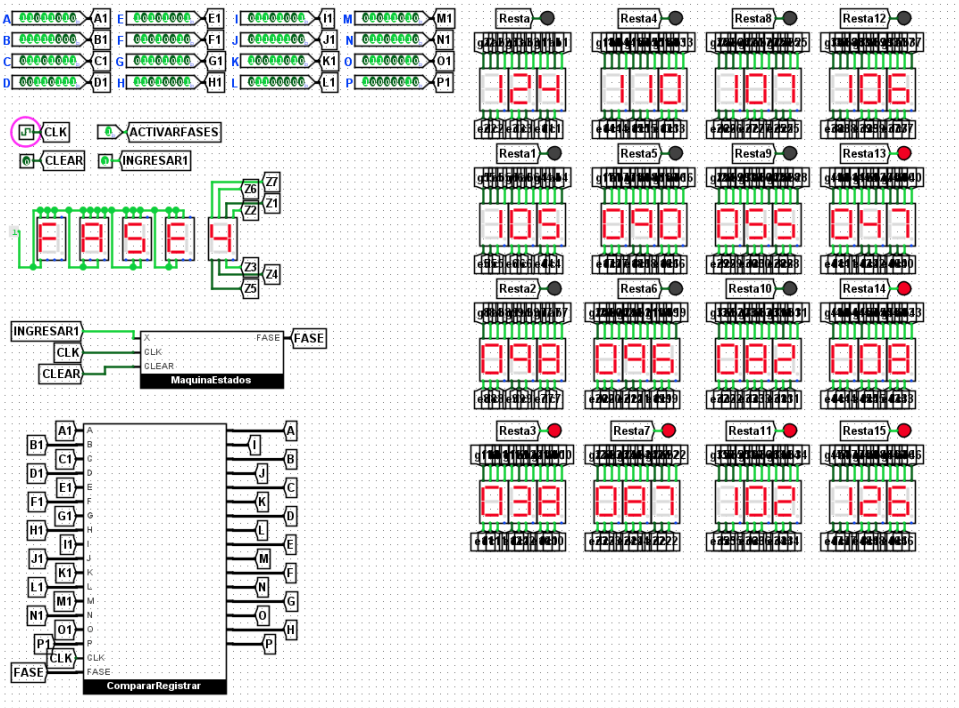
Simulación No.1



Simulación No.2



Simulación No.3



Conclusiones

- Al igual que con la creación e instanciación de métodos cuando se escribe un programa, los módulos básicos facilitan y simplifican la creación de módulos más complejos.
- Las tablas de verdad y los mapas de karnaugh dan una versión simplificada pero no siempre es la más eficiente.
- Los túneles, multiplexores y separadores permiten obtener un circuito más organizado y esto se puede visualizar con la implementación de todos los displays de 7 segmentos en el ***main.***

Enlace vídeo explicativo: https://youtu.be/S74n7woWzdY