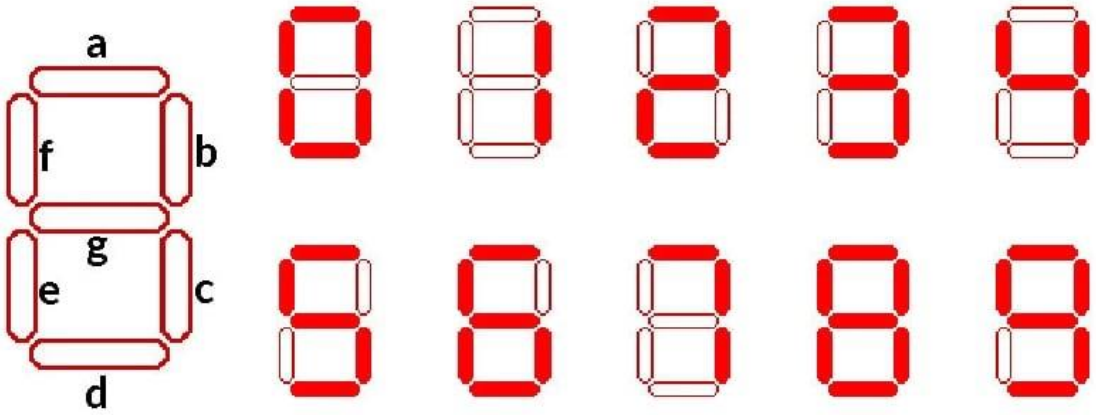
Practica 5

Decodificador

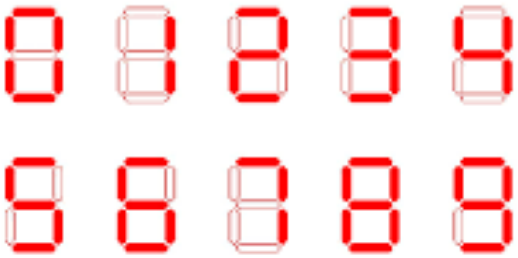
Profesor: Barrón Vera José Emanuel
Materia: Fundamentos de diseño digital
Grupo: 3CV6
Alumno: Cazares Cruz Jeremy Sajid
Boleta: 2021630179



Desarrollo experimental

El desarrollo de la practica consiste en la realización de un decodificador de 4 entradas y 7 salidas con la finalidad de que las salidas se muestren en un display de 7 segmentos, por ello las 7 salidas en el decodificador.

La función del decodificador es dar una representación a lenguaje BCD del dato introducido en la entrada, de tal manera que el número binario introducido en las entradas se pueda visualizar en la salida mediante el display como si se tratara de un número normal, con esto se hace referencia lo siguiente.



Como se observa en la figura 1 es la manera en que se pueden visualizar números dentro del display, teniendo como objetivo la practica el poder visualizar los mismos mediante el uso del sistema BCD, obteniendo números más notables o por lo menos más “reconocibles”

figura 1

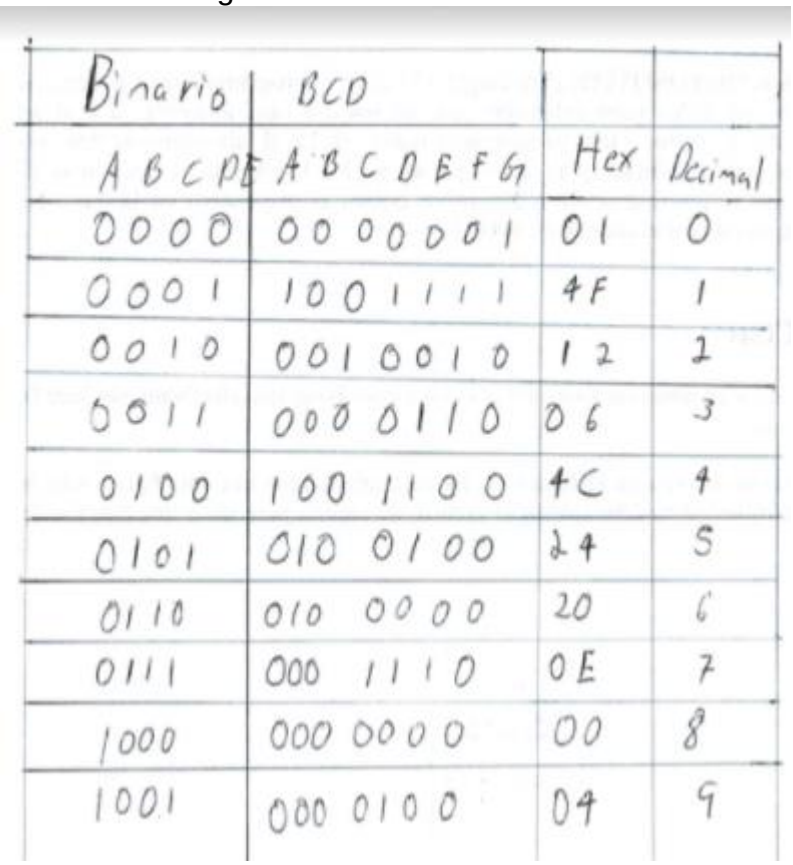
Binario	BCD	Hex	Decimal
A B C D	A B C D E F G	Hex	Decimal ó mensaje
0 0 0 0	0 0 1 0 0 1 0	1 2	2
0 0 0 1	0 0 0 0 0 0 1	0 1	0
0 0 1 0	0 0 1 0 0 1 0	1 2	2
0 0 1 1	1 0 0 1 1 1 1	4 F	1
0 1 0 0	0 1 0 0 0 0 0	2 0	6
0 1 0 1	0 0 0 0 1 1 0	0 6	3
0 1 1 0	0 0 0 0 0 0 1	0 1	0
0 1 1 1	1 0 0 1 1 1 1	4 F	1
1 0 0 0	0 0 0 1 1 1 0	0 E	7
1 0 0 1	0 0 0 0 1 0 0	0 4	9

De igual manera al solo tener un display el número más grande que se puede obtener es el nueve de tal manera que si se quiere realizar un mensaje se podría hacer de manera secuencial por lo que la entrada binaria “0000” en lugar de dar como resultado el número BCD “0000001” dará como resultado el BCD del primero número que forma parte del mensaje, siendo así que en la combinación “0000” el resultado será “0010010” formando un número dos por el cual la mayoría de boletas inician así prosiguiendo con los números faltantes dentro del mensaje, esto se

figura 2 tabla de verdad de boleta 2021630179

puede observar mejor en la figura 2 donde existe una pequeña tabla de verdad del mensaje.

Mientras que la figura 3 sirve de referencia a los resultados que existirán dentro del gal en el simulador.



Binario		BCD							Hex	Decimal
A	B	C	D	E	A	B	C	D	E F G		
0	0	0	0		0	0	0	0	0 0 1	01	0
0	0	0	1		1	0	0	1	1 1 1	4F	1
0	0	1	0		0	0	1	0	0 1 0	12	2
0	0	1	1		0	0	0	0	1 1 0	06	3
0	1	0	0		1	0	0	1	1 0 0	4C	4
0	1	0	1		0	1	0	0	1 0 0	24	5
0	1	1	0		0	1	0	0	0 0 0	20	6
0	1	1	1		0	0	0	1	1 1 0	0E	7
1	0	0	0		0	0	0	0	0 0 0	00	8
1	0	0	1		0	0	0	0	1 0 0	04	9

figura 3 tabla de verdad mensaje

Al momento de la simulación, se hace uso también de la base hexadecimal ya que al no extender los números de entrada los resultados se verán de manera hexadecimal, una vez exponiendo estos valores se puede ver la forma de onda completa de los datos, así como el binario de las mismas como se puede observar en las siguientes figuras.

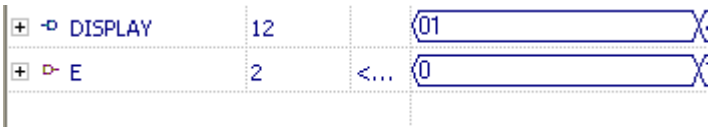


figura 4 Hexadecimal de display

En la figura 5 se puede ver a mayor detalle lo antes mencionado, lo cual es que dando clic en el simulador nos da una versión más detallada de la forma de ondas que se tiene tanto de entrada como de resultado también indicando el número binario, tanto en forma de onda como en decimal mediante la columna de “value”

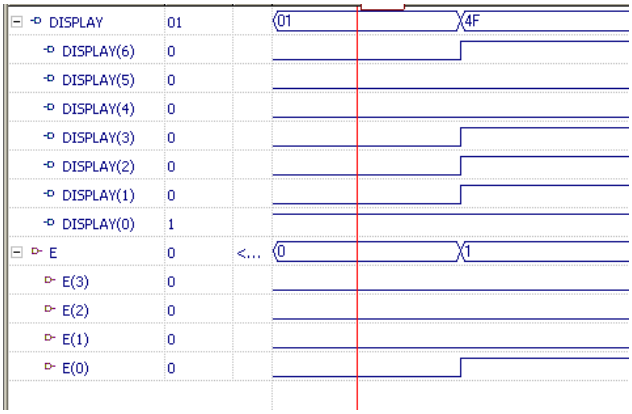


figura 5 expansión de valores

De igual manera el hexadecimal es una forma “resumida” de los datos obtenidos, de tal manera que se muestra aun cuando se hayan expandido los datos de entrada y salida

Como ya se mencionó la función principal del código es el poder mostrar el número binario a un display de 7 segmentos esto se hace de la siguiente manera

```
1  LIBRARY ieee;
2  USE ieee.std_logic_1164.all;
3
4  ENTITY EDeco IS PORT(
5      E:       IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
6      DISPLAY: OUT STD_LOGIC_VECTOR(6 DOWNTO 0)
7      );
8
9  END EDeco;
```

figura 6 código entity

Tal como en la práctica pasada se utilizó la palabra reservada “ENTITY” sirve para declarar la cantidad de pines de entrada y salida mientras que de igual en este código tiene una utilidad similar siendo la diferencia que se declaran entradas y salidas mediante vectores de 4 y 7 bits, a los propios vectores se les asigna nombre tal como es “E” y “DISPLAY”

```
11 ARCHITECTURE ADeco OF EDeco IS
12 BEGIN
13
14     PROCESS( E )
15     BEGIN
16         IF( E = "0000")THEN
17             DISPLAY<="0000001";      -- 0
18            ELSIF( E = "0001") THEN
19             DISPLAY<="1001111";      -- 1
20            ELSIF( E = "0010") THEN
21             DISPLAY<="0010010";      -- 2
22            ELSIF( E = "0011") THEN
23             DISPLAY<="0000110";      -- 3
24            ELSIF( E = "0100") THEN
25             DISPLAY<="1001100";      -- 4
26            ELSIF( E = "0101") THEN
27             DISPLAY<="0100100";      --5
28            ELSIF( E = "0110") THEN
```

figura 7 código proceso 1

Al igual que en una ocasión pasada para determinar o el proceso a realizar se utiliza la palabra reservada “ARCHITECTURE” y se declaran los nombres para el inicio y final de la asignación, siendo así que la estructura para realizar las operaciones es mediante un if else, en otro caso se podría ver como un switch ya que para cada caso predeterminado se tendrá una solución y en caso de no tenerla no se realizara ningún proceso y terminara con la sentencia if regresando al inicio del programa

```
29             DISPLAY<="0100000";      --6
30            ELSIF( E = "0111") THEN
31             DISPLAY<="0001110";      --7
32            ELSIF( E = "1000") THEN
33             DISPLAY<="0000000";      --8
34            ELSIF( E = "1001") THEN
35             DISPLAY<="0000100";      --9
36            ELSE
37             DISPLAY<="-------";
38            END IF;
39     END PROCESS;
40
41 END ADeco;
```

figura 6 código proceso 2

Pantallas de prueba

Estas primeras 5 figuras son en referencia a los números dados en el display de tal manera que la secuencia dada corresponde al número en BCD de tal manera que “0000” daría como resultado “0000001” viéndose en el display como un 0.

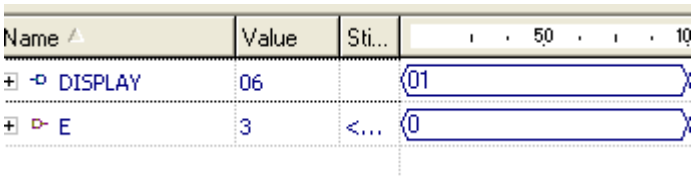


figura 7 número 0

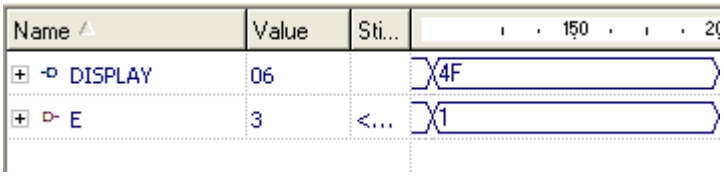


figura 10 número 1

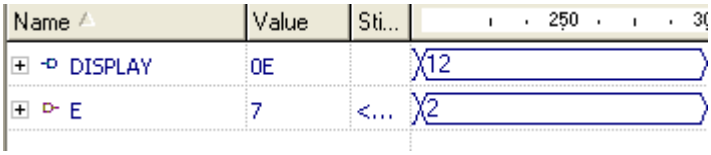


figura 11 número 2

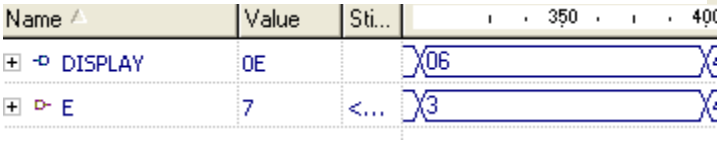


figura 8 número 3

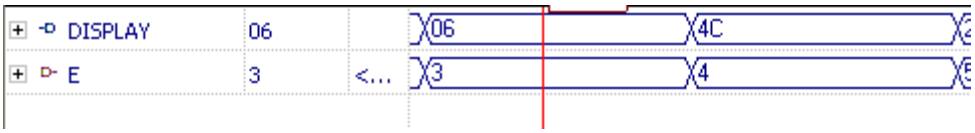


figura 9 número4

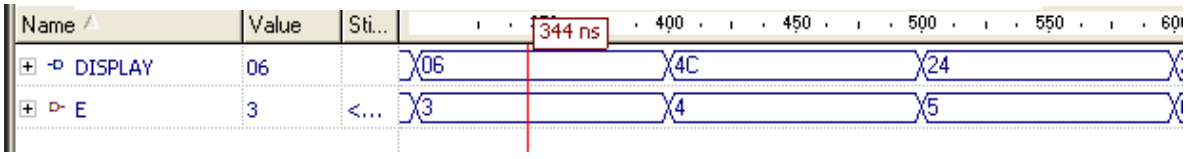


figura 10 número 5

A continuación, se enumerarán las 5 simulaciones del número de boleta

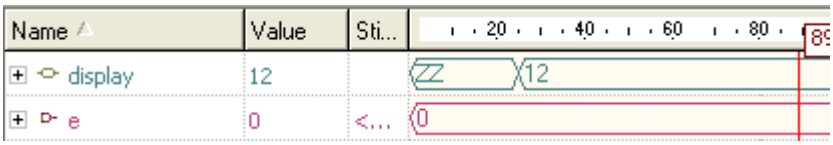


figura 11 secuencia 0 número 2

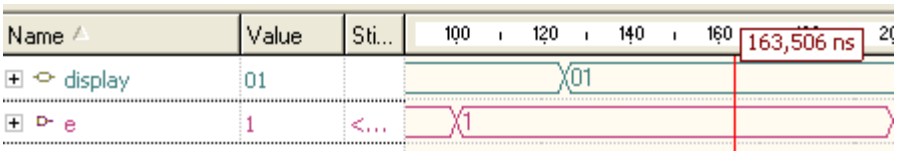


figura 12 secuencia 1 número 0

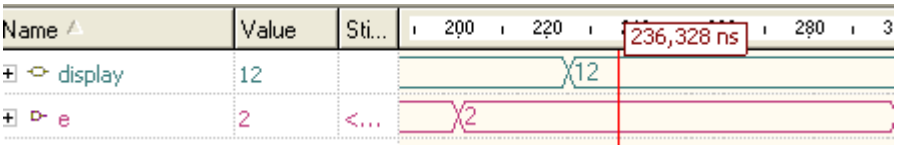


figura 13 secuencia 2 número 2

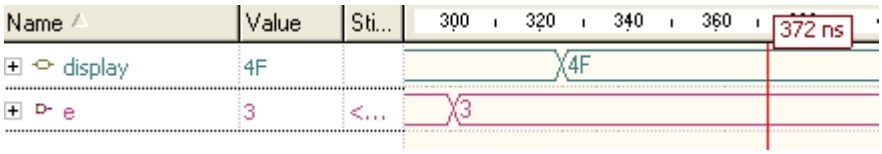


figura 14 secuencia 3 número 1

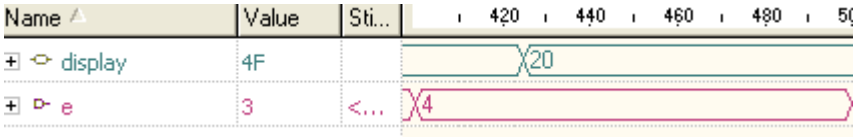


figura 15 secuencia 4 número 6

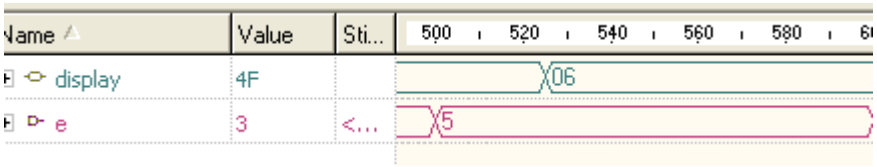


figura 16 secuencia 5 número 3

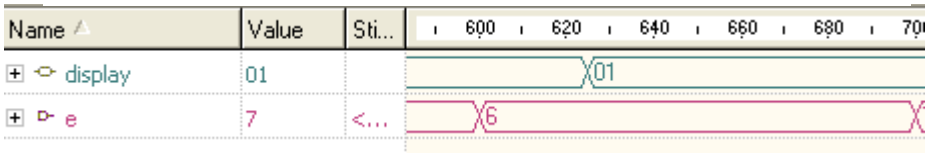


figura 17 secuencia 6 número 0

Como se puede observar en las figuras el número de la boleta se va generando digito por digito de igual manera se muestra en la salida del decodificador, se pueden ingresar más de 9 secuencias para generar un mensaje siendo de esa manera que en cada secuencia la salida tenga como objetivo el mostrar un digito del mensaje

Conclusiones

Mediante la realización de la practica se observo el funcionamiento de un decodificador de binario a BCD siendo de esa manera que se comprendió la utilización de este para el uso de un display como salida del circuito, todo esto en el circuito integrado gal teniendo así una mejor implementación del circuito en este caso