INSTITUTO POLITÉCNICO NACIONAL

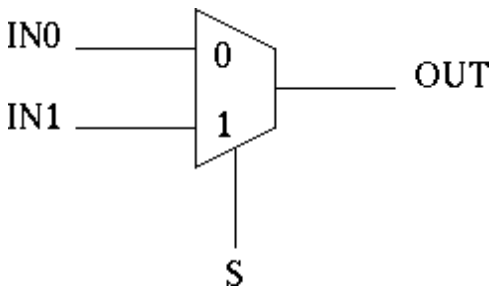
ESCUELA SUPERIOR DE CÓMPUTO

DEPARTAMENTO DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

PRACTICA No. 8

Multiplexor

Nombre: Silva Hernandez Noe Jasiel
Grupo: 2CV1
Materia: Fundamento de Diseño Digital



1) Objetivo general.

Al terminar la sesión, los integrantes del equipo diseñaran e implementaran un circuito multiplexor complementado con un decodificador.

2) Material empleado.

- ✓ 1 Circuito Integrado GAL22V10.
- ✓ 8 Resistores de 220Ω.
- ✓ 8 Resistores de 1KΩ.
- ✓ 1 Dip switch de 8.
- ✓ Alambre telefónico.
- ✓ 1 Tablilla de Prueba (Protoboard).
- ✓ Pinzas de punta.
- ✓ Pinzas de corte.
- ✓ Cables Banana-Caimán (para alimentar el circuito).

3) Equipo empleado.

- ✓ Multímetro.
- ✓ Fuente de Alimentación de 5 Volts.
- ✓ Programador Universal.

4) Desarrollo Experimental y Actividades.

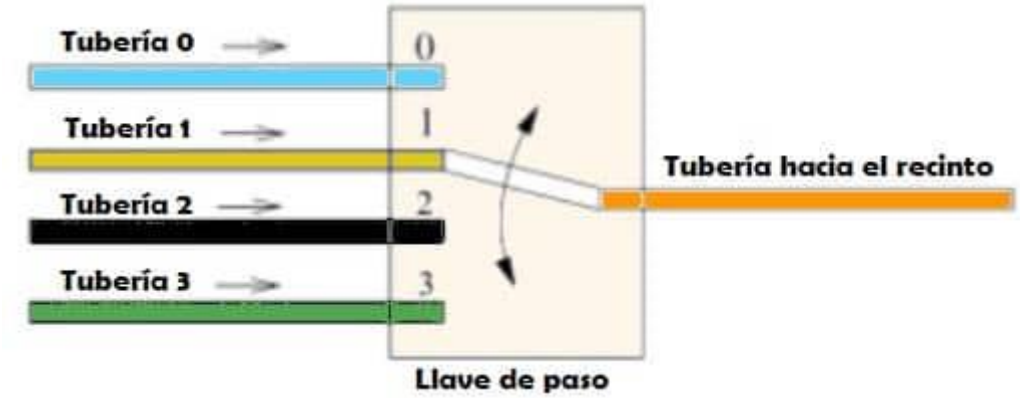
4.1.- Multiplexor de 8 a 1.

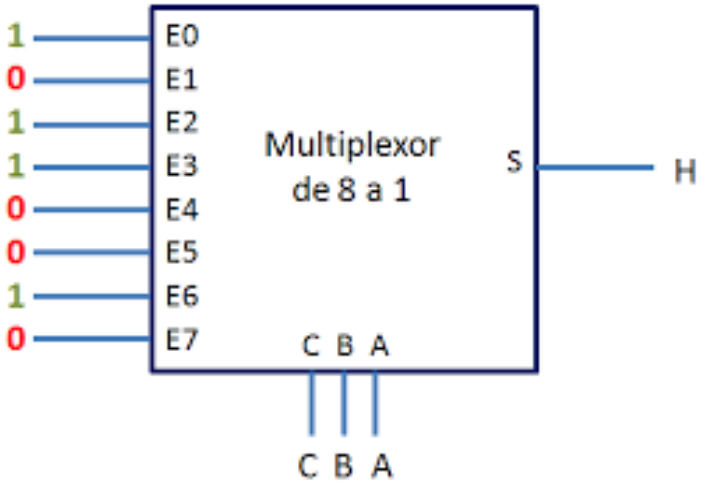
Diseñe un multiplexor 8 a 1, cuya salida va a ser la entrada a un decodificador, para al final conectarse a un display de 7 segmentos, como se muestra en la figura siguiente:

DESARROLLO

Para la realización de la practica se tuvo que investigar el funcionamiento de un multiplexor de 8 a 1… despues de esto se procedio a investigar un poco de codigo relacionado a vhdl para poder obtener una referencia… una vez echo el paso anterior se empezarona definir las entradas y salidas de datos que necesitaremos… además de que se definio las letras mas repetidas del codigo, como: i, p, o ,n ,c, l, e. y unas que se definieron aparte como la u… y todo esto para poder tener mas control en el programa.

Una vez terminando el PORT, en la arquitectura se definio el camino que seguirá el codigo.. ahí fue donde se definio las señales que contineen las letras mas comunes y donde se inicio el process para poder interactuar con mis entradas y poder hacer el if para verificar en que estado se encuentra set y conforme a esto se escoge la entrada que se llevasra al display posteriormente este se lleva a comprobar a nuestra herramienta Active Dhl Sim





Multiplexor

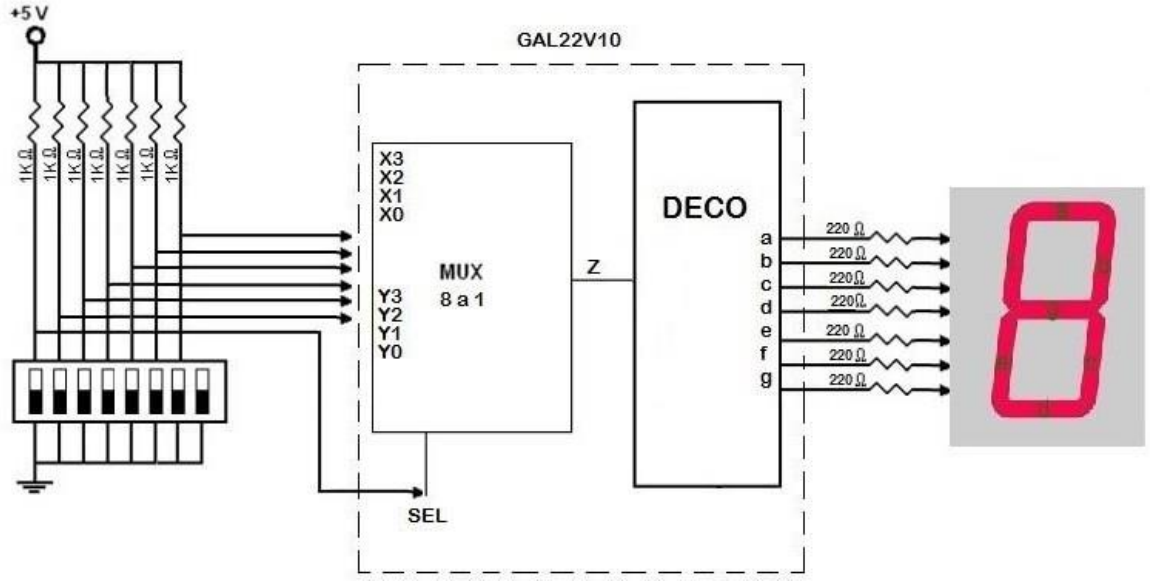


Figura. Desarrollo del Multiplexor 8 a 1.

4.2.- Implemente su solución en VHDL y coloque su informe de pines RPT y su código.

CODIGO

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY EDeco IS
    PORT(

        E: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
        set: in std_logic_vector (1 downto 0);
        DISPLAY: OUT std_logic_VECTOR(6 DOWNTO 0)
    );

END EDeco;

ARCHITECTURE ADeco OF EDeco IS
    SIGNAL i : std_logic_vector(6 downto 0)bus := "1001111";
    SIGNAL n : bit_vector(6 downto 0)bus := B"1101010";
    SIGNAL s : bit_vector(6 downto 0)bus := B"0100100";
    SIGNAL t : bit_vector(6 downto 0)bus := B"1101000";
    SIGNAL o : bit_vector(6 downto 0)bus := B"0000001";
    SIGNAL p : bit_vector(6 downto 0)bus := B"0011000";
    SIGNAL ee : bit_vector(6 downto 0)bus := B"0110000";
    SIGNAL c : bit_vector(6 downto 0)bus := B"0110001";

BEGIN
```

Multiplexor

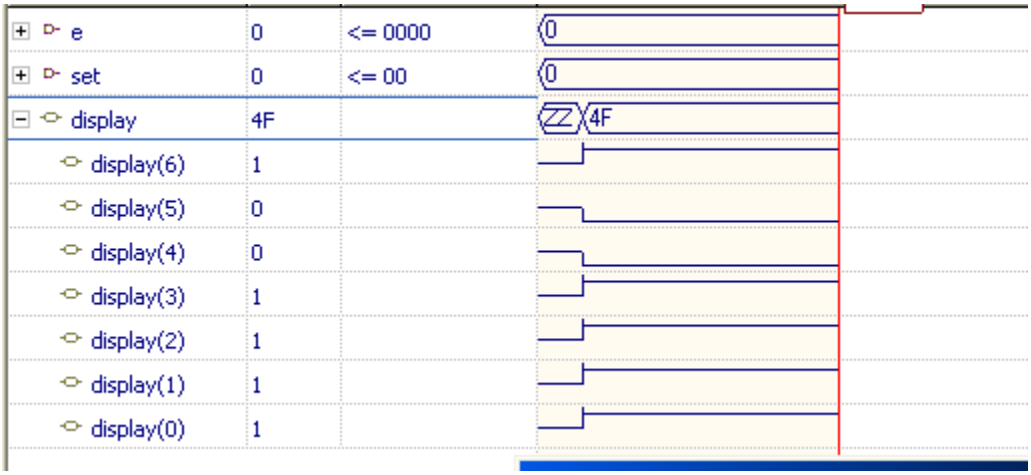
```
24 BEGIN
25
26     PROCESS( E, set )
27     BEGIN
28         IF ((set = "00") and (E = "0000"))THEN
29             DISPLAY <= std_logic_vector(i);   --I
30         ELSIF (set ="00") and (E = "0001")THEN
31             DISPLAY <= std_logic_vector(n);   --N
32         ELSIF(set ="00") and (E = "0010")THEN
33             DISPLAY <= std_logic_vector(s);   --S
34         ELSIF (set ="00") and (E = "0011")THEN
35             DISPLAY <= std_logic_vector(t);   --t
36         ELSIF (set ="00") and (E = "0100")THEN
37             DISPLAY <= std_logic_vector(i);   --I
38         ELSIF (set ="00") and (E = "0101")THEN
39             DISPLAY <= std_logic_vector(t);   --t
40         ELSIF (set ="00") and (E = "0110")THEN
41             DISPLAY <= "1100011";    --u
42         ELSIF (set ="00") and (E = "0111")THEN
43             DISPLAY <= std_logic_vector(t);   --t
44         ELSIF (set ="00") and (E = "1000")THEN
45             DISPLAY <= std_logic_vector(o);   --O
46
47         ELSIF (set ="01") and (E = "0101")THEN
```

```
48             DISPLAY <= std_logic_vector(p);   --p
49         ELSIF (set ="01") and (E = "0110")THEN
50             DISPLAY <= std_logic_vector(o);   --O
51         ELSIF (set ="01") and (E = "0111")THEN
52             DISPLAY <= std_logic_vector(i);   --L
53         ELSIF (set ="01") and (E = "1000")THEN
54             DISPLAY <= std_logic_vector(i);   --I
55         ELSIF (set ="01") and (E = "1001")THEN
56             DISPLAY <= std_logic_vector(t);   --t
57         ELSIF (set ="01") and (E = "1010")THEN
58             DISPLAY <= std_logic_vector(ee);   --E
59         ELSIF (set ="01") and (E = "1011")THEN
60             DISPLAY <= std_logic_vector(c);   --C
61         ELSIF (set ="01") and (E = "1100")THEN
62             DISPLAY <= std_logic_vector(n);   --n
63         ELSIF (set ="01") and (E = "1101")THEN
64             DISPLAY <= std_logic_vector(i);   --I
65         ELSIF (set ="01") and (E = "1110")THEN
66             DISPLAY <= std_logic_vector(c);   --C
67         ELSIF (set ="01") and (E = "1111")THEN
68             DISPLAY <= std_logic_vector(o);   --O
69         ELSE
70             DISPLAY <= "0000000";
71         END IF;
```

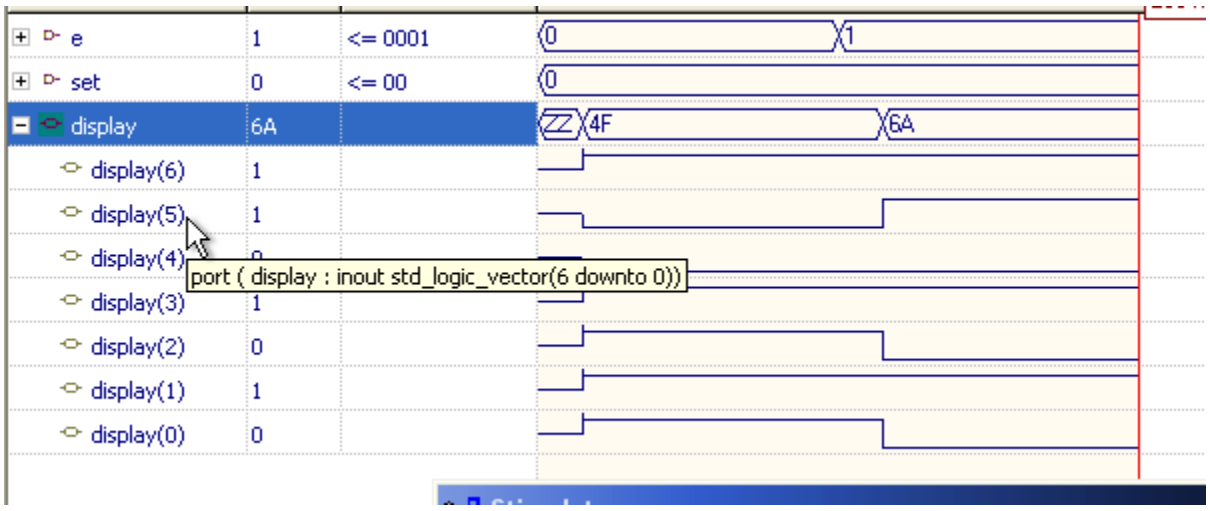
Multiplexor

```
68            DISPLAY <= std_logic_vector(o);    --O
69          ELSE
70            DISPLAY <= "0000000";
71          END IF;
72        END PROCESS;
73
74  END ADeco;
```

I

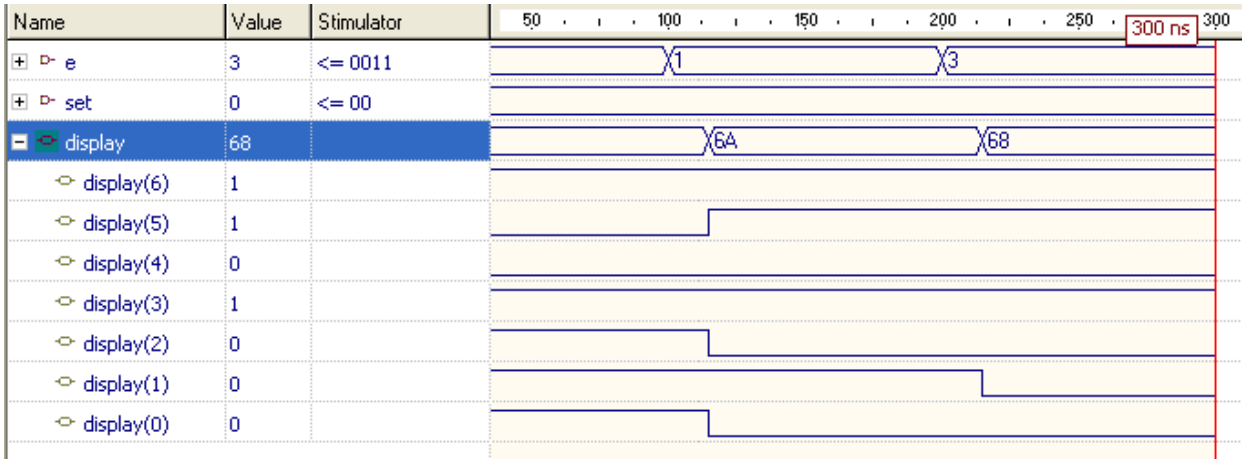


n

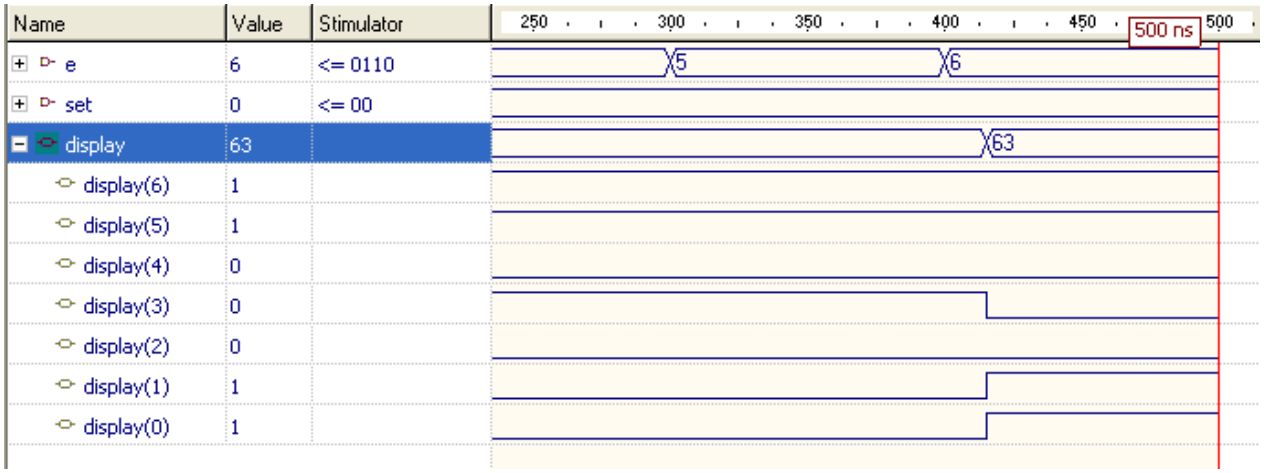


Multiplexor

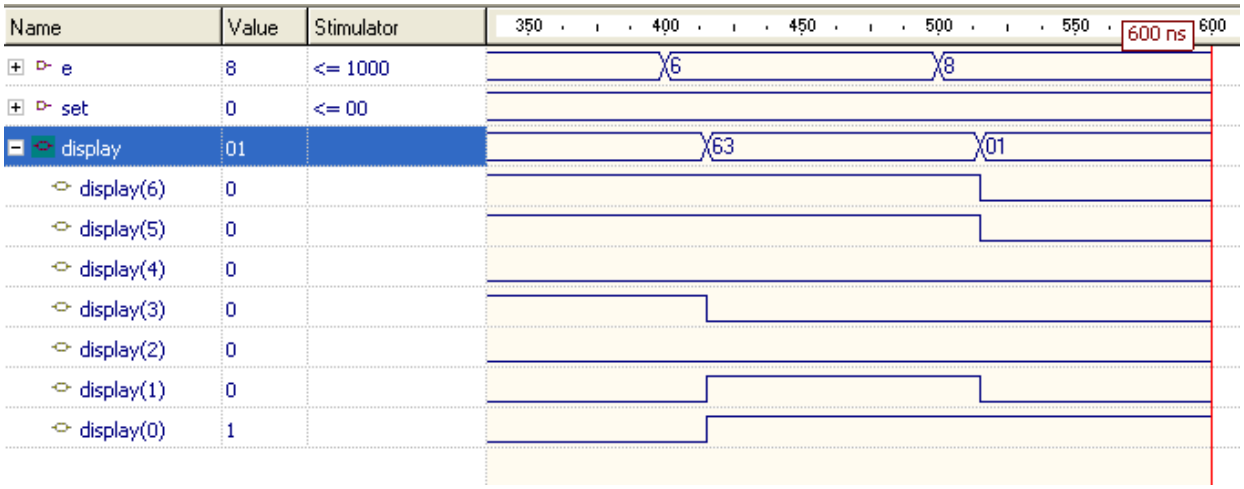
T



U



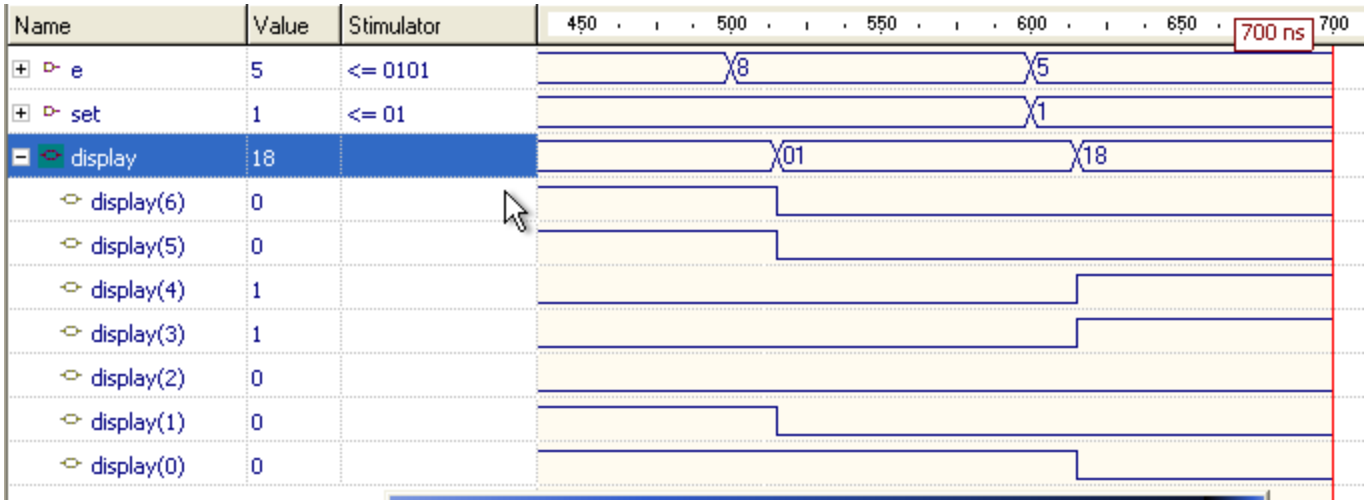
O



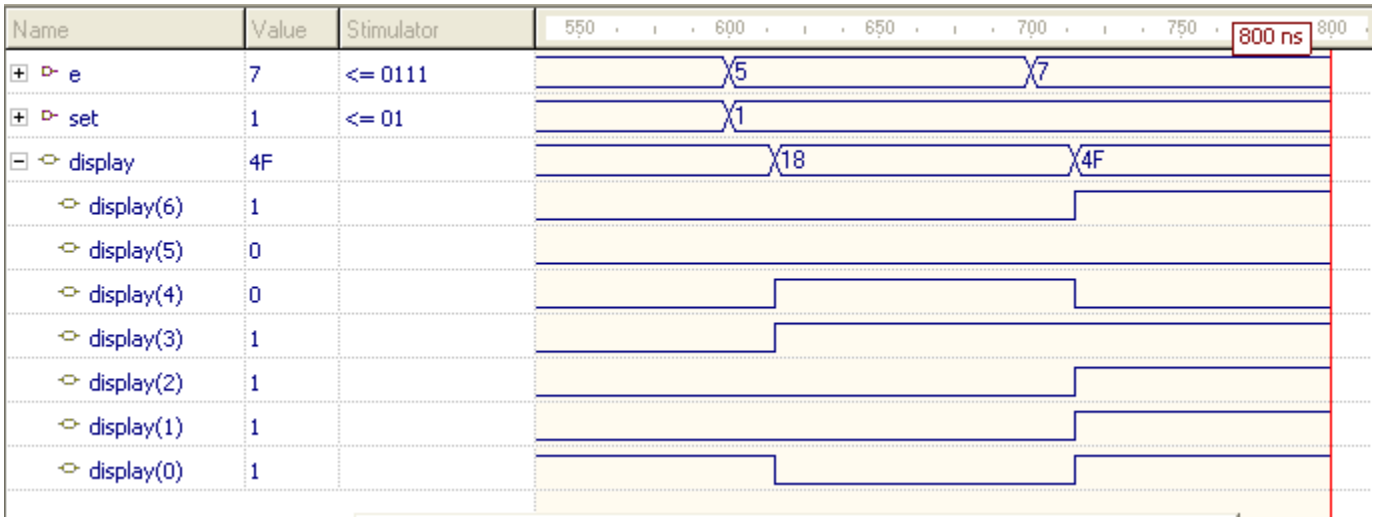
Set 01

p

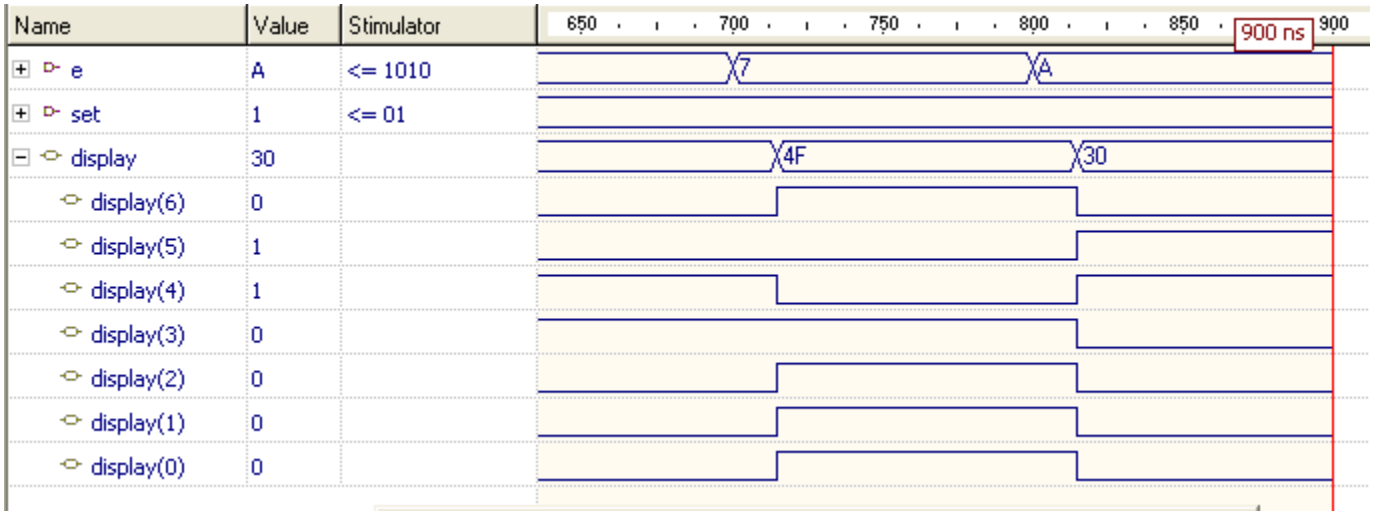
Multiplexor



L

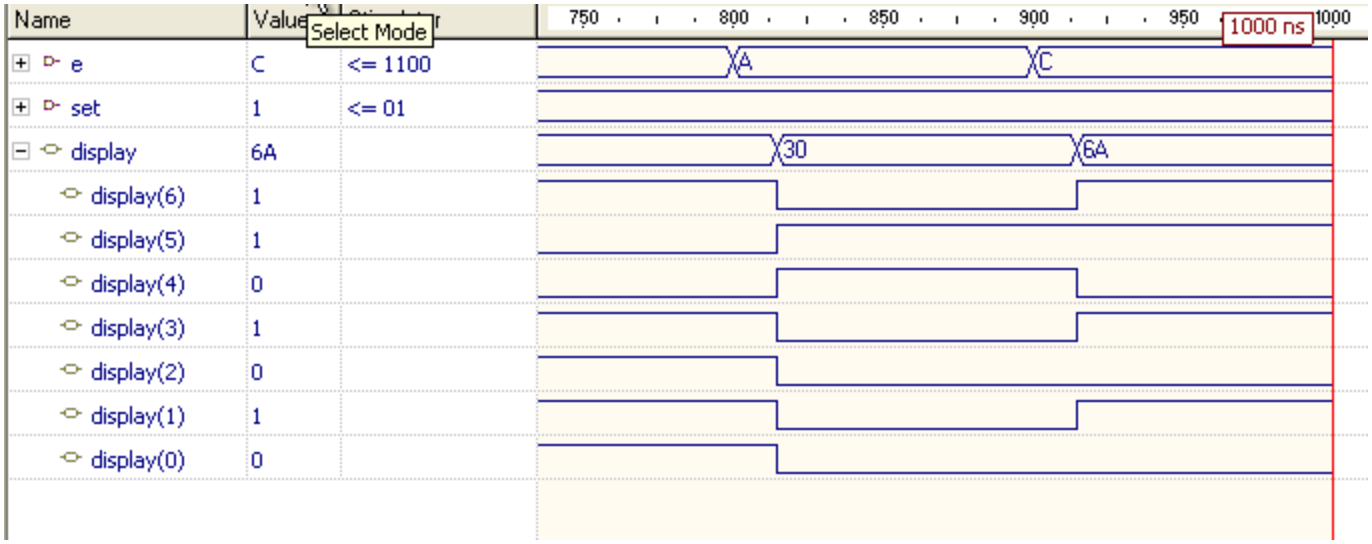


E

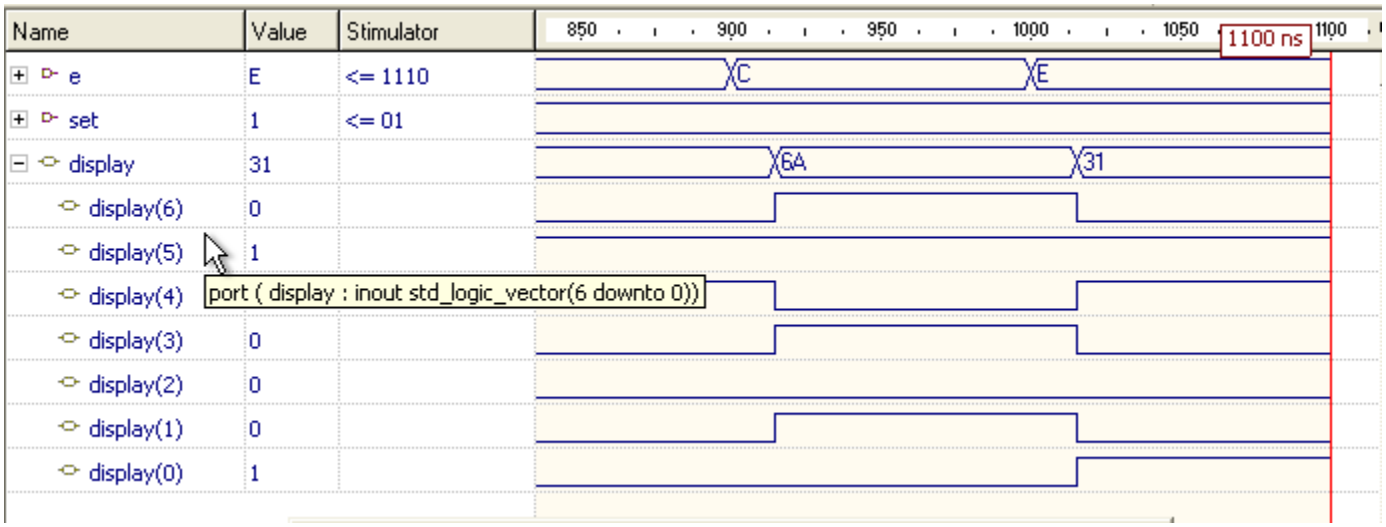


N

Multiplexor



C



5) Conclusiones Individuales.

En esta práctica se entendió claramente el funcionamiento de un multiplexor que es que por cada combinación determinada dada por los selectores se ejecutara una entrada especifica.

Se trabajo con un tema ya dominado que es el mostrar números-letras por medio de un display y estos en su entrada reciben un vector de bits los cuales serán seleccionados por medio del multiplexor. La salida del multiplexor será conectada al display y así se obtiene la magia