Instituto Politécnico Nacional

Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas

Ingeniería Telemática

Diseño Digital

REPORTE DE PRÁCTICA 0

Nombre del Alumno:

Abarca Romero José Angel
Montiel Cruz Ángel Leonardo
Ortiz Alvarado Luis Fernando

Equipo: *2*

Grupo: *1TM5*

Profesora: *David Benjamín Trejo Salazar*

Fecha de Entrega: *17 de marzo del 2021*

Introducción teórica

El lenguaje de descripción hardware VHDL (Very high speed Hardware Description Logic) es un lenguaje orientado a la descripción de hardware pero con muchos elementos heredados de otros lenguajes como C o Pascal. Una vez realizado un programa en VHDL (con extensión VHD) y haberlo compilado con éxito, tendremos un fichero con el mismo nombre y extensión JED, con el cual podremos grabar una PLD (Dispositivo Lógico Programable) con la misma operatividad que el fichero VHD. Entidad o **entity** que es la interfaz del dispositivo con el exterior. Tiene por objeto decir que señales son visibles o accesibles desde el exterior, es decir los puertos o ports del dispositivo.

```
entity prog01 is

    port( a,b,c: in std_logic;
          x: out std_logic;
    end prog01;
```

Arquitectura o arquitecture que es la funcionalidad que realiza el dispositivo, es decir, qué transformaciones se realizarán sobre los datos que entren por los puertos de entrada para producir la salida. Dentro de este apartado es donde se dota de operatividad al circuito. Su estructura general es la siguiente, y debe estar incluida en el mismo fichero de la entidad a la que hace referencia:

```
arquitecture nombre of nombre_entidad is
    begin
        sentencias
    end nombre;
```

Procedimiento

Se solicitó realizar la simplificación de expresiones Booleanas mediante álgebra de Boole. Para comprobar los resultados realizamos las tablas de verdad de dichas expresiones, así como un programa de diseño de hardware que nos permitiera programar un PLD en Proteus para corroborar los resultados obtenidos teóricamente. A continuación, mostramos el ejemplo dado por el profesor.

Expresión: $F(A,B,C,D) = AB + \bar{A}\bar{D} + B\bar{D} + \bar{A}B + C\bar{D}A + \bar{A}D + CD + \bar{A}\bar{B}\bar{C}$
Reducción: $B + \bar{A} + C$

Tabla de verdad:

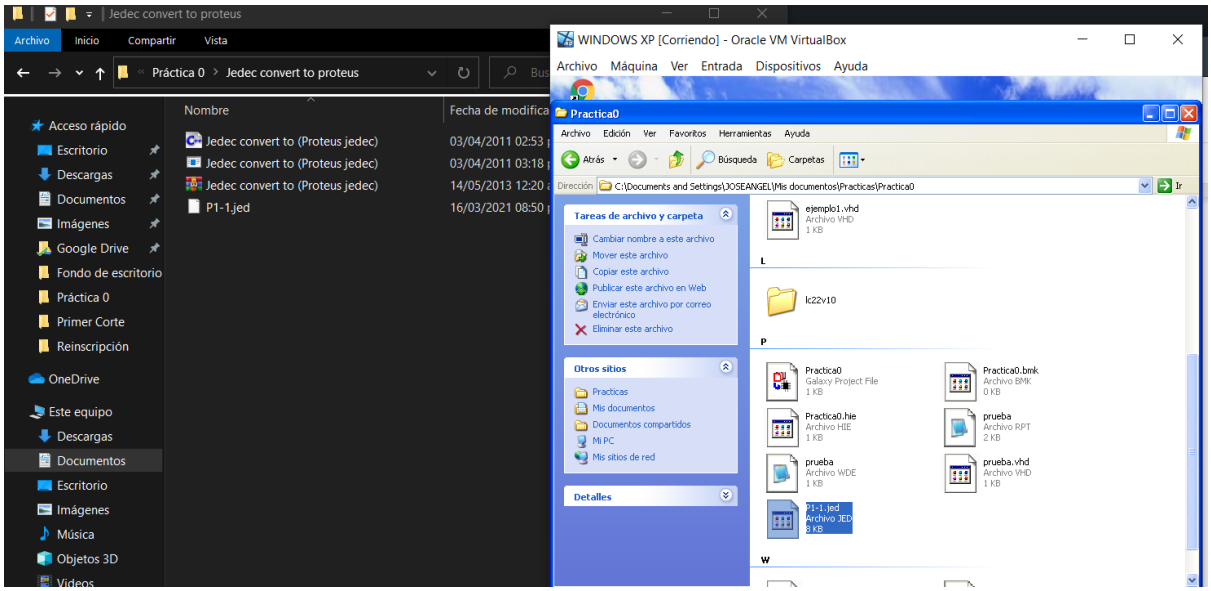
r= AB+A'D'+BD'+A'B
s=CD'A+A'D+A'B'C'

$ABCD$	r	s	$r+s$	$B+\overline{A}+C$
0000	1	1	1	1
0001	0	1	1	1
0010	1	0	1	1
0011	0	1	1	1
0100	1	0	1	1
0101	1	1	1	1
0110	1	0	1	1
0111	1	1	1	1
1000	0	0	0	0
1001	0	0	0	0
1010	0	1	1	1
1011	0	1	1	1
1100	1	0	1	1
1101	1	0	1	1
1110	1	1	1	1
1111	1	1	1	1

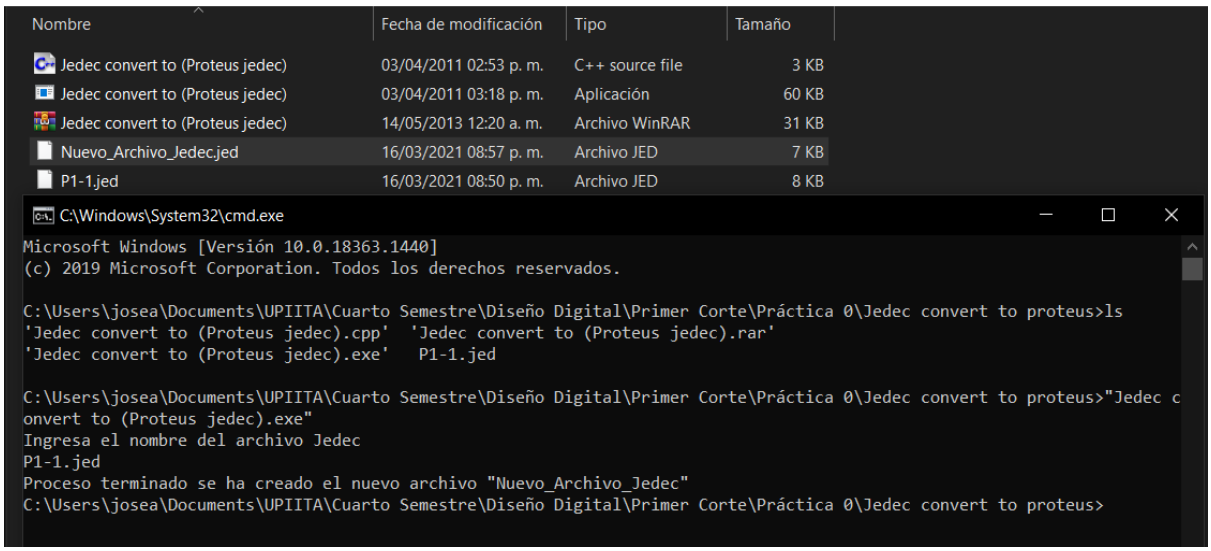
Código en el archivo VHD

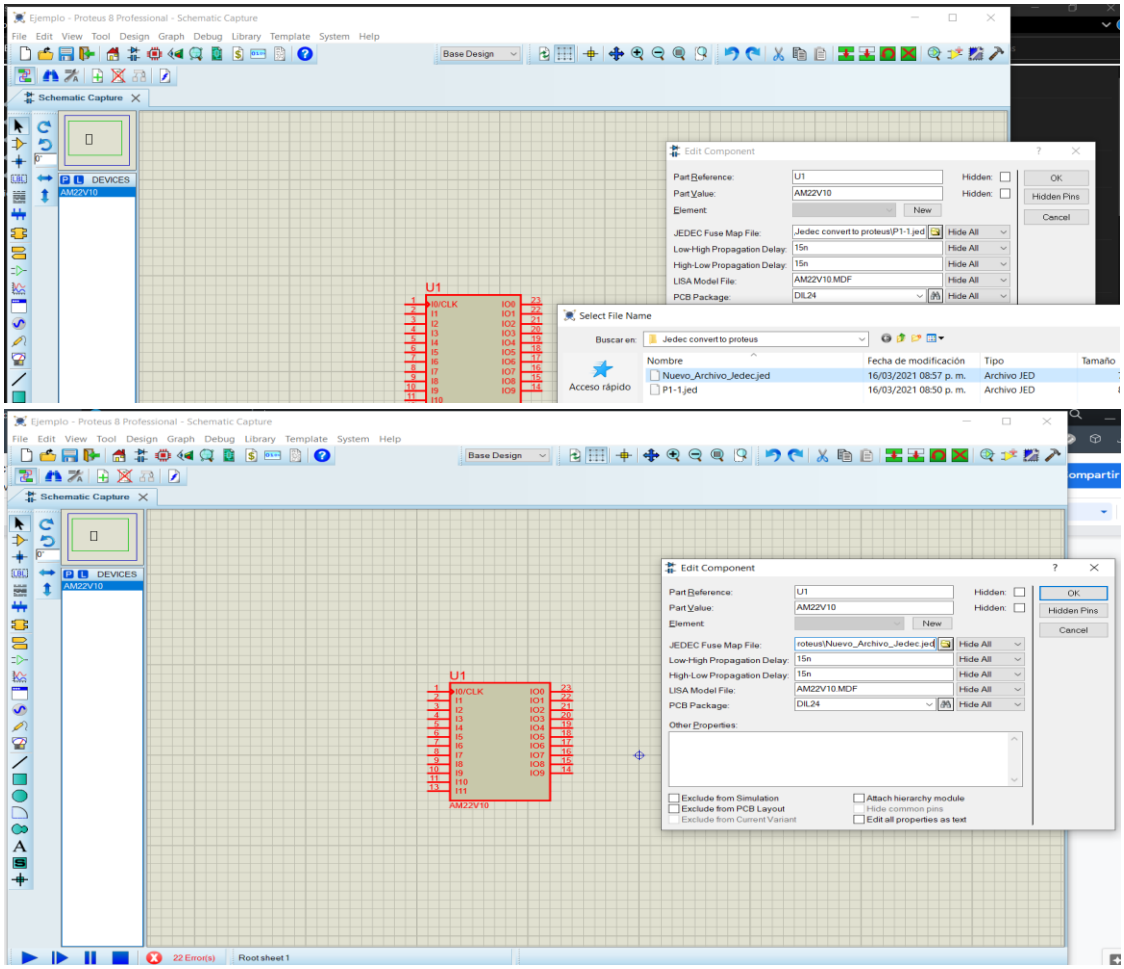
```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  entity prog00 is port(
4  A,B,C,D: in std_logic;
5  x,y: out std_logic);
6
7  end prog00;
8
9  architecture prog of prog00 is
10 begin
11 x<= (A and B) or (not A and not D) or
12    (B and not D) or (not A and B) or
13    (C and not D and A) or (not A and D) or
14    (C and D) or (not A and not B and not C);
15 y<=(B or (not A) or C);
16 end prog; --fin
```

Al compilar el archivo VHD obtuvimos un archivo con extensió JED, el cuál arrastramos desde la máquina virtual hacía nuestro sistema operativo principal



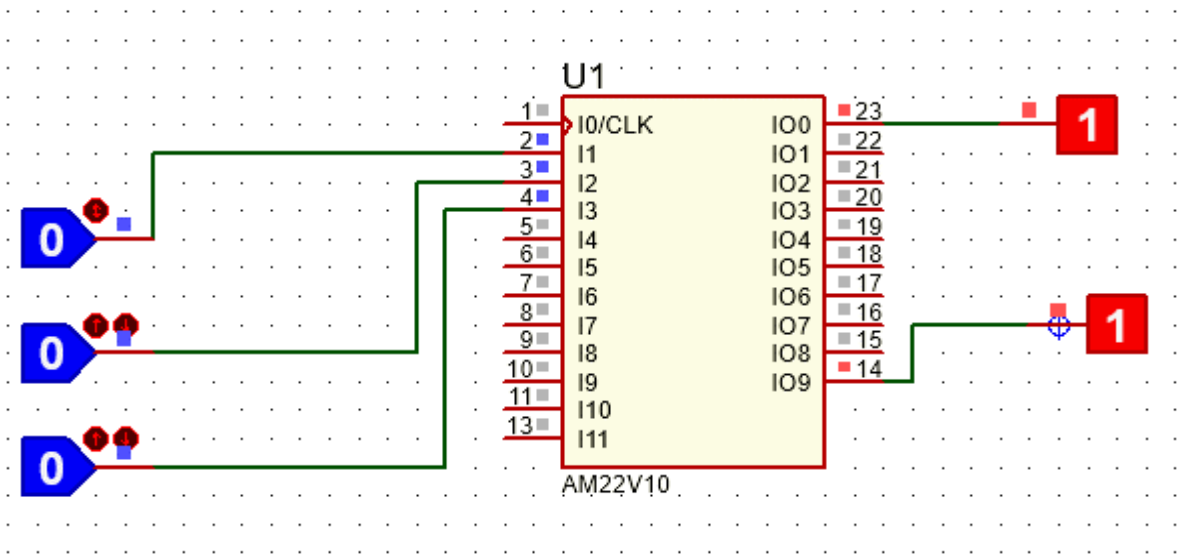
Con la herramienta *“Jedec convert to Proteus”* generamos un archivo con la misma extensión pero que nos permite programar nuestros PLD en Proteus.





Una vez creado el archivo JED, dentro de un proyecto de proteus introducimos este código a un PLD desde la pestaña de propiedades. En este caso utilizamos el integrado AM22V10.

Por último, basándonos en el diagrama dentro del archivo rpt conectamos entradas y salidas lógicas a nuestro PLD para verificar que el funcionamiento de éste es el esperado.



Ejercicio 2

Para los siguientes ejercicios repetimos el mismo proceso realizado para el ejemplo anterior, por lo cual omitiremos el proceso de preparación del archivo JED.

1.

Expresión: $\overline{XYZ} + \bar{Y}(\bar{X}Z + X\bar{Z})$

Reducción:

$\overline{XYZ} + \bar{Y}(\bar{X}Z + X\bar{Z})$
$\overline{XYZ} + \bar{Y}(\bar{X}Z + X\bar{Z})$
$\overline{XYZ} + \bar{Y}(\bar{X}Z + X\bar{Z})$

Tabla de verdad:

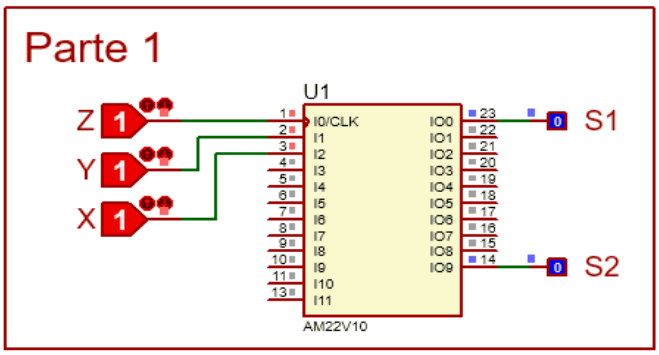
$\boldsymbol{r} = XYZ$
$\underline{\boldsymbol{s} = \bar{Y}(\bar{X}Z + X\bar{Z})}$

XYZ	$\boldsymbol{r} = \overline{\boldsymbol{XYZ}}$	$\boldsymbol{s} = \bar{\boldsymbol{Y}}(\bar{\boldsymbol{X}}\boldsymbol{Z} + \boldsymbol{X}\bar{\boldsymbol{Z}})$	$\boldsymbol{r} + \boldsymbol{s}$	$\bar{\boldsymbol{X}} + \bar{\boldsymbol{Y}} + \bar{\boldsymbol{Z}}$
000	1	0	1	1
001	1	1	1	1
010	1	0	1	1
011	1	0	1	1
100	1	1	1	1
101	1	0	1	1
110	1	0	1	1
111	0	0	0	0

Código VHD:

```
 1 library ieee;
 2 use ieee.std_logic_1164.all;
 3
 4 entity prog0 is port (
 5 a,b,c,d: in std_logic;
 6 x,y: out std_logic);
 7 end prog0;
 8
 9 architecture prog of prog0 is
10 begin
11 x <=     not(A and B and C) or
12          (not B and ((not A and C)or(A and not C)));
13
14 y <=     not a or not b or not c;
15 end prog;
```

PLD en Proteus:



2.

Expresión: $\overline{(Z+\bar{X}\bar{Y})}\overline{(Y+W)}$

Reducción:

$= \overline{(Z+\bar{X}\bar{Y})} + \overline{\overline{(Y+W)}}$

$= \bar{Z}(\bar{X}Y) + Y + W$

$= \bar{X}Y\bar{Z} + Y + W$

$= Y(\bar{X}\bar{Z} + 1) + W$

$= Y + W$

Tabla de verdad:

$\boldsymbol{r} = Z + \overline{\bar{X}\bar{Y}}$

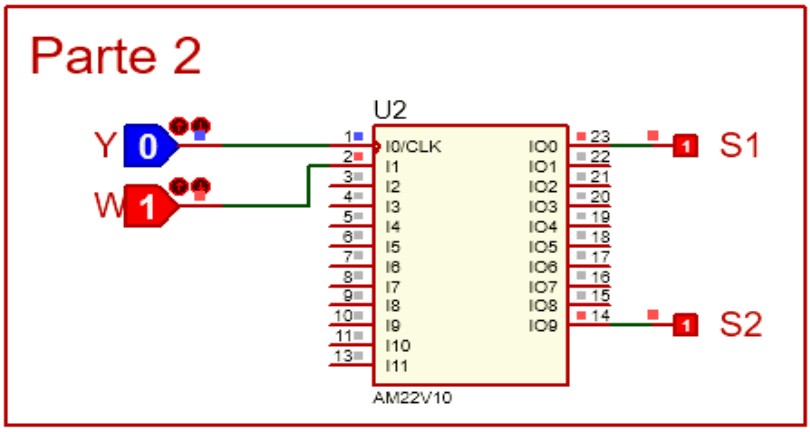
$\boldsymbol{s} = \overline{(Y+W)}$

$WXYZ$	$r = Z + \overline{\bar{X}\bar{Y}}$	$s = \overline{Y+W}$	$\overline{rs}$	Y+W
0000	1	1	0	0
0001	1	1	0	0
0010	0	0	1	1
0011	1	0	1	1
0100	1	1	0	0
0101	1	1	0	0
0110	1	0	1	1
0111	1	0	1	1
1000	1	0	1	1
1001	1	0	1	1
1010	0	0	1	1
1011	1	0	1	1
1100	1	0	1	1
1101	1	0	1	1
1110	1	0	1	1
1111	1	0	1	1

Código VHD:

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity prog0 is port (
5  w,x,y,z: in std_logic;
6  a,b: out std_logic);
7  end prog0;
8
9  architecture prog of prog0 is
10 begin
11 a <=     not((Z or not (not X and Y))and
12          (not (Y or W)));
13 b <=     (y or w);
14 end prog;
```

PLD en Proteus:



3.

Expresión: $\overline{\overline{(XY + YZW)}\,\overline{XY}}$

Reducción:

$= \overline{\overline{(X + Y)}\ \overline{(YZW)}\ \overline{(XY)}}$

$= (X + Y)(\bar{X} + \bar{Y})(\bar{W} + \bar{Y} + \bar{Z})$

$= (X\bar{X} + X\bar{Y} + Y\bar{X} + Y\bar{Y})(\bar{W} + \bar{Y} + \bar{Z})$

$= X\bar{Y} + X\bar{Y}\bar{W} + X\bar{Y}\bar{Z} + Y\bar{X}\bar{W} + Y\bar{X}\bar{Z}$

$= X\bar{Y}(\bar{W} + \bar{Z} + 1) + Y\bar{X}\bar{W} + Y\bar{X}\bar{Z}$

$= X\bar{Y} + Y\bar{X}\bar{W} + Y\bar{X}\bar{Z}$

Tabla de verdad:

$\boldsymbol{r} = \overline{\overline{XY} + YZW}$

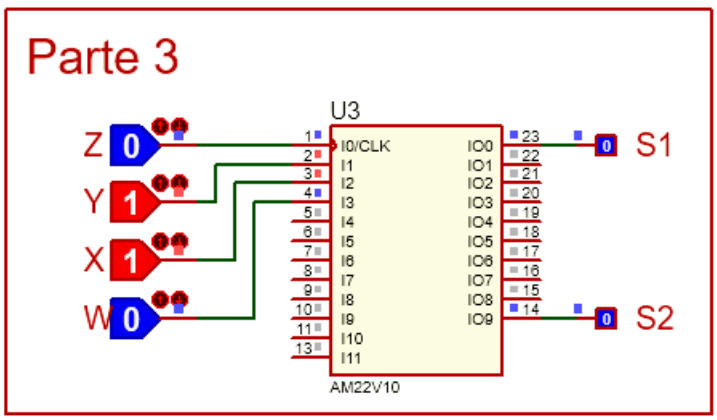
$\boldsymbol{s} = \overline{XY}$

$WXYZ$	$r = \overline{\overline{XY} + YZW}$	$s = \overline{XY}$	rs	$X\bar{Y} + Y\bar{X}\bar{W} + Y\bar{X}\bar{Z}$
0000	0	1	0	0
0001	0	1	0	0
0010	1	1	1	1
0011	1	1	1	1
0100	1	1	1	1
0101	1	1	1	1
0110	1	0	0	0
0111	1	0	0	0
1000	0	1	0	0
1001	0	1	0	0
1010	1	1	1	1
1011	0	1	0	0
1100	1	1	1	1
1101	1	1	1	1
1110	1	0	0	0
1111	0	0	0	0

Código VHD:

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity prog0 is port (
5  w,x,y,z: in std_logic;
6  a,b: out std_logic);
7  end prog0;
8
9  architecture prog of prog0 is
10 begin
11 a <=     not(not(X or Y)or(Y and Z and W))and
12          (not (Y and X));
13 b <=     (x and not y) or (not x and y and not z) or (not x and y and not w);
14 end prog;
```

PLD en Proteus:



Ejercicio 3

En este ejercicio se nos solicitó realizar el lenguaje de descripción de hardware para los siguientes integrados:

- 74LS00
- 74L04
- 74L86
- 74L10
- 74LS21
- 74LS30

Además, también se requirió realizar la exportación de los archivos VHD a Proteus y generar el PLD correspondiente.
A continuación, mostramos el código dentro de los archivos VHD de todos los circuitos.

- 74LS00

```
library ieee;
use ieee.std_logic_1164.all;

entity prog0 is port (
i1, i2, i3, i4, i5, i6, i7, i8: in std_logic;
o1, o2, o3, o4: out std_logic);

attribute pin_numbers of prog0: entity is
" i1:1 i2:2 i3:3 i4:4 i5:5 i6:6 i7:7 i8:8 " &
" o1:23 o2:22 o3:21 o4:20 ";
end prog0;


architecture prog of prog0 is
begin

o1 <= i1 and i2;
o2 <= i3 and i4;
o3 <= i5 and i6;
o4 <= i7 and i8;

end prog;
```

- 74L04

```
library ieee;
use ieee.std_logic_1164.all;

entity prog0 is port (
i1, i2, i3, i4, i5, i6: in std_logic;
o1, o2, o3, o4, o5, o6: out std_logic);

attribute pin_numbers of prog0: entity is
" i1:1 i2:2 i3:3 i4:4 i5:5 i6:6 " &
" o1:23 o2:22 o3:21 o4:20 o5:19 o6:18";
end prog0;


architecture prog of prog0 is
begin

o1 <= not i1;
o2 <= not i2;
o3 <= not i3;
o4 <= not i4;
o5 <= not i5;
o6 <= not i6;
end prog;
```

- 74L86

```
library ieee;
use ieee.std_logic_1164.all;

entity ls86 is port (
i1, i2, i3, i4, i5, i6, i7, i8: in std_logic;
o1, o2, o3, o4: out std_logic);

attribute pin_numbers of ls86: entity is
"i1:1 i2:2 i3:3 i4:4 i5:5 i6:6 i7:7 i8:8 " &
"o1:23 o2:22 o3:21 o4:20";
end ls86;

architecture prog of ls86 is
begin

o1 <= (i1 and not i2) or (not i1 and i2);
o2 <= (i3 and not i4) or (not i3 and i4);
o3 <= (i5 and not i6) or (not i5 and i6);
o4 <= (i7 and not i8) or (not i7 and i8);
end prog;
```

- 74L10

```
library ieee;
use ieee.std_logic_1164.all;

entity ls10 is port (
i1, i2, i3, i4, i5, i6, i7, i8, i9: in std_logic;
o1, o2, o3: out std_logic);

attribute pin_numbers of ls10: entity is
"i1:1 i2:2 i3:3 i4:4 i5:5 i6:6 i7:7 i8:8 i9:9 " &
"o1:23 o2:22 o3:21";
end ls10;

architecture prog of ls10 is
begin

o1 <= not(i1 and i2 and i3);
o2 <= not(i4 and i5 and i6);
o3 <= not(i7 and i8 and i9);
end prog;
```

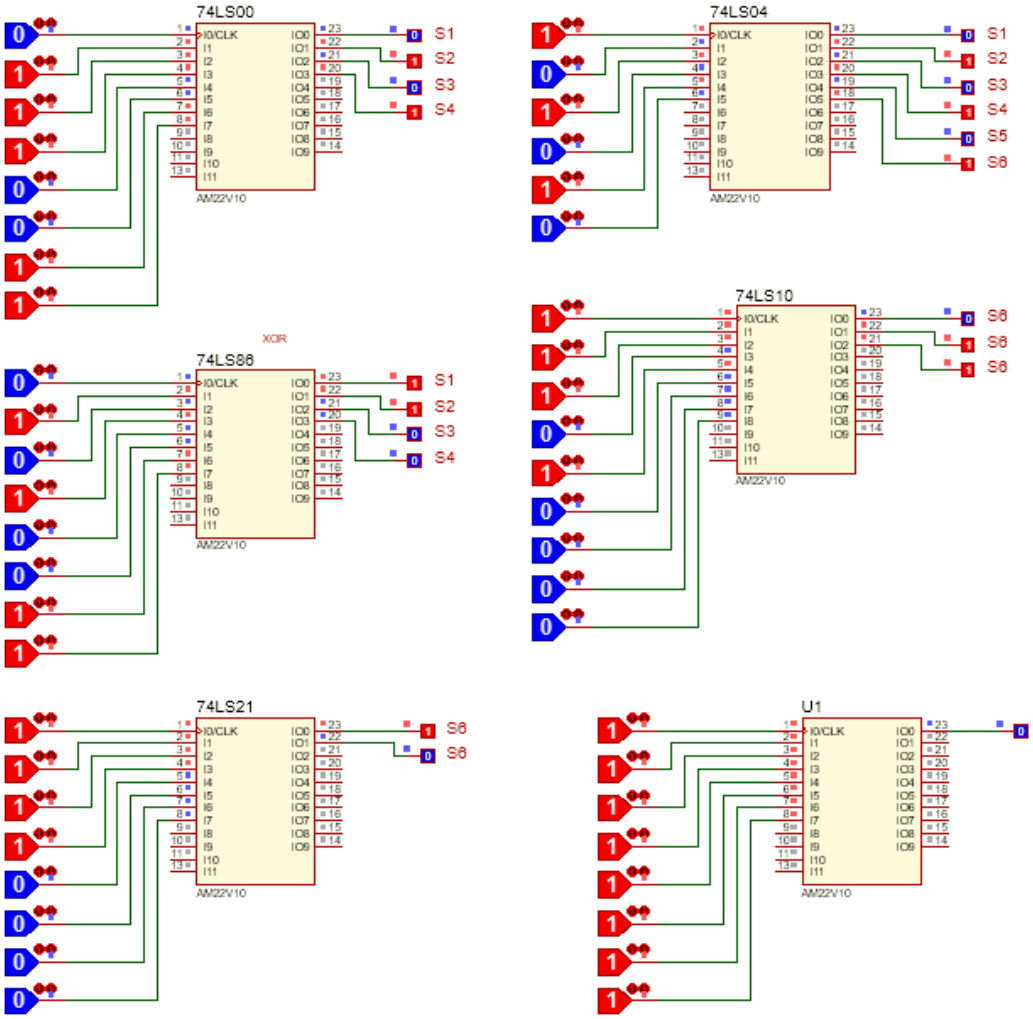
- 74LS21

```
library ieee;
use ieee.std_logic_1164.all;

entity avr is port (

    a,b,c,d,e,f,g,h:    in  std_logic;
    x,y:        out std_logic);

attribute pin_numbers of avr: entity is
"a:1 b:2 c:3 d:4 e:5 f:6 g:7 h:8 " &
"x:23 y:22";
end avr;

architecture prog of avr is
begin

    x <= (a and b and c and d);
    y <= (e and f and g and h);

end prog;
```

- 74ls30

```
library ieee;
use ieee.std_logic_1164.all;

entity avr is port (

    a,b,c,d,e,f,g,h:    in  std_logic;
    y:      out std_logic);

attribute pin_numbers of avr: entity is
"a:1 b:2 c:3 d:4 e:5 f:6 g:7 h:8 " &
"y:23";
end avr;

architecture prog of avr is
begin

    y <= not(a and b and c and d and e and f and g and h);

end prog;
```

De igual forma que en los ejercicios anteriores, generamos archivos con extensión JED y los insertamos a nuestro PLD en Proteus.



Conclusiones

Abarca Romero José Angel:

Con esta práctica pudimos ver de forma práctica y teórica los beneficios de utilizar álgebra de Boole en la reducción de expresiones Booleanas, pues al realizar dichas operaciones pudimos observar una reducción en el número de compuertas lógicas necesarias para obtener el mismo resultado que las expresiones originales nos otorgaban.

Además, las herramientas de diseño digital que utilizamos son realmente versátiles a la hora de crear nuestros propios circuitos lógicos programables, pues nos permiten diseñar nuestro propio hardware en un tiempo considerablemente reducido, además que así evitamos poner en riesgo equipo que de otra forma podría resultar afectado si no se toman las medidas de precaución necesarias.

Montiel Cruz Angel Leonardo:

Con el desarrollo de esta práctica conseguí tener de manera más clara el uso del algebra booleana para la simplificación de expresiones, además de saber implementar de manera adecuada todos los postulados antes visto y sobre todo el teorema de Morgan.

También logre aprender a crear un proyecto, escribí el código, compilarlo, transformarlo y ejecutarlo desde el proteus, por lo que logre entender cómo usar el IDE y todos los pasos posteriores a él, por lo tanto, logramos diseñar hardware con los que pudimos realizar varias pruebas y entender su funcionamiento.

Ortiz Alvarado Luis:

Mediante el uso del álgebra de Boole y las leyes de Morgan se pudo observar como un sistema complejo de compuertas se puede reducir de forma fácil, obteniendo un resultado más sencillo, y logrando comprobar este resultado mediante el uso de tablas de verdad. Y mediante el uso de lenguaje de descripción de hardware se pudo realizar distintos integrados, los cuales ocupan compuertas lógicas para poder hacerlo