Universidad Nacional Autónoma de México

Facultad de Ingeniería

División de Ingeniería Eléctrica

Laboratorio de Diseño Digital Moderno

Práctica 2: Lenguaje de descripción de hardware VHDL

M.I Vicente Flores Olvera

Bautista Pérez Brian Jassiel
Serralde Flores Andrea

14 de junio de 2021

1. Objetivo

El alumno analizará de qué partes esenciales contra un código hecho a traves del lenguaje de hardware VHDL, así también que implica la implemenación de funciones mediante el estilo de flujo de datos.

2. Previo

Obtener las formas mínimas SOP y POS al igual que las formas canónicas de las funciones del combinacional representado por un sistema que tiene cuatro entradas **A; B; C; D** y cuatro salidas, **W; X; Y; Z**. La salida representa un número en código Exceso - 3 cuyo valor es igual al número de unos presentes en la entrada. Por ejemplo, si ABCD = 1101, entonces la salida debe ser WXYZ = 0110.

Obtención de las funciones mediante tablas de verdad

ABCD	xyz
0000	011
0001	100
0010	100
0011	101
0100	100
0101	101
0110	101
0111	110
1000	100
1001	101
1010	101
1011	110
1100	101
1101	110
1110	110
1111	111

Cuadro 1: Tabla de verdad de las funciones.

Finalmente, las funciones nos quedan de la siguiente forma:

$x = a + b + c + d$

$y = ab(d + c) + cd(b + a) + \overline{a}\overline{b}\overline{c}\overline{d}$

$z = \overline{a \oplus b \oplus c \oplus d}$

- ¿Qué son las bibliotecas, la entidad y la arquitecura en una estructura desrita en VHDL?
  R. Las **bibliotecas**, dentro de VHDL, son colecciones de piezas de código usualmente empleadas. Esto permite poder reusar esas piezas o compartirlas con otros diseños.
  La **entidad** Es la zona de declaración de los vectores del módulo, generalmente entradas y salidas
  La **arquitectura** Es la zona en la cual se describe de manera detallada la estructura interna o comportamiento del módulo.

- Investiga cómo se realiza la asignación de una función o de un valor en VHDL.
  R. La asignación de funciones se realiza mediante el operador <=. A la izquierda se escribe el nombre o identificador de la función o variable, mientras que a la derecha se escribe lo que se le va a asignar.

- ¿Qué significa el término concurrente dentro del lenguaje VHDL?
  R. Es la implementación de procesos dentro de un archivo de programación VHDL. Su propósito es definir los límites de un dominio secuencial.

3. Desarrollo de la práctica

3.1. Práctica 2A

Programar las funciones mostradas, en un formato SOP y POS mínimo (reducido) en forma de flujo de datos, para su posterior simulación dentro de la plataforma Quartus II.

$$f(xyzt) = (x + \bar{x}y + \overline{yt})(x + (\overline{xy})z) \tag{1}$$

Función con formato de Producto de Sumas 1.

$$f(xyzt) = (x + z) \tag{2}$$

Función reducida 2.

$$f(uvtw) = vt(u + t\bar{w})(u + \bar{w}) + wu(v + t) \tag{3}$$

Función con formato de Suma de Productos 3.

$$f(uvtw) = vtu + vt\bar{w} + wut + wuv \tag{4}$$

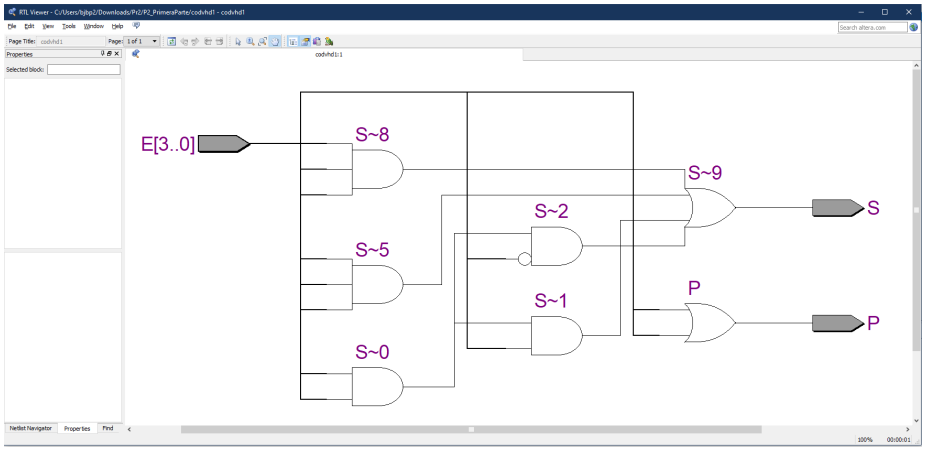
Función reducida 4.

3.1.1. Implementación en Quartus II

```
--Entrada basica
library ieee;
use ieee.std_logic_1164.all;

--Desarrollo de la entidad
entity codvhdl is
   port(E:in bit_vector(3 downto 0);
        S, P: out bit);
end;

architecture behavioural of codvhdl is
begin

P<=(E(3) or E(1));
S<=((E(2) and E(1) and E(3)) or (E(2) and E(1) and not E(0)) or
          (E(0) and E(1) and E(3)) or (E(0) and E(2) and E(3)));
end;
```

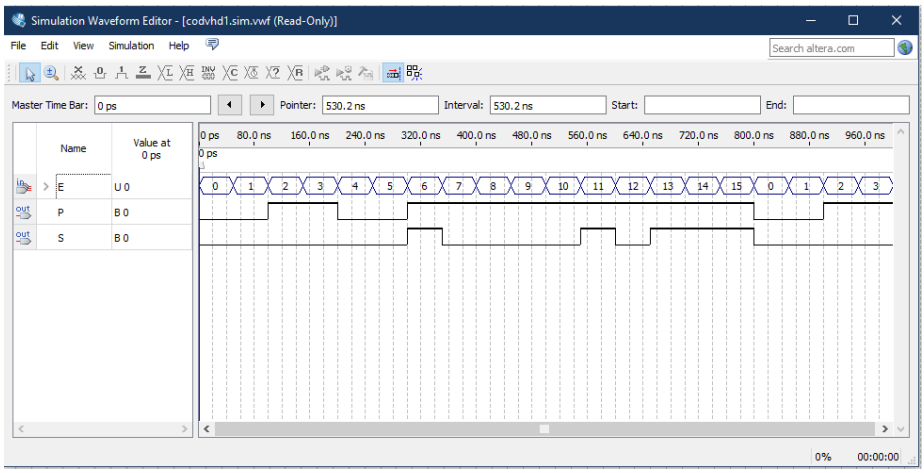
(c) Implementación de las funciones en Quartus II

En este código VHDL se importa la biblioteca **ieee**, después, dentro de una entidad procedemos a declarar un vector **E**, el cual va a representar nuestras variables de entrada en donde el bit más significativo en cada función será **x** y **u**, respectivamente. Dentro de esta entidad, declaramos nuestras variables de salida, las cuales serán nuestras funciones y representadas en SOP y POS, dichas funciones están representadas por las variables **P** y **S**, respectivamente. Finalmente, dentro de una arquitectura procederemos a declarar nuestras funciones, en las cuales, debemos de ser muy cuidadosos con la jerarquía y con los paréntesis para que no tengamos errores durante la compilación de nuestro proyecto.



(d) Esquema de las funciones

En el esquema de arriba se representa el código recién escrito en forma de un circuito lógico, en este caso vemos que dicho circuito está compuesto de compuertas **and** y **or**



(e) Simulación realizada

3.2. Práctica 2B

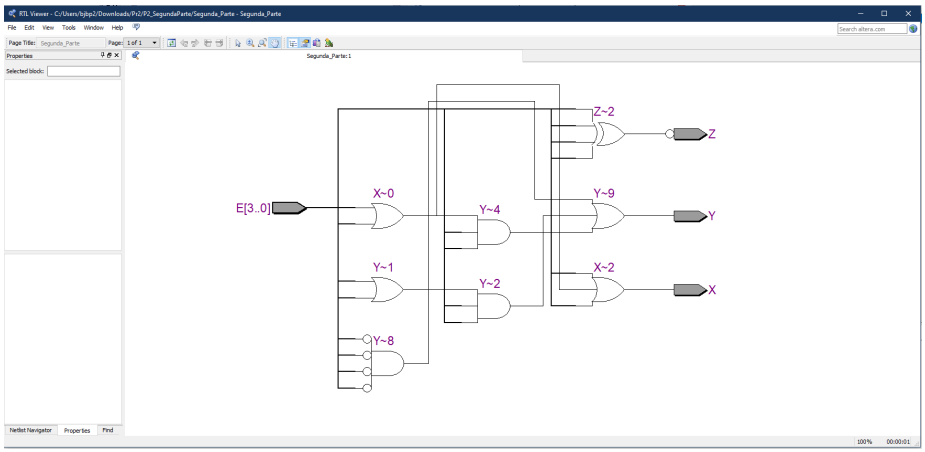
Programar las funciones expresadas en formato POS Y SOP canónicas y mínimas del previo en forma de flujo de datos, para su posterior simulación dentro de la plataforma Quartus II.

3.2.1. Implementación en Quartus II

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity Segunda_Parte is
5     port(E:in bit_vector(3 downto 0);
6          X, Y, Z: out bit);
7  end;
8
9  architecture behavioural of Segunda_Parte is
10 begin
11
12 --a = E(3)
13 --b = E(2)
14 --c = E(1)
15 --d = E(0)
16
17 X<= (E(3) or E(2) or E(1) or E(0));
18 Y<= ((E(3)and E(2) and (E(0) or E(1))) or (E(1) and E(0) and (E(2) or E(3))) or ((not E(3))and(not E(2)) and (not E(1)) and (not E(0))));
19 Z<= not(E(3) xor E(2) xor E(1) xor E(0));
20
21 end;
```

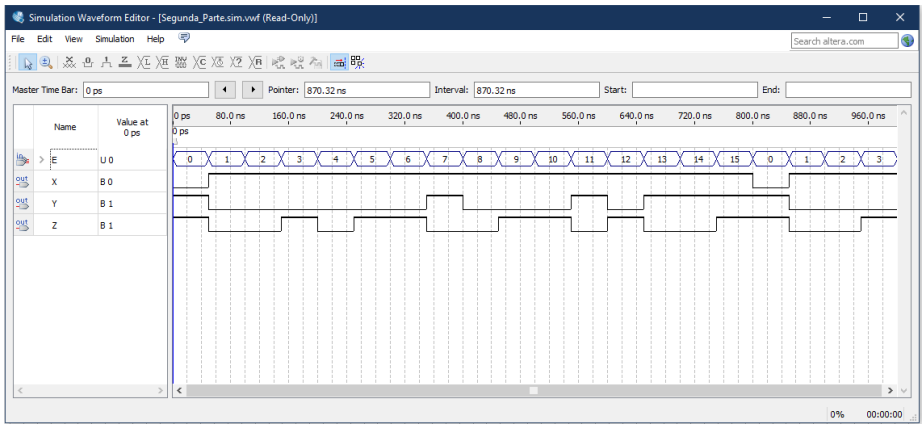
(f) Implementación de las funciones en Quartus II

Esta implementación tiene una forma similar, que al primer ejercicio realizado, la diferencia está en que declaramos tres variables de salida, **x, y, z** en la arquitectura en donde declaramos nuestras funciones. El proceso de compilación y ejecución del programa es el mismo.



(g) Esquema de las funciones

En este diagrama se ve representado nuestro código en un circuito lógico compuesto por compuertas **and, or** y **xor**.



(h) Simulación realizada

En esta simulación verificamos que la tabla de verdad realizada para obtener las funciones es igual a la que obtuvimos en Quartus, por lo que se deduce que realizarmos la obtención y codificación de las variables de manera correcta.

4. Conclusiones

Mediante la aplicacón de las leyes del álgebra de Boole se pudo obtener en primer lugar la representación canónica de las funciones y en segundo, las formas reducidas en POS y SOP. Esto nos fue de gran ayuda ya que de más sencillo escribirlas en código VHDL en el simulador Quartus II.

Al observar que los resultados de la simulación coinciden con las tablas de verdad realizadas para la representación de las funciones se concluye que el objetivo principal de esta práctica se cumplió de manera satisfactoria.
El lenguaje VHDL fue muy importante para la realización de esta práctica y será una herramienta muy útil para las próximas y también para el desarrollo de programas para las tarjetas programables, no sólo en esta materia, sino en las de los siguientes semestres.