UNIVERSIDAD POLITECNICA DE LA ZONA METROPOLITANA DE GUADALAJARA



ING MECATRONICA

Enciso Guerrero Benjamin Salvador

Sistemas Embebidos

Carlos Enrique Moran Garabito.

VHDL

8-B.

Entre los años 1984 y 1987, el IEEE y el Departamento de Defensa de los Estados Unidos (DoD) patrocinan el desarrollo de un Lenguaje llamado VHDL. Su nombre viene de VHSIC HDL, o sea Lenguaje de Descripción de hardware para circuitos integrados de muy alta velocidad.

Considerando que un lenguaje de descripción de hardware es una herramienta formal que permite describir la estructura y comportamiento de un sistema para lograr una adecuada especificación, documentación y simulación del mismo antes de su realización real; para su implementación se establecieron ciertas características fundamentales, que aún hoy siguen siendo válidas, y son:

-Cada elemento de diseño tiene una interfaz única y perfectamente definida, que permite conectarla a otros elementos.

-Cada elemento tiene un comportamiento preciso y unívocamente definido, que permiten su posterior simulación.

-La especificación de comportamiento que permite definir la operatividad puede realizarse a través de un algoritmo ó de una estructura de hardware real.

-Los diseños mantienen una estructura jerárquica, que permite descomponerlo adecuadamente.

-Las características concurrentes, temporizadas y de sincronismo (por ej. reloj) pueden ser modeladas adecuadamente.

-Se puede simular cualquier operación lógica y de temporización.

Se establece así una herramienta que además tiene amplias características de modelado y documentación. De esta forma cualquier circuito digital se puede especificar y simular adecuadamente.

Luego del desarrollo de este lenguaje aparecieron las herramientas adecuadas de síntesis que completan el panorama de diseño de un sistema digital. Así de esta forma se puede decir que si utiliza VHDL se puede diseñar, simular y sintetizar cualquier sistema digital, desde el combinacional más simple hasta la estructura secuencial más compleja. Las nuevas versiones de HDL permiten también el desarrollo de circuitos analógicos.

Como se ve en el diagrama de flujo de diseño planteado, aparecen muchos avances y retrocesos, hasta lograr el diseño adecuado. Lo importante es no dejarse llevar solo por el uso de herramientas automáticas, sino poner en cada paso la experiencia adquirida en los diseños anteriores.

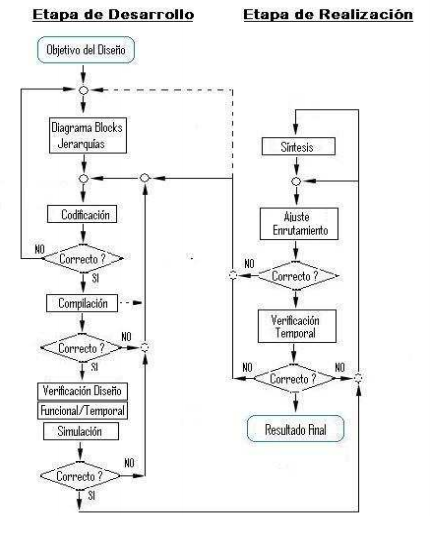


Imagen 1. Diagrama de flujo VHDL.

Descripción General

-VHDL ignora los saltos de línea y espacios, por lo tanto, el contexto formal de la escritura del programa depende de la mejor inteligibilidad del mismo.

-Comentarios: Comienzan con dos guiones – y finaliza automáticamente con el final de línea.

-Palabras clave, especiales ó reservadas: Son cadenas de caracteres especiales reservados por VHDL para su sintaxis. En el ejemplo pueden leerse varias ENTITY, PORT, IN, OUT, ACHITECTURE, ETC.

-Identificadores: Son aquellas cadenas de caracteres que el diseñador utiliza para nominar o identificar algo del diseño. En nuestro ejemplo: X, Y , Z, BIT, Compuerta, etc. Hay algunos identificadores especiales que se usan para ciertos tipos de definiciones. Es el caso de “BIT” en este ejemplo.

-Las palabras clave y los identificadores no reconocen diferencia entre mayúsculas y minúsculas.

Descripción de flujo de datos ó diseño de flujo de datos.

Esta descripción se puede realizar por el uso de otro tipo de instrucciones concurrentes, que le permiten a VHDL describir un circuito lógico en términos del flujo de la información y operaciones activadas en el circuito. Algunas de ellas son:

Instrucción de asignación de señal concurrente. Es una de las más usadas y su sintaxis es:

nombre-señal <= expresión [ Se lee “nombre señal obtiene expresión”]

En este caso el tipo de expresión debe ser compatible con el de nombre señal.

Ejemplo: LIBRARY IEEE;

use IEEE.std\_logic\_1164.all

ENTITY dec2a4 IS PORT

(enta, entb: IN std\_logic;

Sal: OUT std\_logic\_VECTOR (3 downto 0);

END dec2a4

ARCHITECTURE flujo\_datos1 OF dec2a4 IS

BEGIN

SIGNAL entaneg, entbneg : std\_logic

entaneg <= NOT enta

entbneg <= NOT entb

Sal0 <= entaneg AND entbneg ;

Sal1 <= entaneg AND entb ;

Sal2 <= enta AND entbneg ;

Sal3 <= enta AND entb ;

END flujo\_datos1