

**INGENIERIA EN MECATRONICA**

**Asignatura: SISTEMAS EMBEBIDOS**

**Maestro: Carlos Enrique Moran Garabito**

**Alumno: Fonseca Camarena Jonathan 8A**

**Matricula: 17310857**

**Periodo: Enero - Abril 20**

**Lugar:** [Universidad Politécnica de la Zona Metropolitana de Guadalajara](https://upzmg.edu.jalisco.gob.mx/lugar/13061)

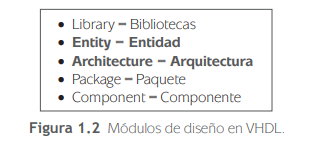
Carretera Tlajomulco-Santa Fe, km 3.5 #595, Colonia: Lomas de Tejeda, CP: 45670   
Municipio: Tlajomulco de Zúñiga

ORGANIZACIÓN Y ESTRUCTURA DEL LENGUAJE VHDL

**VHDL su estructura**

El lenguaje de descripción en hardware VHDL se estructura en módulos o unidades funcionales, identificados mediante una palabra reservada y particular de este lenguaje (véase figura 1.2). En tanto, a su vez, cada módulo tiene una secuencia de instrucciones o sentencias, las cuales, en conjunto con las declaraciones de las unidades involucradas en el programa, permiten la descripción, la comprensión, la evaluación y la solución de un sistema digital.

Al interior de la estructura de un programa, las unidades Entidad (Entity) y Arquitectura (Architecture) —en conjunto— forman la columna vertebral de este lenguaje. Por su parte, los módulos restantes, no necesariamente utilizados en la búsqueda de una solución, sirven entre otras cosas para optimizar y generalizar la aplicación en futuros desarrollos, como se verá cuando la ocasión se presente. Sin embargo, en este momento nuestra atención se centra en describir la función de la entidad y la arquitectura.



**Entidad (entity)**

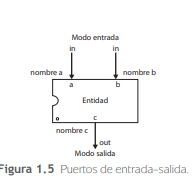
Una entidad básicamente representa la caracterización del dispositivo físico; es decir, exhibe las entradas y las salidas del circuito (llamados pins) que el diseñador ha considerado pertinentes para integrar su idea o aplicación; en la figura 1.3, se puede observar con detalle la secuencia de desarrollo.

Con base en esta idea, una entity —por la palabra reservada del programa— constituye un bloque de diseño que puede ser analizado y programado como un elemento individual, ya sea como una compuerta, un sumador o un decodificador, entre otros, incluso ser considerado como un sistema a través de su relación entre entradas y salidas, las cuales representan los puntos de observación o de conexión a elementos periféricos propios de la aplicación.

**Puertos de entrada–salida**

Cada una de las señales de entrada y salida en una entidad es referida como un puerto, el cual es equivalente a una terminal (pin) de un símbolo esquemático. Todos los puertos que son declarados deben tener un nombre, un modo y un tipo de dato.

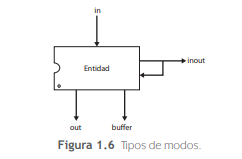
El nombre es utilizado como una forma de llamar al puerto; el modo permite definir la dirección que tomará la información, mientras que el tipo precisa qué clase de información se transmitirá a través del puerto. Por ejemplo, en el caso de los puertos de la entidad representada en la figura 1.5, aquellos que son de entrada están indicados por las variables a y b; mientras que el puerto de salida se representa por la variable c. Por otra parte, el tipo de dato será tratado más adelante.



**Modos**

Como se mencionó antes, un modo permite definir la dirección hacia donde el dato es transferido. Un modo puede tener uno de cuatro valores: in (entrada), out (salida), inout (entrada/salida) y buffer (véase figura 1.6).

* Modo in. Se refiere a las señales de entrada a la entidad. El modo in es solo unidireccional y únicamente permite el flujo de datos hacia dentro de la en tidad.
* Modo out. Indica las señales de salida de la entidad.
* Modo inout. Permite declarar a un puerto de forma bidireccional, es decir como de entrada/salida, además hace posible la retroalimentación de señales dentro o fuera de la entidad.
* Modo buffer. Permite realizar retroalimentaciones dentro de la entidad; pero, a diferencia del modo inout, el puerto declarado se comporta como una terminal exclusiva de salida.



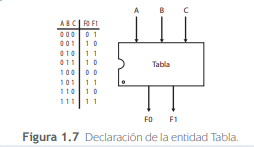
**Tipos de datos**

Los tipos son los valores (datos) que el diseñador establece para los puertos de entrada y salida dentro de una entidad, y que son asignados de acuerdo con las características de un diseño en particular. Algunos de los tipos más utilizados son el bit, el cual tiene valores de 0 y 1 lógico; el tipo boolean (booleano) define valores de verdadero o falso en una expresión; el bit\_vector (vectores de bits), el cual representa un conjunto de bits para cada variable de entrada y/o salida, y el tipo integer (entero), que representa a un número entero.

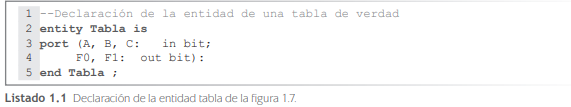
Los anteriores son solo algunos de los tipos que maneja VHDL, aunque no son los únicos.1 Los tipos de datos y su uso se introducirán conforme se vayan requiriendo y empleando a lo largo del texto.

**Declaración de entidades**

La declaración de una entidad consiste en describir las entradas y las salidas de un circuito identificado como Entity (entidad); en otras palabras, la declaración señala las terminales o los pines de entrada y de salida con los que cuenta el circuito. Por ejemplo, considérese la tabla de verdad que se muestra en la figura 1.7; como se puede observar, esta tiene tres entradas, A, B y C, y dos salidas, F0 y F1. En este caso, a la entidad se le ha identificado con el nombre de TABLA, tal y como se muestra.



La descripción de este programa se muestra en el listado 1.1, donde los números de las líneas (1, 2, 3, 4, 5) no son parte del código y serán utilizadas como referencia para explicar alguna sección en particular. En tanto, las palabras en negritas están reservadas para el lenguaje de programación, es decir, tienen un significado especial para el programa.



Ahora comencemos con el análisis del código línea por línea. Así, la línea 1 inicia con dos guiones (- -), los cuales indican que el texto que está a la derecha es un comentario; estos se usan únicamente con el fin de documentar el programa, ya que todos los comentarios son ignorados por el compilador. En la línea 2 se empieza la declaración de la entidad utilizando la palabra reservada entity, seguida del identificador o nombre de la entidad “Tabla” (para este ejemplo) y la palabra reservada is.

Los puertos de entrada y salida (port) son declarados en las líneas 3 y 4, respectivamente; en este caso, los pines de entrada son A, B y C, mientras que los puertos de salida están representados por F0 y F1. Si se observa con atención la tabla de verdad de la figura 1.7, es posible apreciar que en su descripción solo se utilizan valores lógicos (‘0’ y ‘1’); por tanto, es de suponer que el tipo de dato empleado en la declaración es bit. Por último, en la línea 5 termina la declaración de entidad con la palabra reservada end, seguida del nombre de la entidad Tabla.

**Aspectos importantes a considerar**

Como en cualquier lenguaje de programación, es importante hacer notar que VHDL también sigue una sintaxis y una semántica en el código de programación, las cuales es necesario respetar lo siguiente.

* Punto y coma (;): se utiliza para cerrar y finalizar declaraciones.
* Dos puntos (:): en este caso se usan como separador entre el nombre de los puertos y los modos de entrada.
* El paréntesis: después del modo de salida out bit cierra la declaración de los puertos (port).
* El uso de las mayúsculas o las minúsculas en la declaración es irrelevante para el compilador.

**Bibliografía:**

[1] Manuel Coutinho, José Rufino, Carlos Almeida, “Control of Event Handling

Timeliness in RTEMS”, Proceedings of the 17th IASTED International Conference

on Parallel and Distributed Computing and Systems (PDCS 2005), Phoenix, AZ,

USA, November 14 - 16, 2005.

[2] Helen Custer, “Inside Windows NT”, Microsoft Press, 1992.

[3] A. Damm, J. Reisinger, W. Schwabl, y H. Kopetz. “The Real-Time Operating

System of MARS”. Operating System Review, 23(3): 141-157, Julio 1989.