+Taller 1

Sergio Bolaños Ramirez

Universidad Autónoma de Occidente

Ingeniería Electrónica y Telecomunicaciones

Cali, Colombia

Sergio.bolanos@uao.edu.co

Abstract— The following report will introduce concepts of the hardware description language and some of its advantages and structures with which it is built.

Keywords: Constant, Variable, Signal.

Resumen-- En el siguiente informe se dará a conocer conceptos del lenguaje de descripción de hardware y algunas de sus ventajas y estructuras con las que se construye.

Palabras clave: Constante, Variable, Señal.

1. Marco Teórico

**Constante**: Una constante es un elemento de datos con nombre con un valor predefinido. No se puede cambiar el valor asignado a una constante predefinida. Las constantes predefinidas son:

* NULL: Una referencia vacía, lo que sería parecido a un puntero vacío. No es lo mismo una cadena de caracteres nulos.
* TRUE: Es equivalente a 1.
* FALSE: Es equivalente a 0.

**Variables:** Es un elemento de datos con nombre cuyo valor puede cambiar durante el curso de la ejecución de un programa. Un nombre de variable debe seguir el convenio de denominación de un identificador ( Carácter alfabético o numérico y el signo de subrayado). Cuando se define más de una variable en una sola declaración, el nombre debe ir separado por comas. Cada declaración debe finalizar con un signo de punto y coma. Los nombres de variables no pueden coincidir con una palabra reservada.

* Globales: Una variable es global a no se que esté declarada dentro de una definición de función. Las variables globales resultan visibles y disponibles para todas las sentencias de un script.
* Locales: Sólo resultan visibles y disponibles dentro de una función en la que están definidas. Aunque los nombres de variables e identificadores deben ser exclusivos resulta válido declarar una variable local con el mismo nombre de una variable global con el mismo nombre que una variable local definida dentro de otra función.

Aquí las variables las declaramos en procesos o subprogramas donde se ejecutan sentencias en serie.

**Señal:** Una señal es una abstracción de cualquier cantidad mesurable que es función de una o más variables independientes. Este puede ser de varios tipos: bus, register o normal, puede tomar algún valor inicial y solo se actualizan al terminar el proceso en el cual están siendo utilizadas, a diferencia de las variables que estas se actualizan instantáneamente, o sea que su valor cambia en el momento de la asignación. Las señales presentan la propiedad de almacenar el valor actual y de guardar un valor futuro.

Las señales se declaran en paquetes, procesos concurrentes o en las arquitecturas.

Diferencias principales entre señales, variables y constantes:

* Las constantes ya reciben un valor inicial en su declaración.
* Las variables primero se declaran, luego se inician, y luego se usan.
* Las constantes, una vez declaradas mantienen su valor durante toda la ejecución del programa.
* Las variables cambian su valor tantas veces como se desee.
* Las variables también pueden cambiar su tamaño durante el tiempo de ejecución.
* Una señal se declara a nivel de la arquitectura y puede ser leída por varios procesos.
* Una variable se declara dentro de un proceso y puede ser leída y escrita por el.
* La asignación de una variable se realiza mediante el operador := .
* La asignación a una señal se hace mediante la <=.
* Los puertos se comportan como señales.
* La señal toma los valores a medida que avanza el tiempo de la simulación.

**Descripción Algorítmica:** Esta forma permite describir un circuito lógico en términos del comportamiento del mismo, la forma más clara de describirlo es a través del proceso. Para ello se define un proceso como un conjunto de instrucciones secuenciales que se ejecutan en paralelo con otros procesos e instrucciones concurrentes. Utilizando un proceso se puede especificar una compleja interacción de señales y eventos, concretando un circuito combinacional o secuencial que realiza la operación modelada. Esta descripción una las siguiente herramientas :

* Process: Es una instrucción de proceso que puede emplearse en cualquier parte del programa donde se puede utilizar una instrucción concurrente.
* Secuenciales: Es una instrucción que ocurre dentro de un proceso y tiene la misma sintaxis que su versión concurrente: if then, if-then-else, elsif, case, loop, for, exit y next, while.
  + If-then: es una instrucción secuencial muy usada y permite mejorar la descripción de su comportamiento del circuito que se pretende diseñar.
  + Elsif: permite crear instrucciones anidadas if-then-else, la instrucción secuencial de la cláusula elsif se ejecuta si su expresión booleana es verdadera y todas precedentes son falsas.
  + Case: es una instrucción que permite seleccionar entre múltiples alternativas basadas en el valor de una señal o expresión. Se usa como una alternativa más legible y para producir un circuito mejor sintetizado frente a las instrucciones if.
  + Loop: es una instrucción secuencial muy importante y de mucho uso, ya que permite crear un ciclo o lazo infinito.
  + For: es una instrucción que permite realizar ciclos definidos.
  + Exit y Next: se trata de un par de instrucciones muy útiles cuando se usan lazos, que permite pasar de estos ciclos o finalizarlos.
  + While: es una instrucción que permite realizar ciclos, aquí la expresión booleana se prueba antes de cada interacción del ciclo y el ciclo se ejecuta solo si el valor de dicha expresión es verdadera.

Referencias

1. Señales y variables. <http://panamahitek.com/diferencia-entre-variables-y-senales-en-vhdl/>
2. VHDL. <http://www1.frm.utn.edu.ar/tecnicad1/_private/Apuntes/VHDL.pdf>
3. Algoritmico. <https://www.fceia.unr.edu.ar/eca1/files/LDD/vhdl-Introduccion2007.pdf>