**VHDL**

ING MECATRONICA

8.B

PROGRAMACION DE SISTEMAS EMBEBIDOS

BARAJAS MORALES MARTIN

MORAN GARABITO CARLOS ENRIQUE



VHDL es un lenguaje de especificación definido por el IEEE utilizado para describir circuitos digitales y para la automatización de diseño electrónico. VHDL es acrónimo proveniente de la combinación de dos acrónimos: VHSIC y HDL.

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:

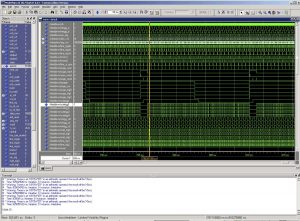
* Cada elemento de diseño tiene una interfaz única y perfectamente definida, que permite conectarla a otros elementos.
* Cada elemento tiene un comportamiento preciso y unívocamente definido, que permiten su posterior simulación.
* La especificación de comportamiento que permite definir la operatividad 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.

VHDL es un lenguaje de descripción de hardware, no es un lenguaje de programación, aunque su sintaxis y forma de uso se asemeje mucho. Con VHDL no programamos, lo que se hace es tomar un diseño electrónico y describirlo mediante el lenguaje, de forma que sirva como punto de entrada a herramientas que convertirán esa descripción en una implementación real del sistema.

De esta forma las descripciones de circuitos en VHDL son siempre compatibles independientemente de las herramientas utilizadas, ya que el lenguaje fue convertido en un standard del IEEE y el problema del intercambio de diseños desaparece, surgiendo de esta forma todo un mercado de compra y venta componentes electrónicos reusables, llamados también de IP (Intellectual Property).

Aparte de permitir un intercambio eficiente de componentes, el uso de VHDL ofrece muchas de ventajas. La más significativa es que utilizando el lenguaje es posible tanto describir como simular el sistema. El propio lenguaje ofrece las herramientas para escribir módulos que apliquen entradas al sistema y mediante una herramienta llamada [simulador](http://model.com/) poder ver las salidas y verificar si son correctas. Esto permite crear entornos complejos de verificación para comprobar la corrección del sistema, algo fundamental en diseños que van a ser enviados a dispositivos físicos y donde un error en el diseño puede acarrear unos costes catastróficos para una empresa.



Ejemplo de simulación en VHDL usando la herramienta Modelsim

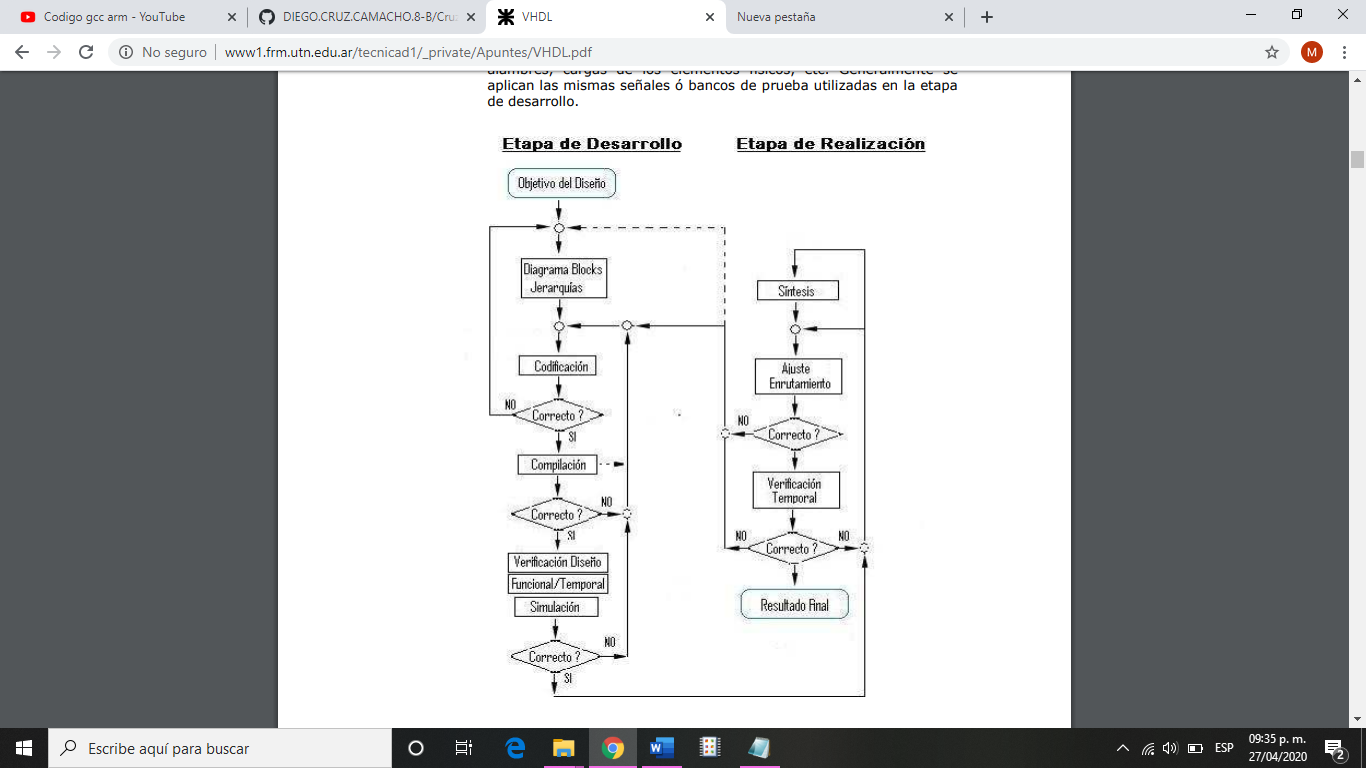
En cualquier diseño basado en VHDL se puede subdividir el flujo de diseño en dos partes bien diferenciadas:

Etapa de desarrollo:

* Planteo General del Sistema: Consiste en hacer un Diagrama en Blocks funcional y jerárquico del sistema a desarrollar. Definición de módulos e interfaces generales del sistema.
* Codificación: Consiste en escribir el código VHDL para todos los componentes planteados, módulos específicos e interfaces. Se trata de un simple discurso de texto, por lo que puede realizarse en cualquier editor conocido por el diseñador. Sin embargo, los ambientes de diseño incluyen un Editor VHDL especializado, por lo que el desarrollo de esta etapa es más sencillo. Estos editores incluyen características propias del lenguaje, como sangría automática, resaltado de palabras clave, verificación sintáctica, etc.
* Compilación: En esta etapa el compilador VHDL transforma el programa fuente en objeto, por lo cual analiza la sintaxis de lo escrito y verifica la compatibilidad con cualquier otro módulo ingresados como fuente del presente programa. Junto a esto produce toda la información necesaria para la posterior simulación del proyecto. Nota: A veces, en proyectos complejos conviene realizar compilaciones parciales, ganando así tiempo posterior de desarrollo.
* Verificación: Esta etapa es muy importante, pues permite establecer si el circuito obtenido funciona como se pretendió al fijar las pautas de diseño. Existen dos características a verificar, y ellas son las Funcional y la Temporal. En el primer caso se analiza el funcionamiento lógico del circuito, sin considerar el tiempo como variable. Aquí los elementos lógicos son considerados como ideales, es decir sin retardos. En la verificación temporal se analizan los resultados del circuito real, considerando retardos estimados, ya que aún no se han seleccionado los circuitos reales de síntesis. De esta manera se verifica la funcionalidad temporal de circuitos combinacionales y especialmente los secuenciales con características de memorización. La verificación funcional y temporal se realiza a través de un proceso complejo conocido como simulación, el cual permite detectar errores en el diseño obtenido, y de esta forma hacer las correcciones adecuadas antes de pasar a la etapa de síntesis.
* Simulación: Este procedimiento de verificación permite definir entradas y aplicarlas al prototipo de software, analizar el comportamiento de los diversos módulos definidos y observar las salidas. Todo esto sin tener que realizar el prototipo físico con circuitos reales. Para proyectos pequeños se pueden generar entradas y verificar salidas en forma manual; pero en grandes sistemas se crean ¨bancos de prueba” con la capacidad de establecer entradas, verificar salidas y realizar las comparativas con los valores esperados. La simulación funcional es completa y precisa, sin embargo, la temporal sólo es aproximada pues se basa en valores estimados, y sirve para establecer que vamos en el camino correcto. Esto es así pues es muy dependiente de la síntesis, dónde se establecen los circuitos específicos y es allí dónde salen los verdaderos valores de retardos de acuerdo al circuito, a la tecnología, al layout, tipo de componentes discretos, etc.

Etapa de realización:

* Síntesis: Convierte el modelo descripto por VHDL en un conjunto de primitivas o componentes que se articularán en un circuito real en una tecnología adecuada. Generalmente estas herramientas presentan la posibilidad de establecer filtros, premisas o restricciones específicas del tipo de circuito ó tecnología. Por ejemplo, en los circuitos lógicos programables, como PAL, GAL, CPLD, FPGA las herramientas de síntesis pueden generar ecuaciones booleanas; y en el caso de ASIC ó ASSP generar una lista de compuertas y la malla que fija el cableado correspondiente de interconexión.
* Ajuste y Enrutamiento: Son herramientas que mapean las ecuaciones o componentes de acuerdo a los recursos disponibles de cada dispositivo. Como en el paso anterior, el diseñador puede especificar restricciones o asignaciones específicas para lograr el módulo correcto.
* Verificación temporal y total del circuito: Este es la etapa crucial ya que establece la funcionalidad temporal correcta basada en todos los parámetros reales introducidos en el diseño, como circuitos integrados reales, longitud de los buses incorporados, longitud de los alambres, cargas de los elementos físicos, etc. Generalmente se aplican las mismas señales ó bancos de prueba utilizadas en la etapa de desarrollo.



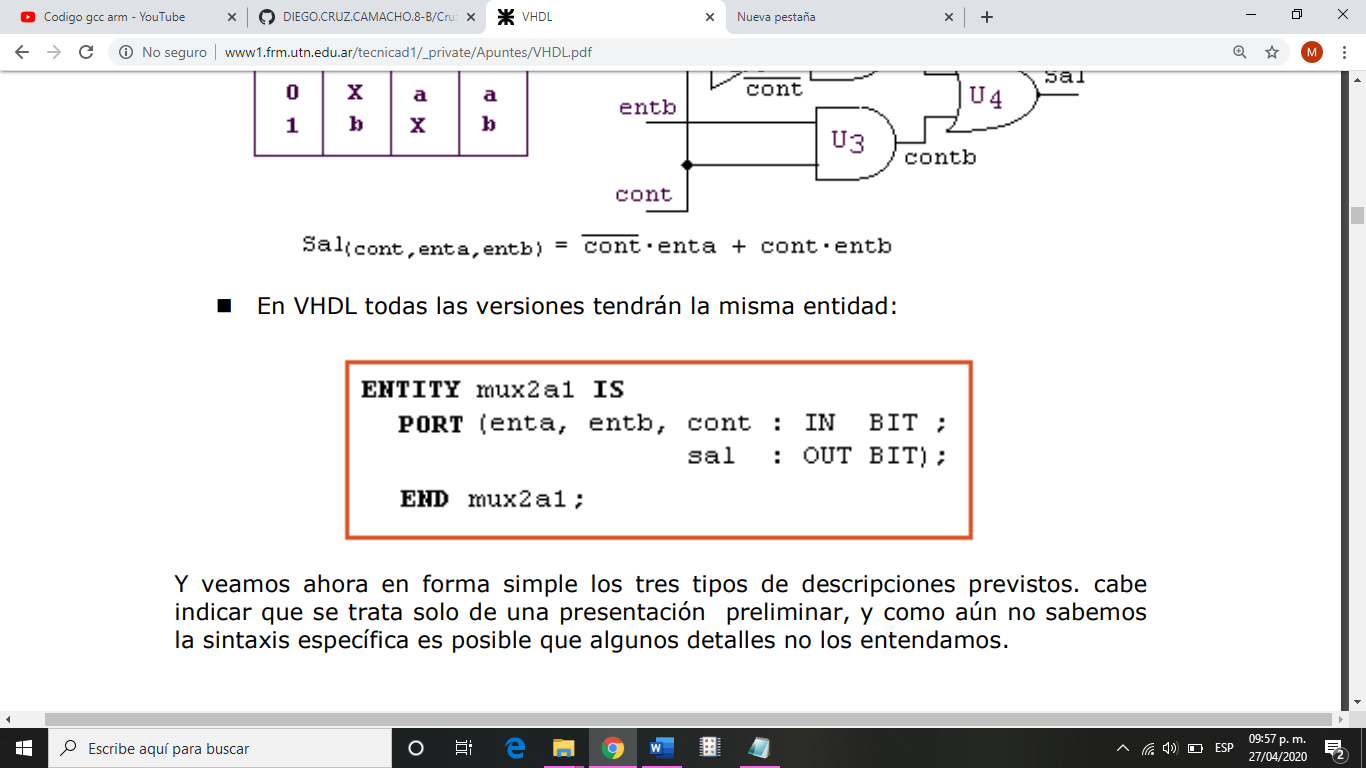
VHDL ignora los saltos de línea y espacios, por lo tanto, el contexto formal de la escritura del programa depende de la mejor inteligibilidad del mismo. Comentarios: Comienzan con dos guiones – y finaliza automáticamente con el final de línea.

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.

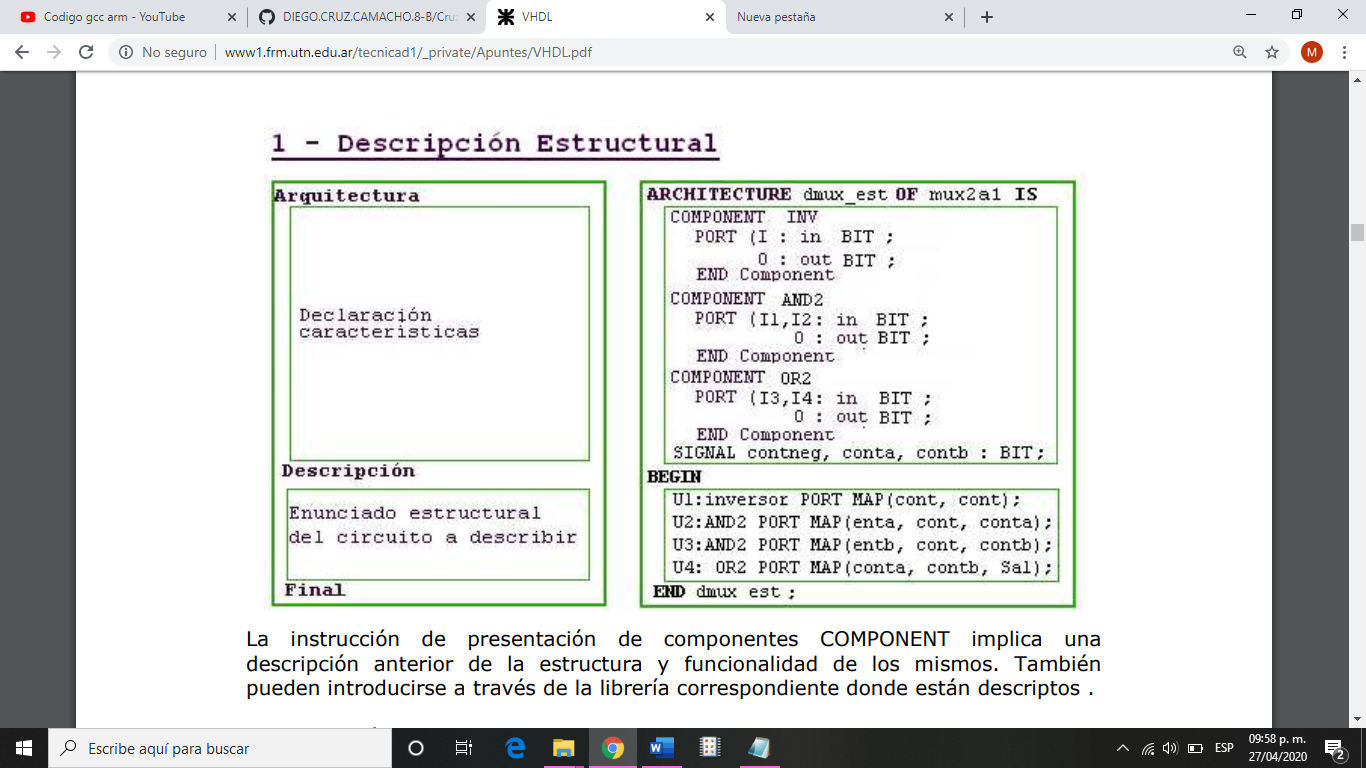
Identificadores: Son aquellas cadenas de caracteres que el diseñador utiliza para nominar o identificar algo del diseño. En nuestro ejemplo: X, Y , Z, BIT, Compuerta, etc. Hay algunos identificadores especiales que se usan para ciertos tipos de definiciones. Es el caso de “BIT” en este ejemplo.

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

En VHDL todas las versiones tendrán la misma entidad:

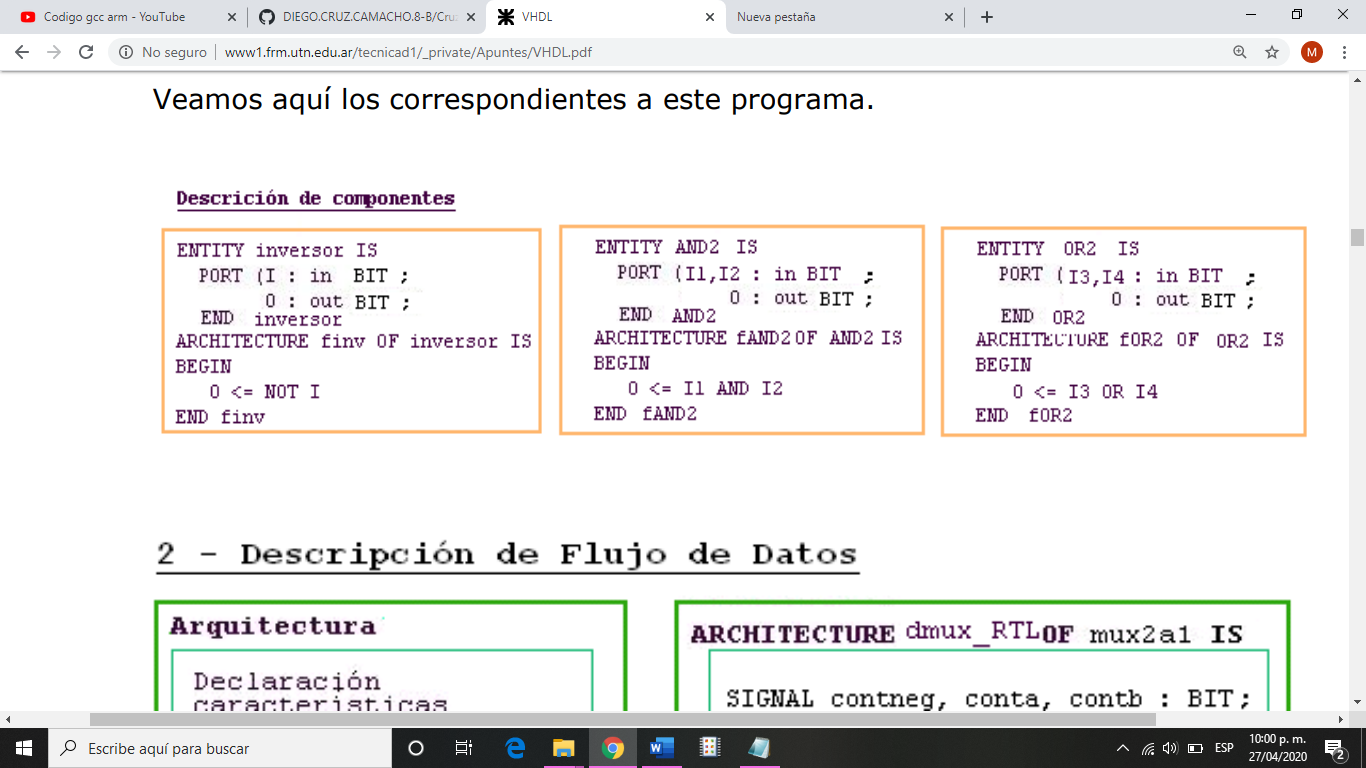


Descipcion estructural

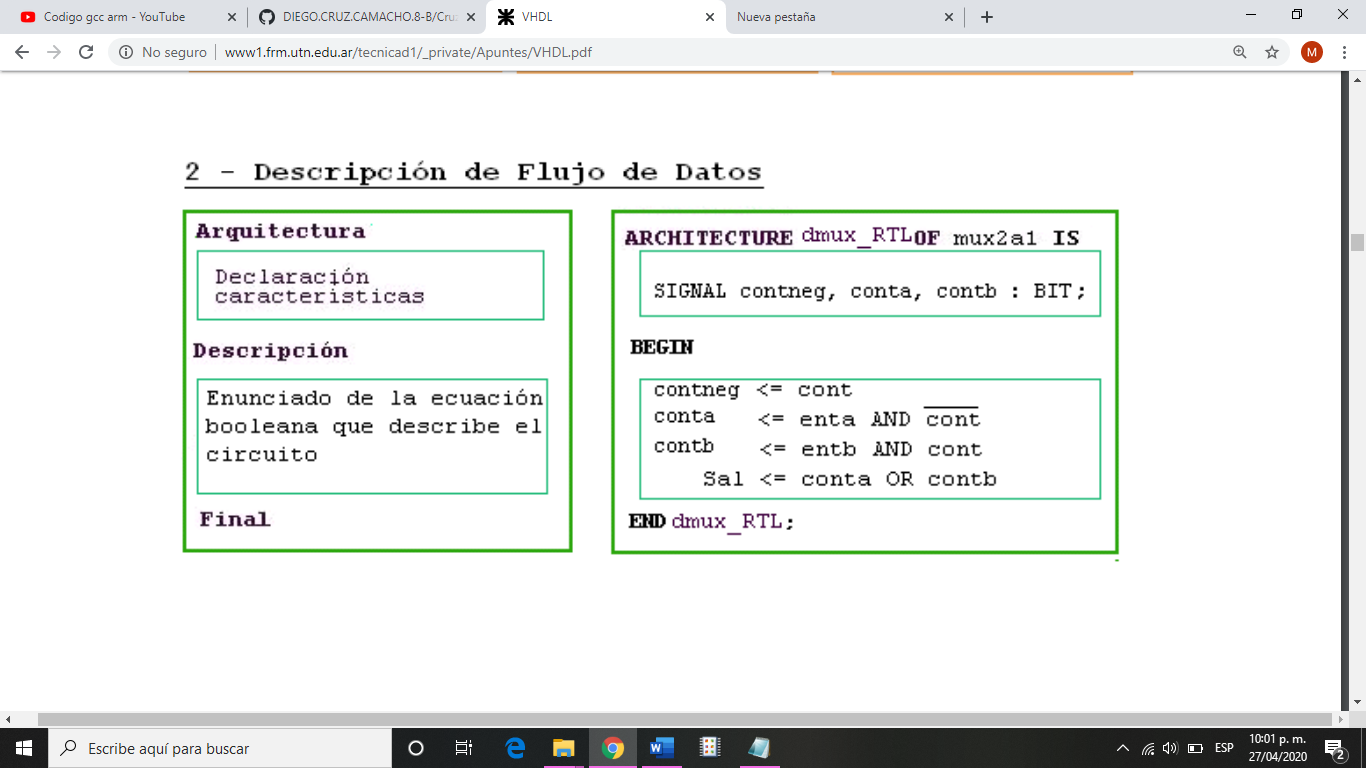


La instrucción de presentación de componentes COMPONENT implica una descripción anterior de la estructura y funcionalidad de los mismos. También pueden introducirse a través de la librería correspondiente donde están descriptos

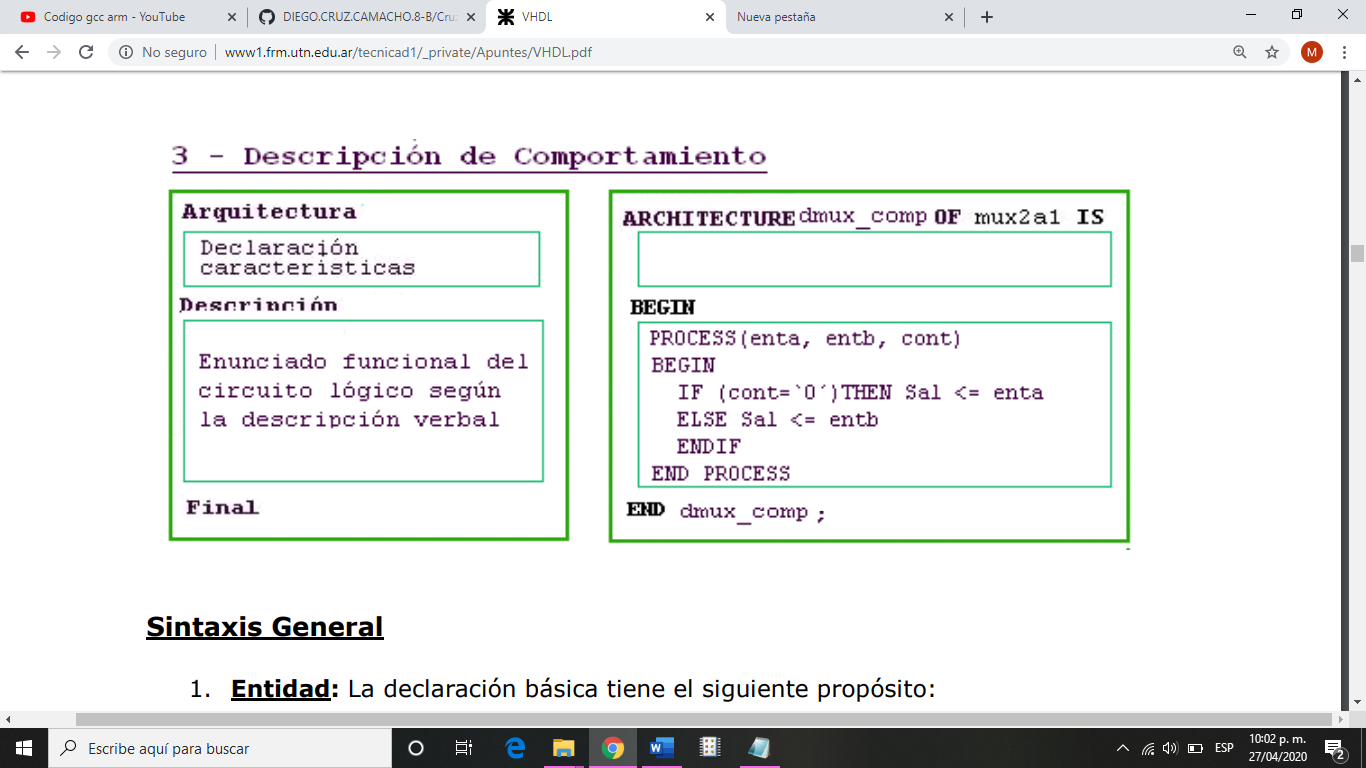
Descripcion de componentes



Descripcion de datos



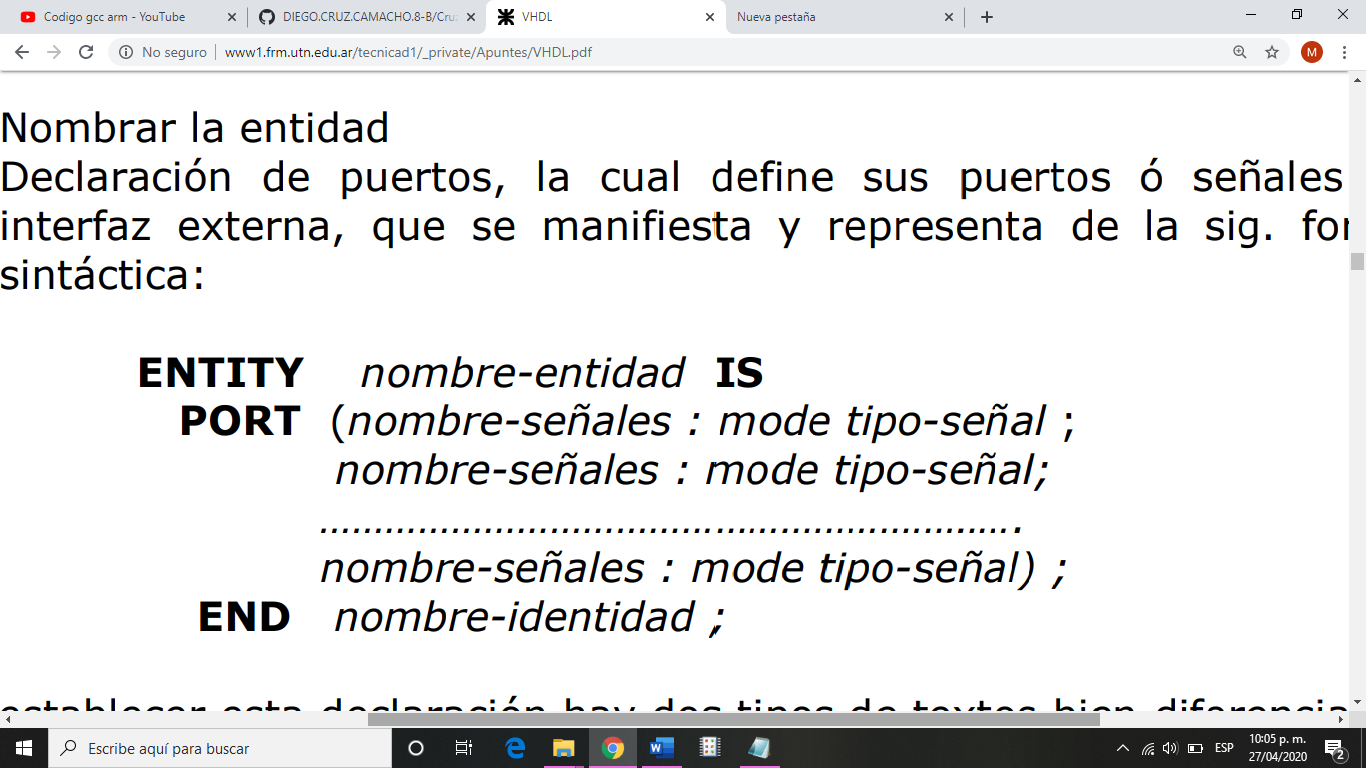
Descripción de comportamiento



SINTAXIS GENERAL

Entidad: La declaración básica tiene el siguiente propósito:

* Nombrar la entidad
* Declaración de puertos, la cual define sus puertos ó señales de interfaz externa, que se manifiesta y representa de la sig. forma sintáctica:



Para establecer esta declaración hay dos tipos de textos bien diferenciados:

* Palabras clave: ENTITY , IS , PORT y END.
* Palabras nominativas:
* nombre-entidad: Identificador que nombra la entidad, no puede ser palabra clave.
* nombre-señales: Identificadores que nombran las variables y funciones de la interfaz externa.
* modo: Identificador que especifica la dirección de la señal, usando 4 palabras clave, que son:
* IN : Indica una señal de entrada a la entidad
* OUT : Indica una señal de salida. El valor de la señal no puede ser leído dentro de la arquitectura de la entidad.
* BUFFER: Indica una señal de salida. El valor de la señal si puede ser leído dentro de la arquitectura de la entidad.
* INOUT: Indica una señal de entrada o salida de la entidad. Se utiliza mucho en los terminales entrada/Salida de tercer estado de los dispositivos lógicos programables.

CONCLUSION

El lenguaje vhdl de ser necesario, verificar cada componente del sistema por separado y también permite diseñar, modelar un sistema de alto nivel. Los diseños echos con vhdl no solo pueden ser simulados sino también sintetizados con herramientas adecuadas.