Tecnología de Computadores

GUIA DEL SIMULADOR VERIBEST

Facultad de Informática – Universidad Politécnica de Madrid – V. Rodellar

GUIA del SIMULADOR VHDL de VERIBEST

El objetivo de esta guía es facilitar un primer contacto con el uso del simulador de VHDL de Veribest. La utilización de los comandos y funciones más comunes de la herramienta, se presenta a través de la realización de un ejemplo sencillo, el cual aborda desde la edición de un fichero hasta la visualización de las formas de onda de los resultados, pasando por la compilación y la simulación.

La herramienta tiene una ayuda ***on line***, que se recomienda utilizar cuando se considere necesaria. Para activarla hay que seleccionar el botón de la ayuda ***on line*** (véase Figura-1), aparecerá un símbolo de interrogación que se puede desplazar por la ventana, y que deberá posicionarse sobre el icono, ventana, palabra etc., que sea motivo de consulta, apareciendo a continuación la correspondiente información de ese aspecto especifico.

1) Arranque del simulador

Si la instalación de la herramienta se ha realizado en el disco C y no se han modificado las rutas que vienen por defecto hay que seleccionar:

C:\Programs → VeriBest VB99.0 → VeriBest VHDL simulator → VeriBest VHDL

La ventana del simulador con indicación de algunas de las funciones de la barra de herramientas y del espacio de trabajo se muestra en la siguiente figura:

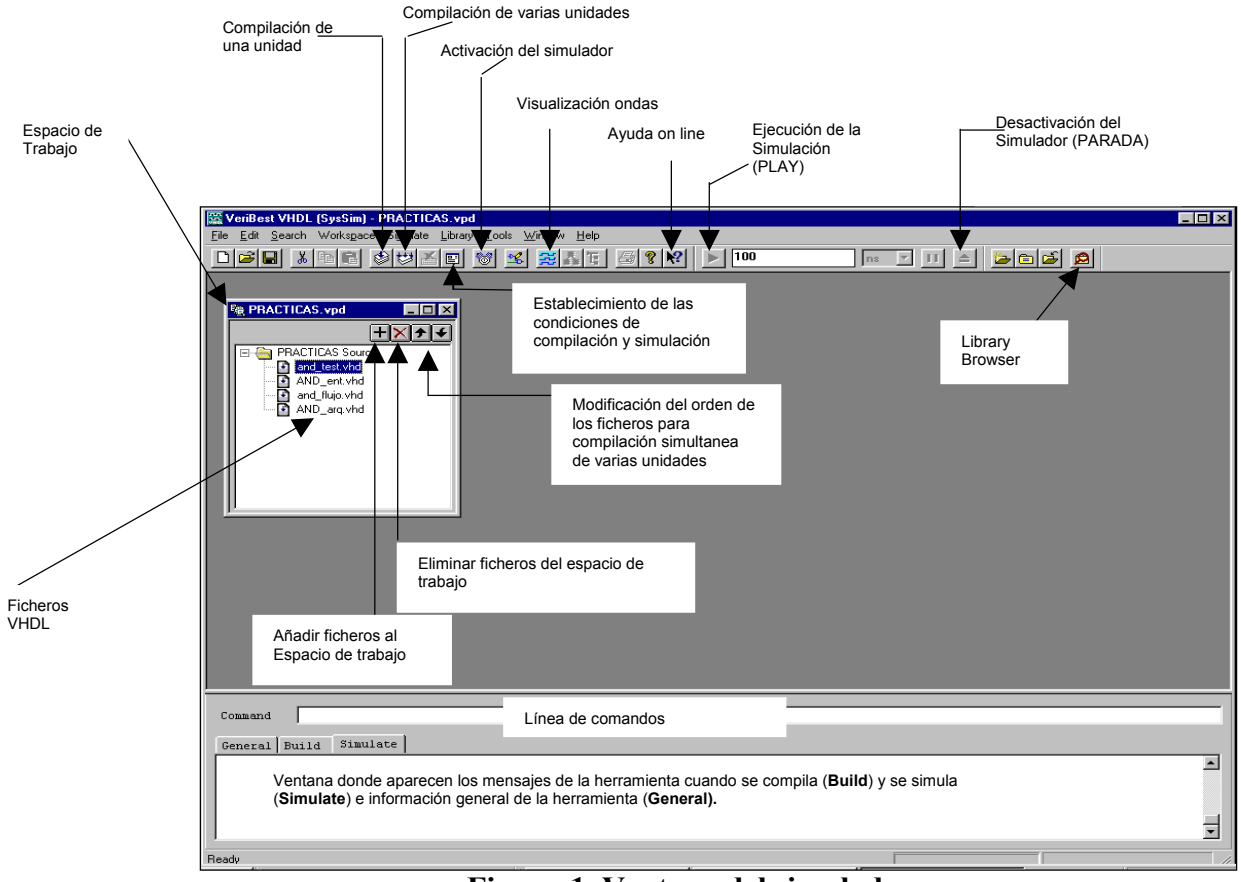


Figura-1. Ventana del simulador

2) Creación de un espacio de trabajo

- Cuando se comienza a trabajar por primera vez con la herramienta, no hay ficheros VHDL dentro de la misma, solamente las bibliotecas propias del simulador. Entonces lo primero que hay que hacer es crearse un espacio de trabajo, seleccionando en la barra de herramientas **Workspace** y dentro de ella **New.**

- Nos aparecerá una ventana pidiéndonos el nombre del espacio de trabajo, que puede ser cualquiera, por ejemplo **PRACTICAS**, y su ruta (**PRACTICAS_VHDL)**, a continuación activar el botón **Create.**

- De forma inmediata aparece otra ventana con el nombre del espacio de trabajo que hemos creado y con extensión **vpd** (**PRACTICAS.vpd).** Dentro de ella, aparece una serie de botones cuya utilidad comentaremos más adelante o a lo largo del desarrollo de las sesiones de prácticas, y una carpeta con el nombre **PRACTICAS source,** que es donde se van a depositar todos los ficheros que escribamos dentro de este espacio de trabajo.

3) Edición de un fichero nuevo

- Para editar un nuevo fichero se selecciona en la barra de herramientas **File (New)**, a continuación aparece una nueva ventana que nos da las opciones de **VHDL source file o VHDL Workspace,** elegimos la primera opción, ya que lo que queremos es editar un fichero nuevo y no crear otro espacio de trabajo, ya lo tenemos creado en el paso anterior **(2)**, aunque también se puede crear desde aquí, si se desea.

- Nos aparece una ventana con un nombre por defecto **(VHDL1**), que nosotros vamos a personalizar salvándolo con el nombre del fichero que nos interese. El ejemplo que vamos a escribir es una entidad de una puerta **and** de dos entradas, por tanto un posible nombre para ese fichero podría ser el nombre de la puerta, y un complemento que nos recordase que se trata de una entidad y su número de puertas, aunque esto no es imprescindible. El nombre del fichero que contiene la entidad, que es lo que estamos guardando ahora, y el nombre de la entidad pueden ser iguales o no, depende de las preferencias del diseñador **(Save as AND2_ent.vhd)**. Se observará que tras esta acción, aparece la ventana actualizada el nombre del fichero.

- A partir de este momento ya podemos introducir nuestro código, obsérvese que las palabras reservadas del lenguaje aparecen escritas con mayúsculas y las demás con minúsculas, y aunque el compilador no distingue entre mayúsculas y minúsculas, se recomienda esta práctica:

```
ENTITY and_2 IS
  PORT (e0,e1: IN BIT; s: OUT BIT);
END and_2;
```

A continuación almacenamos el diseño **(Save)**. Se debe observar que la ventana del espacio de trabajo permanece abierta (**PRACTICAS Source),** y lo primero que vamos a hacer es añadir el fichero que acabamos de escribir a nuestro

espacio de trabajo. Para ello, activaremos la tecla + del espacio de trabajo (véase Figura-1), nos saldrá una nueva ventana, **Add Files into Workspace,** y sobre ella, seleccionamos el fichero **AND2_ent.vhd**. Se observará entonces, que el fichero se añade a la carpeta **PRACTICAS Source,** a partir de este momento se puede cerrar la ventana donde hemos editado el fichero.

4) Compilación

- Lo primero que hay que establecer son las condiciones de compilación. Para ello hay que ir de nuevo a la barra de herramientas, seleccionar **Workspace** y dentro de ella **Settings** o bien activar el correspondiente botón (véase Figura-1). Aparecerá una nueva ventana con la carpeta **PRACTICAS Source,** y si damos dos clics sobre ella se abrirá, apareciendo el fichero **AND2_ent.vhd.** Seleccionamos el fichero anterior, y de las posibles opciones que nos da la ventana seleccionamos **Compile y Debug**.

- Una vez establecidas las condiciones de compilación, esta se puede lanzar de dos formas:

  - Desplegando **Workspace** en la barra de herramientas, aparece la opción **Compile AND2_ent.vhd.** Cuando esta opción se activa en la ventana inferior del simulador **(General, Build, Simulate)** (véase la Figura-1), aparece una serie de mensajes relativos a las distintas acciones que va realizando la herramienta**.** Si hemos seleccionado **General** nos informa de todas las acciones que se van realizando, si se selecciona **Build** nos dará información sólo de cómo se ha realizado la compilación, los errores de sintaxis que se hayan podido cometer, y el número de sentencia donde está, y por último **Simulate,** nos proporciona únicamente información de la fase relativa a la simulación**.** En el caso que acabamos de describir, la información será similar al texto que se reproduce a continuación:

    ```
    vc comp -udebug -uvital:off  C:\PRACTICAS_VHDL\AND2_ent.vhd
    VeriBest VHDL Compiler - 15.00.00.25

    Compiling Entity Declaration AND_2

    Done
    ```

  - Seleccionando el fichero **AND2_ent.vhd** en la ventana del espacio de trabajo y activando en la barra de herramientas el botón para compilación de una sola unidad (véase Figura-1). Si se hace esto, se observará los mismos efectos en la ventana inferior del simulador que en el caso anterior.

Una vez que la compilación no de errores, se almacena el fichero que se ha compilado **(SAVE o icono).** La unidad compilada se almacenará en la librería **WORK**. Para ver donde está ubicada esta unidad de diseño, en la barra de herramientas se selecciona **Library** y dentro de ella **Library Browser** o bien se activa el correspondiente botón (véase Figura-1), aparecerá una nueva ventana con unas carpetas resaltadas en amarillo (IEEE, STD, VB), estos son los nombres lógicos de las librerías que están accesibles a la herramienta, y que pueden visualizarse e imprimirse, si se desea. También aparece una carpeta resaltada en rojo con el nombre de **WORK, WORKLIB,** si se abre esta librería aparece el

símbolo de un chip con el cuerpo en blanco que lleva asociado el nombre de nuestra entidad AND_2, ¡no el nombre del fichero donde se encuentra almacenada!. La parte derecha de esta ventana nos da la información de la ruta o nombre físico de las librerías y ficheros, la fecha de cuando se compilaron y las opciones de compilación que se utilizaron. En la parte inferior de la ventana, en el caso de haber seleccionado **General,** aparecerá la palabra **libbrows,** que no es mas que la abreviatrura de **Lib**rary **Brows**er.

5) Preparación de un diseño

- ***Descripción arquitectura***

Todavía no podemos simular ningún diseño, ya que para ello es necesario que la entidad tenga asociada al menos una arquitectura y un fichero de test. Vamos entonces a escribir una arquitectura en estilo flujo de datos para la entidad de la puerta **and** anterior, siguiendo los pasos indicados en el punto **3).** Para tener una mayor flexibilidad, se aconseja que la entidad y arquitectura estén en ficheros separados. El nombre que le daremos al fichero será **AND2_arq.vhd,** y a la arquitectura **and2_flujo.**

```
ARCHITECTURE and2_flujo OF and_2 IS
   BEGIN
    s<= e0 AND e1;
END and2_flujo
```

El fragmento de código anterior tiene un error de sintaxis, en la ultima sentencia falta el identificador de terminación de línea (**;**). Cuando compilemos la arquitectura (siguiendo los pasos indicados en el punto **4**) en la ventana de mensajes (**Build)**, aparecerá el siguiente:

```
vc comp -udebug -uvital:off  C:\PRACTICAS_VHDL\and2_flujo.vhd
VeriBest VHDL Compiler - 15.00.00.25

Compiling Architecture AND2_FLUJO of AND_2

-------------------
    6:
        ^
[Error] Unexpected End of File

Done
```

A continuación se corregirá el error de sintaxis, y se volverá a recompilar el diseño. En este momento, se aconseja explorar la librería **WORK,WORKLIB,** como se indica en el punto **4).** Se observará, que colgada de la entidad **AND_2** aparece el símbolo de un chip con el cuerpo en rojo que tiene el nombre de nuestra arquitectura **AND2_FLUJO**, todas las arquitecturas que asociemos a esta entidad ocuparan el mismo lugar en la jerarquía.

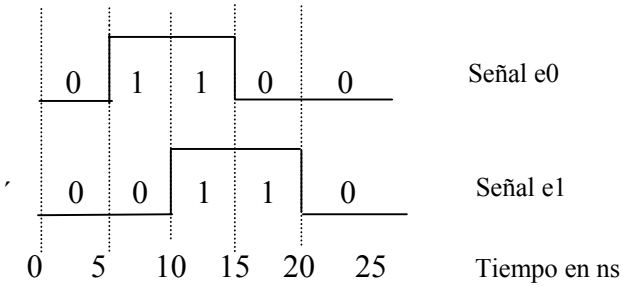
- ***Descripción del test***

  - ***Preparación de los estímulos de entrada***

El test, siempre que las dimensiones del problema lo permita, debe contener todas las posibles combinaciones de los valores de las entradas, en nuestro caso las posibles combinaciones se reducen a las siguientes:

e1	e2	s
0	0	0
0	1	0
1	0	0
1	1	1

Unas posibles formas de onda (drivers) que recogiesen la totalidad de los casos de la tabla anterior serian:



- ***Escritura del fichero de test***

Para testear un componente es necesario escribir una entidad y una arquitectura. Las entidades para test no tienen puertos. Y las arquitecturas son de tipo estructural, incluyendo en su parte declarativa el componente que se va a testear, la arquitectura que se le asocia y la declaración de señales.

```
ENTITY and2_test IS
END and2_test;

ARCHITECTURE test_flujo OF and2_test IS
--
--Parte declarativa
--
 COMPONENT
   and_2 PORT(e0,e1: IN BIT; s: OUT BIT);
 END COMPONENT;
--
 FOR I: and_2 USE ENTITY WORK.and_2(and2_flujo);
--
 SIGNAL e0,e1,s: BIT;
--
--Cuerpo de la arquitectura
--
 BEGIN
--
 I: and_2 PORT MAP (e0,e1,s);
--
```

```
    e0 <='0','1'AFTER 5 NS, '0'AFTER 15 NS;
--
    e1<='0','1'AFTER 10 NS,'0'AFTER 20 NS;
--
END test_flujo;
```

Aunque las entidades y arquitecturas deben ir normalmente en ficheros separados, en este caso, para simplificar el manejo de ficheros, la entidad para test y su arquitectura se editaran en un mismo fichero, que tendrá por nombre **and2_test.vhd**. Se compilará el fichero depurando los posibles errores de sintaxis, y por último se incorporará al espacio de trabajo, siguiendo los pasos mencionados en los apartados anteriores. Se recomienda ver como se han incorporado la nueva entidad y arquitectura a la librería **WORK.**

6) Preparación de la simulación

- ***Establecimiento de las condiciones de simulación***

Lo primero que hay que hacer es establecer las condiciones de simulación, para ello se despliega **Workspace** en la barra de herramientas y se selecciona **Settings,** o bien se activa el correspondiente botón (véase Figura-1), como en el caso de la compilación. Pero ahora en la ventana de dialogo que aparece se selecciona **Simulate** y se abre la carpeta **WORK.** Dentro de ella, aparece la entidad **AND2_TEST** y su arquitectura asociada **TEST_FLUJO**, así como la entidad **AND_2** y su arquitectura **AND2_FLUJO,** que es el componente que se desea testear. Hay que simular el **test***,* que es donde se ha configurado la entidad **AND_2** con la arquitectura **AND2_FLUJO,** y se han incluido las correspondientes señales de test. Se selecciona la arquitectura **AND2_TEST**, se pone el cursor sobre el espacio en blanco que acompaña a la palabra **Entity,** y a continuación se activa el botón de **Set.** Tras esta última acción, se observará que el nombre de la entidad seleccionada aparece copiado en el espacio en blanco. A continuación seleccionaremos la arquitectura **TEST_FLUJO** posicionando el cursor en el espacio en blanco identificado con la palabra ***Arch*** (abreviatura de arquitectura), y lo activaremos mediante el botón de **Set**, apareciendo el nombre de la arquitectura en el sitio donde habíamos posicionado el cursor. Y por último, activaremos la opción **Trace On** (esta selección nos va a permitir visualizar las formas de onda resultantes de la simulación). La visualización de las formas de onda también puede seleccionarse desplegando la función **Simulate** de la barra de herramientas y activando dentro de ella **Trace**.

- ***Activación del simulador***

La activación del simulador se puede hacer por dos procedimientos, desde **Workspace (Execute Simulator)** o bien desde la barra de herramientas activando el símbolo de activación del simulador (véase la Figura-1). Se recomienda tener activada la ventana de mensajes en **General,** en ella podremos leer un mensaje similar al siguiente:

```
vbvhdl4ive -d AND2_TEST TEST_FLUJO -tr on -r ns -i -dbg
VeriBest VHDL Simulator Version 15.00.00.25.
Starting Simulation ... Wed Oct 18 05:04:21 2000

License error: Error reading license file
```

```
Cannot find license file
 The license files (or server network addresses) attempted are
 listed below.  Use LM_LICENSE_FILE to use a different license file,
 or contact your software provider for a license file.
Feature:       VBVHDL_NT
Filename:      C:\flexlm\license.dat
License path:  C:\flexlm\license.dat
FLEXlm error:  -1,359.  System Error: 2 "No such file or directory"
For further information, refer to the FLEXlm End User Manual,
available at "www.globetrotter.com".
License file is C:\flexlm\license.dat
Feature is VBVHDL_NT.

------------------------------------------------------------
No license was found.
The simulator will run in reduced capacity mode.

If you would like a license for full capacity operation,
please contact us at sales@veribest.com
or in the USA call us at 1-888 482-3322.

http://www.veribest.com/vhdl.html
------------------------------------------------------------
```

que nos indica esencialmente, que tenemos una licencia de capacidad reducida de uso (hasta 2000 líneas de código), y que no encuentra una librería, **no hay que preocuparse**, activar **aceptar** y en ese momento obtendremos el siguiente mensaje:

```
NOTE: Number of component (cell) instances in the design: 1


Checkpointing at simulation time 0 ns.
Checkpoint completed.
Ready to simulate ... Wed Oct 18 05:12:21 2000

-- Message Summary:
Total: 0 error(s), 0 warning(s), 0 note(s)
```

Donde se puede observar que no hay errores y que se puede empezar la simulación.

- ***Ejecución de la simulación***

En la barra de herramientas (véase Figura-1)**,** aparece el símbolo convencional de **play** (▸), resaltado en verde. Y próximo a él, una ventana que pone 100 y ***ns***, esto indica el tiempo de ejecución total de la simulación y las unidades de tiempo asociadas. Ambos valores se pueden cambiar, estos son los valores por defecto. Para ejecutar la simulación hay que activar la tecla del símbolo del **play**. Obteniendo el siguiente mensaje:

```
run 100 ns

Run for 100 ns.
```

Otra opción para ejecutar la simulación es desplegar **Simulate** en la barra de herramientas y activar **Run**.

Si se desea desactivar el simulador se selecciona la función **quit** o el símbolo de parada en la barra de herramientas (véase Figura-1)**.**

7) *Visualización de los resultados de simulación*

Las formas de onda resultantes de la simulación se pueden analizar activando el botón de visualización de ondas en la barra de herramientas (véase la Figura-1). Aparecerá una ventana con el nombre **WaveForm Viewer**, ella contiene una escala de tiempos, una hoja con el fondo azul, y un botón con unas formas de onda, activar ese botón. Aparecerá una nueva ventana con el nombre de **Setup Signals**, y dentro de ella, la arquitectura realizada para el test y sus señales asociadas. Podemos entonces seleccionar las señales a visualizar de una en una **Add**, o bien todas de una vez **Add All,** observaremos entonces como las señales seleccionadas se incorporan a la ventana **WaveForm Viewer** con sus correspondientes formas de onda, tal como se muestra en la Figura-2.

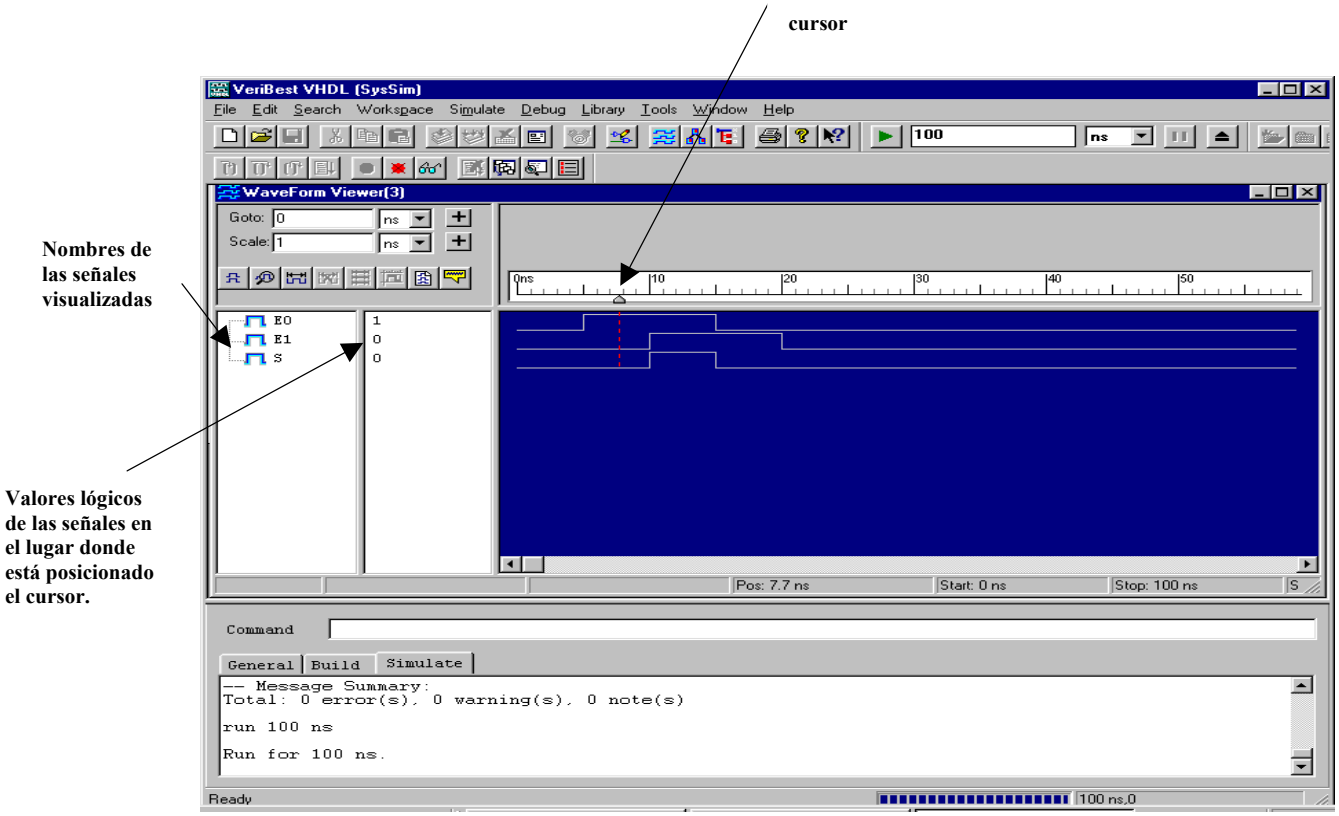


Figura-2. Formas de ondas resultantes de la simulación de una puerta AND

La impresión de las formas de onda que aparecen en la pantalla puede realizarse mediante las funciones especificas de la herramienta (**Icono impresora**, **File-Print**). Aunque en algunas configuraciones de algunas máquinas, las funciones de impresión presentan incompatibilidades con WINDOWS y abortan el simulador. En estos casos, se recomienda capturar la pantalla (Alt + Impresión Pantalla) y pegarla a un fichero de texto.

En la Figura-2, se puede observar la existencia de un cursor, que se puede desplazar por la regleta graduada en unidades de tiempo. Además de los nombres de las señales, y los valores lógicos de las mismas en el punto especifico donde está posicionado el cursor. Se recomienda mover el cursor para observar los cambios en los valores de las señales. Así como, experimentar con el zoom y medidas de tiempos entre dos puntos de las señales.