El VHDL EN LA ENSEÑANZA DE LA ELECTRONICA

Juana Mª López[1], Juan Carlos López[2]

[1]TGI. S.A.
Tecnología y Gestión de la Innovación
Dpto. de Electrónica
c/ Velazquez 134 bis 28006 Madrid

Tfno: 34 1 396 49 87
Fax: 34 1 396 48 32
e-mail: juana.lopez@sp1.y-net.es

[2]Universidad Politécnica de Madrid
E.T.S.I. Telecomunicación.
Dpto. Ingeniería Electrónica
Ciudad Universitaria s/n 28040 Madrid

Tfno: 34 1 336 73 01
Fax: 34 1 336 73 23
e-mail: lopez@die.upm.es

RESUMEN

En este artículo se presenta una herramienta informática de aplicación a la enseñanza de la electrónica, basada en la introducción de los lenguages de descripción de hardware, y en particular, del VHDL (VHSIC Hardware Description Language), en la docencia. Se trata pues de un tutorial de VHDL que conecta con un analizador y simulador comercial para ordenador personal y que incluye tanto la formación teórica como los ejemplos prácticos necesarios para su inclusión en los estudios Universitarios.

1. INTRODUCCION

Actualmente, la utilización de lenguajes de descripción de hardware para el diseño de circuitos electrónicos de cierta complejidad se está haciendo cada vez más necesaria, si bien, todavía está muy arraigada una metodología que no los incluye. Lo que podemos entender por "metodología clásica", basada en un proceso *top-down*, utiliza la descomposición del circuito global en bloques funcionales más sencillos, a fín de hacer más fácil abordar su diseño. Cuando la complejidad de dichos bloques se hace manejable, se utilizan tablas de verdad y ecuaciones lógicas para realizar su diseño.

Pero el uso de los lenguajes de descripción hardware permite abordar a nivel global un sistema complejo a partir de su especificación funcional, incluyendo importantes ventajas como son reducción del tiempo de diseño, documentación, reutilización de los diferentes

bloques funcionales..... Además, estos lenguajes son la puerta para la utilización de modernas herramientas de síntesis que logran una optimización automática de los diseños realizados. Sin embargo, los diseñadores ya habituados a otro tipo de herramientas y metodologías muestran cierta inercia que dificulta el cambio de mentalidad necesario para su utilización. Dicha inercia está mayormente motivada por la necesidad de dedicar un tiempo demasiado grande al aprendizaje de los HDL's antes de alcanzar con ellos la misma habilidad que ya se tienen con los métodos habituales. Además, dichos diseñadores no conocen suficientemente bien las facilidades y limitaciones de los lenguajes de descripción hardware como para confiarse y arriesgar dedicando un tiempo a ellos.

Por tanto, una enseñanza adecuada de las técnicas y metodologías de diseño basadas en lenguajes de descripción hardware aparece como una necesidad importante, tanto en el ámbito universitario para los que se inician en el diseño electrónico como en el industrial, para facilitar su adaptación a los ya expertos. En este documento se presenta una herramienta realizada con tal finalidad.

En el apartado 2 se presenta el VHDL y por qué se ha considerado como base de la enseñanza de la electrónica. El apartado 3 presenta un ejemplo sencillo de aplicación del VHDL al diseño de una puerta AND. El apartado 4 explica algunos nuevos conceptos introducidos por este lenguaje. Por último el apartado 5 muestra la herramienta tutorial realizada y sus principales características. En el apartado 6 se exponen las conclusiones.

2. El VHDL COMO BASE DE LA ENSEÑANZA DE LA ELECTRONICA

En aproximadamente treinta años de historia, han surgido gran cantidad de lenguajes de descripción hardware. Pero en esta evolución se ha tendido a que cada uno se centrase en un nivel de abstracción específico, cubriendo una determinada etapa en el proceso de diseño. Ahora bien, tras surgir el VHDL (VHSIC Hardware Description Language), si bien lo hizo orientado a la documentación, pronto fue detectada su potencia para otras tareas distintas y más relacionadas con el proceso de diseño (ver figura 1).

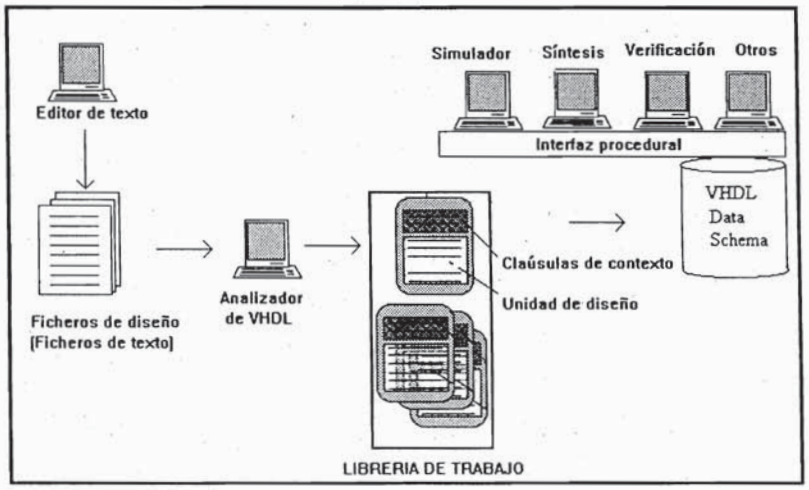


Figura 1. Proceso de diseño con VHDL.

El VHDL es un lenguaje legible tanto por el hombre como por una máquina, soportando el desarrollo, verificación, síntesis, test y mantenimiento de los diseños hardware[1]. Así mismo, introduce los conceptos de esenciales del diseño de sistemas electrónicos complejos, como son modularidad, jerarquía, evolución temporal, simulación, concurrencia. Es por todo ello que se muestra como base incomparable para la enseñanza de la electrónica, permitiendo al futuro ingeniero conocer todas las etapas involucradas en el diseño de un sistema complejo, y al diseñador ya experto el adaptarse a las nuevas tendencias en el proceso de concepción de tales sistemas.

En este documento se presenta una herramienta [2] basada en ordenador personal para el aprendizaje del VHDL, y, en general, de todos los conceptos involucrados en el diseño electrónico. Teniendo como base un conocimiento básico de electrónica digital, esta herramienta proporciona una importante ayuda, especialmente si tenemos en cuenta los principales problemas de la enseñanza universitaria, como son la cantidad limitada de tiempo de lectura y práctica y masificación y falta de recursos materiales y de profesorado. Por otra parte, considerando también las necesidades más importantes de la enseñanza de la electrónica (gran cantidad de ejemplos prácticos para la asimilación correcta de los conceptos, aprendizaje sencillo y atractivo que vaya aumentando en complejidad progresivamente), la utilización de una herramienta como la que posteriormente se describe se hace hasta cierto punto imprescindible.

3. UN EJEMPLO: DISEÑO DE UNA SIMPLE PUERTA AND.

Veamos como ejemplo de diseño con VHDL el modelado de una sencilla puerta "and" (ver figura 2). En este caso el dispositivo tiene dos entradas (e1 y e2) y una salida (s1). Obsérvese el código fuente que lo modela en la tabla 1 y que a continuación se explica.

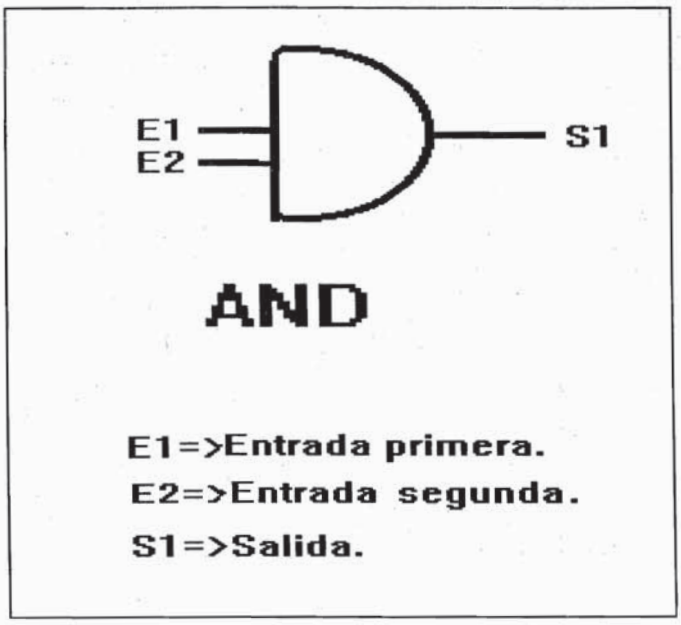


Figura 2. Dispositivo a modelar.

```
library IEEE;
USE IEEE.std_logic_1164.all;
ENTITY and_micro IS
 ------------------------------------------------
 --Declaración de los puertos
 --de entrada y de salida.
 ------------------------------------------------
 PORT (e1,e2 : in std_logic;
  s1 : out std_logic:='0');
END and_micro;
ARCHITECTURE funcion of and_micro is
BEGIN
        PROCESS (e1,e2)
        BEGIN
 ------------------------------------------------
 --Espera hasta que las entradas
 --cambien (lista de sensibilidad).
 ------------------------------------------------
                IF e1='1' and e2='1' THEN
                        s1 <='1' after 2 ns;
                ELSE s1 <='0' after 2 ns;
                END IF;
 ------------------------------------------------
 --Se modela con un retardo de 2 ns.
 ------------------------------------------------
        END process;
END funcion;
```

Tabla 1- Modelado de una puerta AND.

Como puede verse es modelado por una entidad (llamada "and_micro") y con un cuerpo de arquitectura (llamada "función"). Es una arquitectura "comportamental" pues no llama a entidades más pequeñas sino que contiene un algoritmo o comportamiento del dispositivo. El modelado de una simple puerta podría complicarse acercando más la simulación a la realidad mediante la introducción de chequeo de errores, retardos dependientes de la tecnología,....[5]

En las dos primeras líneas se llama al paquete std_logic_1164 que se encuentra en la librería IEEE. Esto es debido a que en este paquete se encuentra definido el tipo std_logic, que consta de varios valores posibles, muy útiles y que son ampliamente usados en el diseño de sistemas complejos. Es un tipo con nueve valores posibles, que son los que aparecen en la tabla 2.

'U'-Sin inicializar: usado como valor por defecto.
'X'-Desconocido (fuerte).
'0'-Cero (fuerte): transistor puesto a tierra.
'1'-Uno (fuerte): transistor puesto a alimentación.
'Z'-Alta impedancia.
'W'-Desconocido (débil).
'L'-Cero (débil): resistencias pull-down.
'H'-Uno (débil) :resistencias pull-up.
'-'-Tipo especial útil para síntesis

Tabla 2-Valores del tipo std_logic

Vemos que la entidad consta de un sólo proceso (Que comienza por la palabra "PROCESS"). Este proceso tiene una lista de sensibilidad (e1, e2). Esto quiere decir que el proceso se despierta cuando alguna entrada (e1 ó e2) cambia de valor. A continuación hay una instrucción condicional que dice que si las dos entradas valen la unidad, se realiza una asignación de '1' al puerto de salida trás un retardo de 2 ns. En caso contrario se asigna un cero a la salida. De esta forma el comportamiento es el de una puerta AND.

Una vez modelado, nos interesa saber si funciona correctamente y para eso es necesario simularlo. Para esta prueba se requiere otra entidad externa que le inyecte los estímulos a las entradas y observar sus salidas. Esta entidad se suele llamar en VHDL "test_bench". Es una entidad que no tiene entradas ni salidas, solamente señales internas que se conectan al dispositivo bajo estudio (en este caso, a la puerta AND). El código de esta entidad de test puede verse en la tabla 3 y se explica a continuación.

En la línea 6 se hace la declaración del componente AND que luego será instanciado una sola vez, en este caso. En la línea 10 se hace una especificación de configuración que indica que para la llamada a AND_MICRO que empieza con la etiqueta AAA, se usará la entidad que está en la librería de trabajo (WORK), que se llama AND_MICRO y se usa con la arquitectura llamada Función. En la línea 11 se declaran las señales que se atarán a los puertos de la puerta AND y que se llaman entrada1, entrada2 y salida. Se inicializan con el valor '0'. En la línea 13 ya comienza el primer proceso, que es la instanciación del componente AND_MICRO, con la etiqueta AAA. En la lista de asociación (PORT MAP) se nombran las señales "entrada1", "entrada2" y "salida".

```
library IEEE;
USE IEEE.std_logic_1164.all;
--------------------------------------------------------
--Entidad que chequea el circuito bajo estudio.
--------------------------------------------------------
ENTITY test_and is
END test_and;
ARCHITECTURE primera of test_and IS
--------------------------------------------------------
--Declaración del componente bajo estudio.
--------------------------------------------------------
COMPONENT AND_MICRO                --línea 6
PORT(e1,e2:in std_logic;
s1:out std_logic);
END COMPONENT;
--------------------------------------------------------
--Declaración de configuración. Línea 10
--------------------------------------------------------
FOR AAA:AND_MICRO USE ENTITY work.and_micr
--------------------------------------------------------
--Declaración de señales "atadas" a los puertos.
--Línea 11.
--------------------------------------------------------
- SIGNAL entrada1,entrada2,salida: std_logic:='0';
BEGIN
--------------------------------------------------------
--Primer proceso:el circuito bajo estudio. Línea 13.
--------------------------------------------------------
AAA:AND_MICRO port map(entrada1,entrada2,salida);
PROCESS
--------------------------------------------------------
--Segundo proceso:se aplican estímulos a las entradas.
--------------------------------------------------------
BEGIN
--------------------------------------------------------
--Se prueban los casos posibles.
--------------------------------------------------------
entrada1<='0';  --Línea 16
entrada2<='0';
WAIT FOR 4 ns;          --Línea 18
entrada2<='1';
WAIT FOR 4 ns;
entrada1<='1';
entrada2<='0';
WAIT FOR 4 ns;
 entrada2<='1';
WAIT;
END PROCESS;
END;
```

Tabla 3- Modelado del test_bench para la puerta AND.

A continuación comienza el segundo proceso, el que inyectará estímulos a las entradas. En la línea 16 se asigna un '0' a la entrada1 y mas tarde un '0' a la entrada2. Luego aparece una instrucción de espera con condición temporal. En ese momento el proceso se suspenderá hasta dentro de 4 ns, en que volverá a despertar. Mientras, le habrá dado tiempo a la puerta "and" a mostrar a su salida el valor que corresponda tras el retardo de 2 ns. Y esta operación se repite inyectando todos los casos posibles. Realizamos la simulación de esta entidad para ver si en las señales atadas a la salida se producen los resultados esperados. Para ello se compilan primero la entidad AND_MICRO y luego la entidad "test_and" y se pasa al simulador ésta. El resultado de listar las entradas y salidas se encuentra en la tabla 4.

ns	entrada1	entrada2	salida
0	0	0	0
4	0	1	0
8	1	0	0
12	1	1	0
14	1	1	1

Tabla 4- Resultado de la simulación.

En ella se puede observar que el funcionamiento corresponde a una puerta AND. El último resultado tarda 2 ns en ser correcto debido al retardo de la puerta. Este retardo siempre se cumple pero en los casos anteriores no se observa porque esta simulación solo lista los valores cuando hay un evento en alguna señal (y en los casos anteriores la salida no variaba de valor).

4. NUEVOS CONCEPTOS INTRODUCIDOS POR EL VHDL.

En uno de los primeros capítulos teóricos que presenta la herramienta tutorial se presentan los primeros conceptos necesarios para una buena comprensión del lenguaje, ya que el VHDL tiene unas características que le distinguen de otros muchos lenguajes de programación y de otros lenguajes de descripción de hardware. Además de incluirse una breve historia del VHDL, se presentan los conceptos de: mentalidad concurrente, drivers, niveles de abstracción, entidades y arquitecturas, etc.

El VHDL LRM [1] establece que la elaboración de un modelo VHDL crea una colección de procesos interconectados por redes, y ésta puede ser ejecutada para simular el comportamiento del diseño (ver figura 3).

Por tanto no es un lenguaje de programación que ejecute unas sentencias secuencialmente sino que está formado por una serie de procesos que se ejecutan concurrentemente de forma que simulan el comportamiento del sistema global modelado.

Cada proceso generado por el usuario tiene comienzo pero no fin, sólo se detiene al encontrar una sentencia de espera y suspenderse en cada ciclo de simulación. Además de los procesos generados por el usuario, siempre hay un proceso llamado kernel que representa al simulador

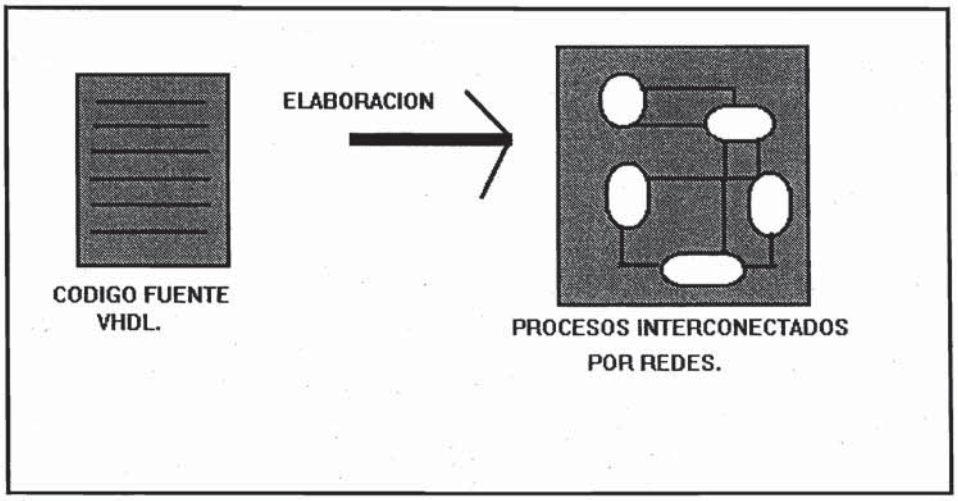


Figura 3. Modelo plano de procesos tras la elaboración.

y que se encarga del avance del tiempo, de las asignaciones a las señales y de la comprobación de cuáles de los procesos pueden despertarse en cada ciclo de simulación.

Así pues, el usuario genera una serie de procesos que contienen un conjunto de sentencias secuenciales y una o varias sentencias de espera que provocan que el proceso se "suspenda" y espere a que suceda la condición de espera para despertarse y continuar su ejecución.

Pero estos procesos no se comunican entre ellos mas que cuando están suspendidos todos. Y lo hacen a través de señales y por medio del Kernel (ver figura 4). Las asignaciones de valores a las señales sólo se producen cuando todos están suspendidos. Estas asignaciones producen el almacenamiento de un valor y un tiempo. Este tiempo indica para cuándo se quiere producir el cambio de valor. Así, para cada señal hay una cola de valores con sus instantes. Hay que distinguir entre ciclos de simulación y pasos de simulación. Para ello se ha de ver el modelo de tiempos del VHDL[3]. Este permite comunicación entre procesos sin que

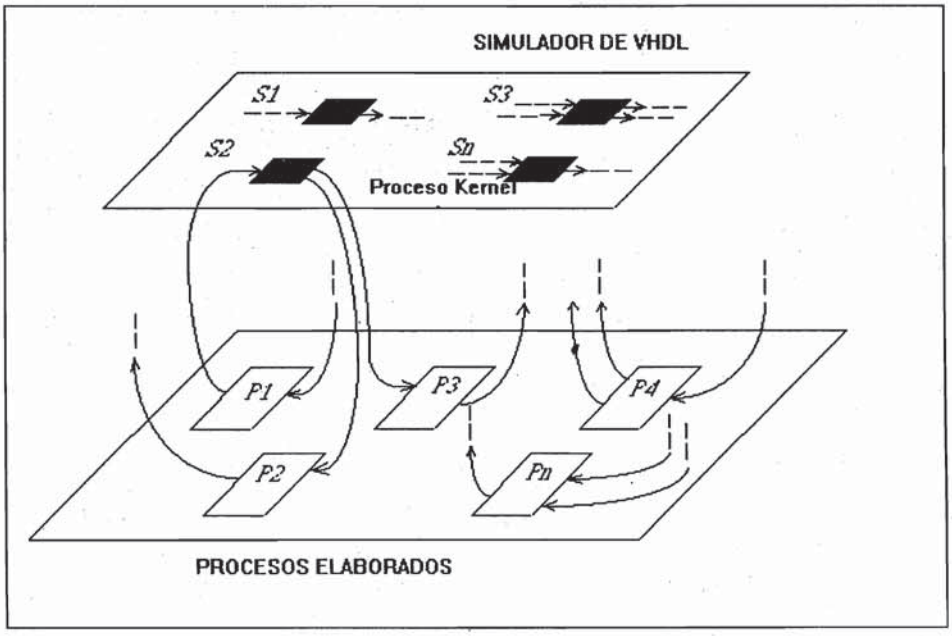
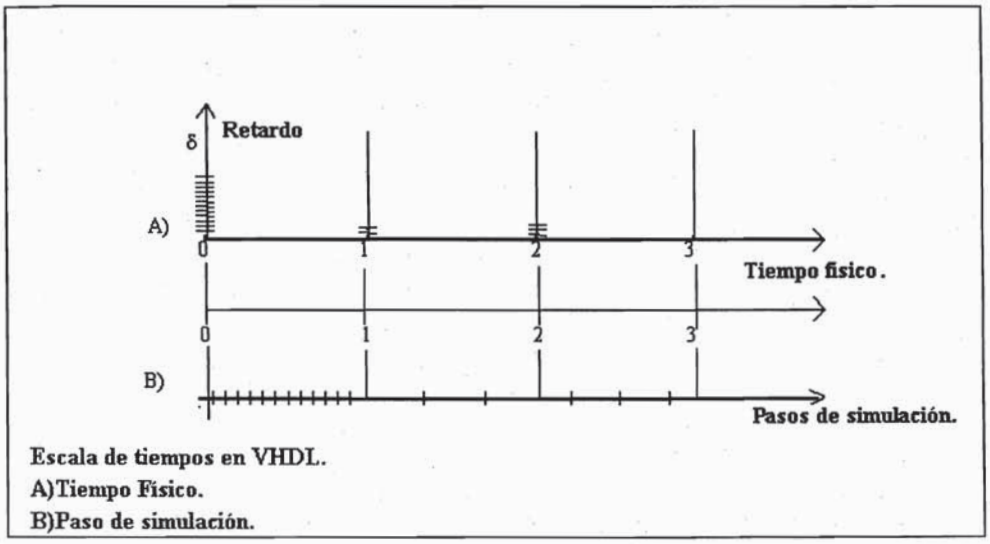


Figura 4. Procesos elaborados y proceso Kernel para Simulación.



Escala de tiempos en VHDL.
A)Tiempo Físico.
B)Paso de simulación.

Figura 5. Escala de tiempos en VHDL.

avance el tiempo físico sino que se incrementa otra variable que representa avances infinitesimales de tiempo: delta. Delta es una sub-unidad de tiempo que representa a un paso de simulación. El paso de simulación comienza cuando se despiertan los procesos que pueden y termina con las asignaciones de señal al estar todos suspendidos de nuevo.

Pero el tiempo físico no ha avanzado y pueden tener lugar varios de estos pasos de simulación en tiempo delta sin que avance el tiempo físico. Hay pues una macro-escala de tiempos (tiempo físico) y una micro-escala llamada delta (ver figura 5). En un mismo ciclo de simulación puede haber varias pasadas o pasos de simulación.

5.UN TUTORIAL DE VHDL PARA ORDENADOR PERSONAL.

Antes de realizar una herramienta para la enseñanza del VHDL, se analizaron los temas de que debe constar lo que consideramos una adecuada formación del diseñador. Identificamos dos requisitos para la educación Universitaria del diseño de sistemas electrónicos: enseñanza de una base teórica de los temas y entrenamiento práctico para resolver problemas concretos. Un programa de enseñanza efectivo se obtendría, por lo tanto, del balance apropiado entre ellos. Para solucionar estos aspectos se realizó una herramienta sencilla y atractiva, que disminuyese el tiempo de aprendizaje teórico y práctico, funcionando sobre un ordenador personal compatible IBM. El programa realizado consiste en un curso teórico y práctico de VHDL en el que se aprende a modelar y simular circuitos electrónicos digitales.

El flujo de diseño en VHDL comienza con el modelado. El modelado en VHDL consiste en la descripción de un circuito electrónico mediante este lenguaje de alto nivel y se realiza utilizando un editor de textos convencional creado para tal fin. Sobre esta descripción, se realiza un análisis y elaboración [4] para que pueda ser *ejecutada* a fin de comprobar que la funcionalidad del circuito, según su modelo, es correcta. Esta simulación implica la ejecución de los diferentes procesos con los que se ha modelado el comportamiento del circuito. A fín de que nuestra herramienta interactuase con un entorno de simulación real, se utilizó el

simulador comercial para PC V-SYSTEM/WINDOWS de Model Technology. El programa, en su conjunto, se ha realizado utilizando VISUAL BASIC. Se trata de un sistema de programación para crear aplicaciones para Windows. Permite crear la interfaz gráfica de usuario de una aplicación dibujando objetos de una forma gráfica. Luego se fijan una serie de propiedades en esos objetos para refinar su apariencia y comportamiento. Después se puede hacer a la aplicación responder a acciones del usuario escribiendo un código que responde a

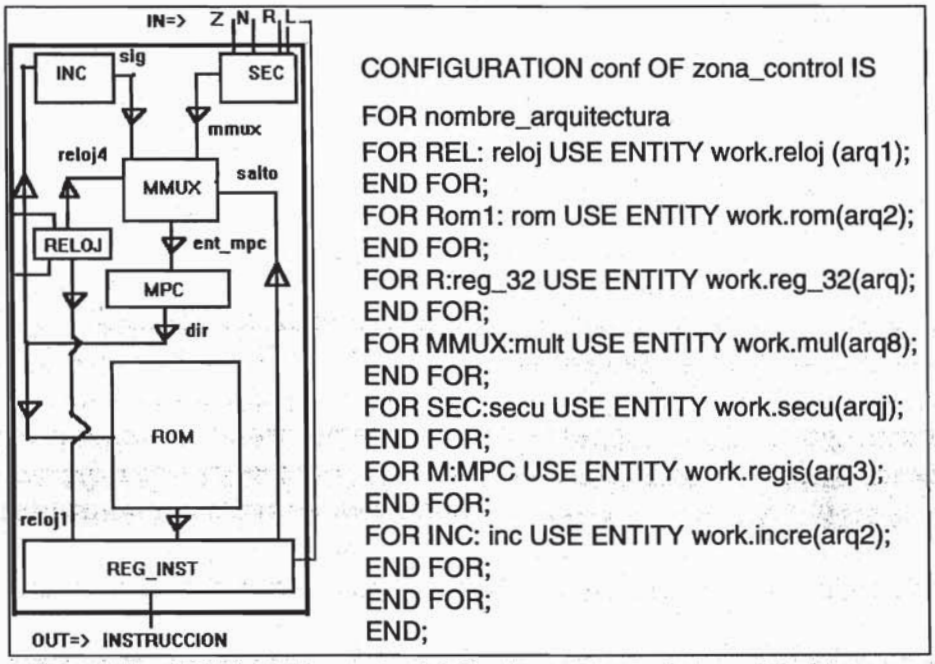


Figura 6-Zona de control del microprocesador modelado.

eventos que ocurran en el interfaz. La aplicación final que se crea es un auténtico fichero ejecutable dispuesto para su distribución.

El tutorial de VHDL realizado enseña el modelado y simulación de circuitos digitales mediante una serie de lecciones teóricas y prácticas que se van siguiendo mediante un interfaz de usuario gráfico muy práctico. De esta forma se van introduciendo conceptos, problemas y soluciones de forma progresiva, suponiendo una alternativa incomparable al aprendizaje a partir de manuales de un lenguaje tan complejo como el VHDL. En concreto, el tutorial consta de:

1. Dos lecciones teóricas, donde se introducen los conceptos mas novedosos (concurrencia, *delta-delays*) y se muestra la sintaxis (un resumen de las principales sentencias y reglas del VHDL)
2. Diez lecciones prácticas (ver figura 7), donde se van modelando paso a paso circuitos de menor a mayor complejidad: puertas lógicas, codificadores, selectores, ALU´s, registros, RAM, ROM,...hasta el control de un microprocesador, en el que se incluyen los diseños anteriores (ver figura 6). En cada una de estas lecciones prácticas se puede acceder interactivamente al simulador comercial, manteniendo ventanas de ayuda con los esquemáticos de los distintos circuitos y los comandos más importantes.
3. Una última lección donde se le facilita al alumno un editor de texto diseñado con conexión con el simulador para que cree sus propios diseños.

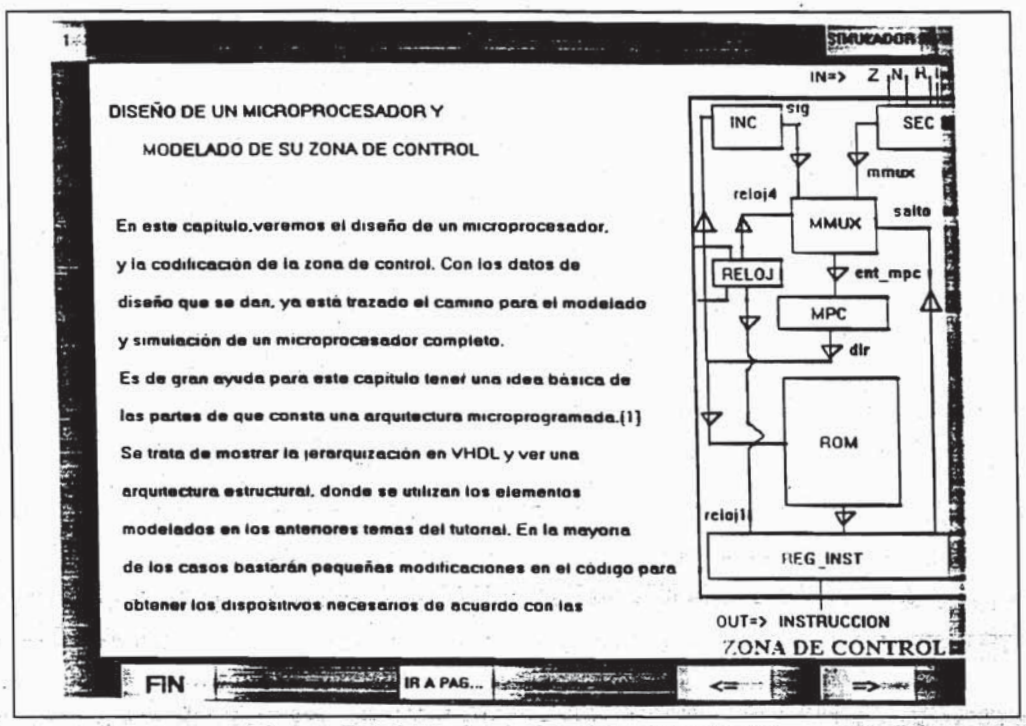


Figura 7- Ejemplo de lección del tutorial.

6. CONCLUSIONES.

Se ha presentado una herramienta de ayuda a la enseñanza de la electrónica, basada en la introducción del uso de los lenguajes de descripción de hardware, y, en particular, del VHDL, en la docencia. A través de ella, el alumno se podrá habituar a la utilización de las nuevas técnicas de diseño electrónico, introduciéndolo al modelado y simulación de circuitos electrónicos de complejidad creciente. De esta manera se logra, por otra parte, una enseñanza de la electrónica más adaptada a las más modernas tecnologías.

7. BIBLIOGRAFIA

[1]"1076-1987 Standard VHDL Language Reference Manual", IEEE Press, 1991.

[2] J.M. López, "Metodología para modelado y simulación de circuitos mediante lenguajes de descripción de hardware y su aplicación a la docencia", Proyecto Fin de carrera, ETSI Telecomunicación, Febrero 1994.

[3] S. Olcoz, J. M. Colom. "*VHDL Through the Looking Glass*" VHDL-FORUM for CAD in Europe,Insbruck, pp. 13-22. Spring 1993.

[4] J.López, G. Ricalde, A. García, L.Entrena, J.Goicolea, S. Olcoz. "*Integrating tools in a VHDL Framework*" VHDL International User's Forum. Spring 1994, Oakland, California.pp 154-162.

[5] D. Coello, "The VHDL handbook", Kluwer Academic Publishers, 1989.