APLICACIÓN DEL VHDL EN PRÁCTICAS DE DISEÑO DE SISTEMAS DIGITALES

A. Acosta[1], M. Bellido[2], M. Valencia[2], A. Barriga[1]

Facultad de Informática y Estadística
[1]Dpto. de Electrónica y Electromagnetismo
[2] Dpto. de Tecnología Electrónica

Centro Nacional de Microelectrónica, Avda. Reina Mercedes s/n
Universidad de Sevilla, 41012 Sevilla

Tfno: 95-4239923
Fax: 95-4624506
E-mail: acojim@cnm.us.es

RESUMEN

Se describe un conjunto de prácticas para la docencia de los sistemas digitales enfocadas al diseño y simulación de un procesador simple. Estas prácticas están basadas en el uso del VHDL como lenguaje de descripción. Se discute tanto la realización de las prácticas como la aplicación del VHDL en la docencia de los sistemas digitales.

1. INTRODUCCIÓN

La docencia práctica en el diseño de sistemas digitales de complejidad media presenta fuertes inconvenientes en nuestros centros de enseñanza. A la propia complejidad de la materia se une el alto coste de los recursos necesarios para hacer prácticas de diseño reales, tanto en material de laboratorio (instrumentación, hardware) como en tiempo. Por otra parte, al menos los alumnos de 2º ciclo de Ingenierías tales como Informática, Electrónica y Telecomunicaciones, deben adquirir dominio sobre las herramientas de ayuda al diseño por computador (CAD/CAEE) así como de la propia metodología de diseño.

El propósito de esta comunicación es presentar el sistema de prácticas, que hemos desarrollado y venimos realizando en 5º curso de Informática, el cual cubre conjuntamente los objetivos docentes anteriores. Más concretamente, el objetivo de las prácticas es diseñar un sistema digital complejo (i.e. un procesador), hacerlo a nivel práctico (usando CAD/CAEE) y utilizando un

lenguaje con gran interés en sí mismo (como es VHDL). El entorno que se requiere está constituido por un ordenador junto a la herramienta software que recibe la descripción y realiza la simulación.

La utilidad del uso de los lenguajes de descripción de hardware viene dada, por un lado, por el bajo coste del material de laboratorio. Por otro lado, los lenguajes de descripción de hardware permiten la descripción y simulación de sistemas digitales con un cierto nivel de complejidad, lo cual sería inabordable desde el punto de vista experimental. Por lo tanto, uniendo el bajo coste, en lo que se refiere a material de laboratorio, a su alto contenido pedagógico, al ilustrar conceptos que de otra forma no sería posible debido a la propia limitación de recursos, hacen que estos lenguajes constituyan una herramienta de gran ayuda para impartir esas disciplinas.

La metodología empleada debe cubrir la descripción del sistema aplicando técnicas de diseño jerarquizado. Parte de las especificaciones del sistema para realizar un estudio de las soluciones arquitecturales. Fruto de esta tarea se obtiene la descripción del procesador como una caja negra de la cual se especifican los puertos de entrada/salida. El siguiente paso consiste en la partición de la arquitectura en módulos funcionales, así como en la determinación de las celdas que configuran los módulos. También en esta tarea se especifica la temporización y sincronización de las operaciones que realizan los módulos. Finalmente, la última parte del proceso consiste en el diseño de las celdas básicas, así como en el análisis del comportamiento de los bloques.

Nuestra ponencia la vamos ha organizar en los siguientes apartados. En el siguiente presentaremos muy brevemente algunos conceptos del lenguaje VHDL. A continuación discutiremos las especificaciones de nuestro sistema. Finalmente, describiremos la realización de las prácticas.

2. VHDL

La justificación en escoger el VHDL como lenguaje que soporta nuestras aplicaciones está avalada por tratarse de un estándar (IEEE Std 1076) [1]. Este hecho ha motivado que sea el lenguaje de descripción de hardware que actualmente más difusión esté alcanzando. También sirve de criterio de elección su versatilidad ya que permite distintos estilos de descripción, cubriendo, por lo tanto, los diferentes niveles de descripción de los sistemas digitales (nivel de arquitectura, algorítmico, RT y lógico).

El desarrollo del VHDL (VHSIC Hardware Description Language) comenzó en 1981 con un programa del Departamento de Defensa de EEUU (VHSIC: *Very High Speed Integrated Circuits*) [2]. Desde que en 1987 fue estandarizado por el Grupo de Análisis y Estandarización del VHDL, constituido por IEEE/CS/DATC/DASS/VASG, se ha extendido su uso en el proceso de diseño de manera creciente [3-5]. De hecho, hoy en día, la mayoría de los entornos de diseño tienden a reconocer descripciones en VHDL. Así pues, las empresas de software están realizando fuertes inversiones en adaptar sus productos a este lenguaje y a desarrollar nuevos productos que lo utilicen como plataforma de descripción.

VHDL proporciona gran ayuda en el proceso de diseño de un sistema digital. Permite describir la estructura del sistema, especificando su descomposición en subsistemas y la interconexión de estos últimos. Además de este estilo de descripción, que llamaremos descripción estructural, la gran ventaja de este lenguaje es que permite otros estilos en los que se describe el comportamiento funcional del sistema. Así, una segunda visión del sistema permite representarlo como una transformación de datos de entrada en datos de salida. Una tercera visión del sistema representa el comportamiento algorítmico de la máquina que se describe. En este sentido, VHDL permite describir algoritmos mediante instrucciones secuenciales típicas de un lenguaje de programación de alto nivel, mediante el uso de procesos, procedimientos y funciones.

El diseño de un sistema en VHDL consiste en una jerarquía de bloques conectadas entre sí. Cada uno de estos bloques constituye una unidad de diseño que consiste en una entidad (descripción del conexionado externo) y una arquitectura (descripción de la operación del circuito). Por lo tanto, la descripción de un bloque puede tener varias arquitecturas con la misma funcionalidad ya que la sustitución de una arquitectura por otra no tendrá efectos externos. Sin embargo, sólo debe tener una entidad única.

Para especificar la organización y operación de un diseño se utilizan declaraciones concurrentes y/o secuenciales. Las declaraciones secuenciales especifican algoritmos y están contenidas en procesos y subprogramas para utilizarse luego en un contexto concurrente. El tipo de instrucción es similar a las de un lenguaje de programación de alto nivel ("*if*", "*case*", "*while*", llamadas y retornos a/de procedimientos, asignaciones de variables), así como otras específicas (asignamiento de señales, declaraciones de espera, "*wait*"). Las declaraciones concurrentes especifican interconexión de componentes, estructura jerárquica, estructura regular y flujo de datos u operaciones de transferencia de registros.

Nuestro sistema de prácticas ha sido diseñado para aprovechar la flexibilidad que ofrece VHDL de mezclar la descripción estructural junto con la funcional.

3. DESCRIPCIÓN DEL SISTEMA

El objetivo de estas prácticas es diseñar un procesador básico constituido por un conjunto de registros y una ALU. El esquema del sistema se muestra en la figura 1. El procesador dispone de un bus de direcciones de 8 bits capaz de direccionar una memoria de 256 palabras y un bus de datos de 12 bits. La memoria se activa en modo de lectura o escritura con la señal $R/\overline{W}$. El

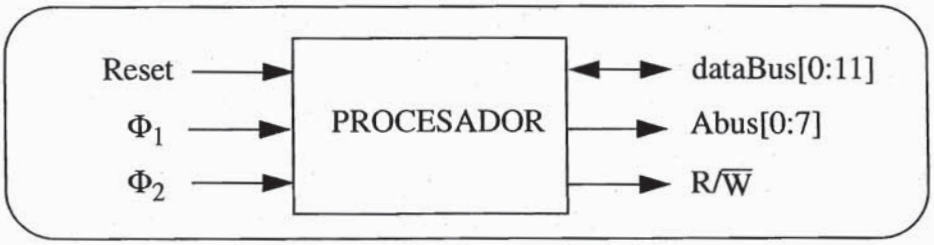

Figura 1. Estructura del procesador.

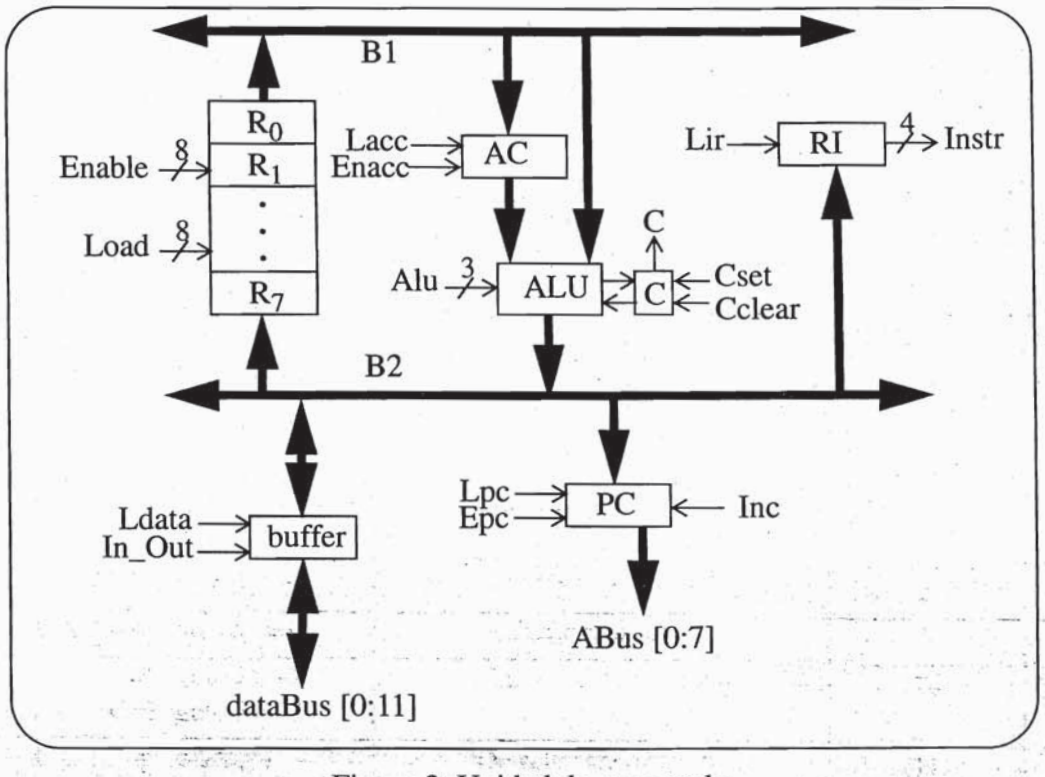

Figura 2. Unidad de procesado.

sistema opera con un esquema de reloj de dos fases no solapadas Φ_1 y Φ_2. Finalmente, existe una señal de *Reset* que permite inicializar al procesador. Observamos que por simplicidad no hemos considerado otras funcionalidades que complicarían el diseño y requerirían una dedicación en tiempo mayor que la inicialmente prevista para las prácticas.

La estructura de la unidad de procesado se ilustra en la figura 2. Está constituida por un conjunto de 8 registros de almacenamiento de datos de 12 bits, un acumulador y una ALU capaz de realizar operaciones lógicas, aritméticas y de desplazamiento. La descripción de la ALU se muestra en la figura 3. Observamos que recibe un vector de tres señales de control denominado *Alu* que especifica la operación a realizar.

La red de interconexión interna está basada en dos buses, uno para lectura de los registros (*B1*) y otro de escritura (*B2*). El sistema está temporizado mediante un reloj de dos fases no solapadas Φ_1 y Φ_2. Así, los registros ($R_{0:7}$) tienen una estructura maestro-esclavo, con salidas de tres estados. Esto permite separar las operaciones de lectura (*Enable*) de las de escritura (*Load*).

La unidad de procesado también dispone de dos registros adicionales que son: un registro de instrucciones (RI) y un contador de programa (PC). El registro de instrucciones recibe datos del bus de datos *B2* y suministra sus salidas a la unidad de control. Por su parte el registro contador de programa actúa sobre el bus de direcciones y recibe datos a través del bus de datos *B2*.

La unidad de procesado de la figura 2 opera bajo la supervisión de la unidad de control del procesador. Esta última se encarga de generar las señales de control para la unidad de procesado así como las señales de control externas. La unidad de control permite ejecutar el conjunto de instrucciones del procesador. En el siguiente apartado discutiremos y presentaremos cual será el juego de instrucciones de nuestro sistema.

4. REALIZACIÓN DE LAS PRÁCTICAS

El diseño y simulación del sistema completo está organizado en tres sesiones de prácticas con una duración de tres horas para cada sesión. Un primer paso en nuestro proceso de diseño consiste en describir la unidad de procesado. A continuación se desarrolla la unidad de control que permita ejecutar un conjunto de instrucciones (reducido en nuestro caso). Finalmente, se ensambla el sistema completo incluyendo los registros de instrucción y de contador de programa. Cada etapa del proceso supone la descripción de un circuito y su simulación con un doble objetivo: verificar el correcto comportamiento del circuito y estudiar la aplicación de los vectores de test.

Con esta estructura, cada sesión de prácticas es autocontenida y, el conjunto de las tres, cubre los objetivos indicados completamente. Para la realización de las prácticas se requiere que el alumno conozca tanto el lenguaje (VHDL) como la herramienta que va ha utilizar. Para adquirir soltura en el manejo de la herramienta es útil realizar una práctica previa, en la que se especifique un circuito simple, de manera que le permita al alumno recorrer de forma guiada el proceso de trabajo. Por otro lado, si bien los alumnos deben tener los conocimientos suficientes para afrontar estas prácticas (conocimientos sobre arquitectura de ordenadores), conviene hacer una discusión previa. Esta discusión se establece resolviendo los problemas que se plantean como clases de problemas. Además, la elaboración del sistema debe estar previamente meditada por el alumno como paso previo a afrontar el diseño en el puesto de trabajo.

En lo que sigue, y por razones de espacio, vamos a explicar muy brevemente los contenidos de

```
-- Operaciones de la ALU
case Alu is
    when "000" => Out_alu := In1_alu + In2_alu;
    when "001" => Out_alu := In1_alu - In2_alu;
    when "010" => Out_alu := In1_alu nand In2_alu;
    when "011" => Out_alu := In1_alu nor In2_alu;
    when "100" => Out_alu := sh_left(In2_alu,CF);
    when "101" => Out_alu := sh_right(In2_alu,CF);
    when "110" => Out_alu := In2_alu;
    when "111" => Out_alu := In1_alu;
end case;
```

Figura 3. Descripción de la ALU.

```
entity procesador is
   port(dataBusIn:  in  wire_vector (0 to 11);    -- Bus de entrada de datos
        dataBusOut: out wire_vector (0 to 11);    -- Bus de salida de datos
        Abus:       out wire_vector(0 to 7);      -- Bus de direcciones
        RW:         out bit;                      -- Lectura/escritura
        Reset:      in  bit                       -- Reset
        Phi1,Phi2:  in  bit);                     -- Reloj de dos fases no solapadas
end procesador;
```

Figura 4. Entidad del procesador.

las prácticas. Básicamente nos centraremos en completar las especificaciones del sistema e indicar las tareas que se realizan en cada práctica.

Práctica 1: Descripción de la ruta de datos.

El primer paso consiste en describir la entidad y arquitectura del procesador. La figura 4 muestra la entidad del procesador. Dicha entidad consiste en una descripción de los puertos de entrada y salida. La entidad tiene asociada una determinada arquitectura que representa su funcionalidad. Así, en la figura 5 se muestra una parte de la descripción de la arquitectura del procesador que corresponde a las operaciones de escritura y lectura de los registros. Observamos que el estilo de descripción representa el comportamiento del sistema, para lo cual se emplean instrucciones secuenciales encapsuladas en procesos.

Una vez descrito el sistema el siguiente paso consiste en simular su comportamiento. Para ello

```
-- Escritura en un registro
if Load/=cero and Phi2='1' then
   for i in 0 to 7 loop
      if Load(i)='1' then
         Mreg(i) := B2 after TpR;
      end if;
   end loop;
end if;
if Phi1='1' then
   for i in 0 to 7 loop
      Sreg(i) := Mreg(i) after TpR;
   end loop;
end if;
```

a)

```
-- Lectura de un registro.
if Enable/=cero then
   for i in 0 to 7 loop
      if Enable(i)='1' then
         B1 <= Sreg(i) after TpB;
      end if;
   end loop;
end if;
```

b)

Figura 5. a) Descripción de la escritura en un registro. b) Descripción de la lectura de un registro.

CO	Mnemónico	Operación	Tipo de Operación
0	LDA	AC←R_i, i=0,...,7	Transferencia: cargar en AC
1	STA	R_i←AC, i=0,...,7	Transferencia: almacenar en Ri
2	LDR	R_i←M, i=0,...,7	Transferencia: cargar en Ri
3	STR	M←R_i, i=0,...,7	Transferencia: almacenar en M
4	ADD	AC←AC+R_i, i=0,...,7	Aritmética: suma
5	SUB	AC←AC-R_i, i=0,...,7	Aritmética: resta
6	ROL	shL(AC,C), C←AC_{11}	Lógica: desplazamiento a la izquierda (con acarreo)
7	ROR	shR(C,AC), C←AC_0	Lógica: desplazamiento a la derecha (con acarreo)
8	NAND	AC←AC nand R_i, i=0,...,7	Lógica: nand
9	NOR	AC←AC nor R_i, i=0,...,7	Lógica: nor
10	BCS	C=0: goto PC+1 C=1: goto PC+2	De salto: condicional
11	JMP	goto $\$M_{0-7}$	De salto: incondicional
12	JMPI	goto $\$[\$M]_{0-7}$	De salto: incondicional indirecto
13	CLC	C←0	De estado: borrar acarreo
14	SEC	C←1	De estado: poner 1 en acarreo
15	HLT	--------	De control: parada

Tabla 1. Juego de instrucciones del procesador.

se especifica en VHDL el esquema de testado del procesador. El procedimiento seguido se basa en una descripción en la cual se coloca el procesador como un componente junto con un proceso que suministre los valores de las señales de entrada.

Práctica 2: Unidad de control.

El objetivo de esta práctica es describir la unidad de control del procesador. Para ello es necesario identificar las señales de control y describir el conjunto de instrucciones del procesador. Vamos a considerar que el procesador cuenta con el conjunto de instrucciones que se muestra en la tabla 1.

En una primera aproximación al diseño de la unidad de control no nos planteamos la realización del ciclo de búsqueda de instrucciones con objeto de centrarnos en la realización del juego de instrucciones y establecer las interacciones entre el controlador y la unidad de procesado. Partimos de considerar que el sistema dispone de un registro (RI) que ya contiene el código de la instrucción que se va a ejecutar.

```
-- INSTRUCCION STA
wait on Phi1 until Phi1='1';
EnAcc <= '1'                          -- Habilitar la salida del acumulador
Alu   <= "111";                       -- Habilitar la ALU para pasar dato
NumReg := to_integer(RI(5 to 7));
Load(NumReg) <= '1';                  -- Habilitar el registro destino
```

Figura 6. Instrucción de carga de un registro.

El primer paso consiste en describir en VHDL la entidad y arquitectura de la unidad de control. Para ello usaremos de nuevo el estilo de descripción de comportamiento. La unidad de control recibe las señales de reloj Φ_1 y Φ_2 y genera una microinstrucción en cada ciclo de reloj.

A continuación pasamos a describir el fichero de control de la simulación y simular la unidad de control. La figura 6 muestra un ejemplo de la descripción de la instrucción de carga de un registro. La especificación del registro viene dada por los bits 5 al 7 del registro de instrucción, mientras que el código de la instrucción se especifica por los cuatro bits menos significativos.

Practica 3: Descripción del sistema completo.

Se trata de describir el fichero de simulación y simular el sistema completo que está constituido por la unidad de control y el procesador. En esta práctica ya se incluyen los registros de instrucción y contador de programa y se realiza la colocación de los componentes unidad de procesado y unidad de control. Se incluye además en la unidad de control el ciclo de búsqueda de instrucción.

El fichero de control de la simulación describe un pequeño programa que permita la ejecución de instrucciones por el procesador. Básicamente este fichero se apoya en un proceso que actúa sobre las señales del procesador y simula el comportamiento de los componentes externos al mismo (como por ejemplo, la memoria principal o el bus del sistema).

5. CONCLUSIONES

Se ha discutido la necesidad de aplicar entornos de herramientas CAD/CAEE en la docencia de las materias de los sistemas digitales. Para ello resulta muy útil el uso de lenguajes de descripción hardware ya que permiten afrontar el estudio de sistemas complejos con un bajo coste. Para mostrar esta utilidad hemos presentado un conjunto de experimentos de laboratorio que permiten al alumno enfrentarse a las tareas de diseño y verificación de un sistema de complejidad media como es un procesador. Los resultados observados de la realización de las

prácticas nos llevan a concluir que realmente se profundiza en el nivel de aprendizaje del alumno, ya que éste debe afrontar y resolver los problemas que el diseño de un sistema de este tipo presenta. A su vez hemos de notar que el alumno se encuentra altamente motivado al experimentar lo que ya conoce sobre el papel.

6. BIBLIOGRAFÍA

[1] IEEE Standard VHDL Language Reference Manual. 1987

[2] J.P. Mermet: "Fundamentals and Standards in Hardware Description Languages". Kluwer Academic Pub., 1993.

[3] R. Lipsett, C. Schaefer, C. Ussery: "VHDL: Hardware Description and Design". Kluwer Academic Pub., 1990.

[4] L.M. Augustin, D.C. Luckham, B.A. Gennart, Y. Huh, A.G. Stanculescu: "Hardware Design and Simulation in VAL/VHDL". Kluwer Academic Pub., 1991.

[5] L.J.M. Bergé, A.Fonkoua, S.Maginot, J. Rouillard: "VHDL Designer's Reference" Kluwer Academic Pub., 1992.