12º Congreso Nacional de Mecatrónica, ISBN 978-607-95347-8-3
Octubre 10 y 11, 2013. León, Guanajuato.

Diseño de un software para programación de Dispositivos Lógicos Programables basado en Bloques Descriptivos de Hardware.

Méndez Sánchez Kikey Stephanie, Vázquez Guerrero Mónica, Juárez Buenrostro, Marco Antonio Aceves Fernández, Juan Manuel Ramos Arreguín

Universidad Autónoma de Querétaro, Facultad de Informática.

Resumen

Una de las grandes ventajas al diseñar sistemas digitales mediante dispositivos lógicos programables radica en el bajo costo de los recursos requeridos para el desarrollo de aplicaciones. Hoy en día existen en el mercado herramientas de diseño del lenguaje VHDL, sin embargo, el software SHDL 1.0 desarrollado, tiene un gran impacto en el área del desarrollo tecnológico, competente a las herramientas que ofrecen diversas compañías internacionales. La funcionalidad del software radica en el desarrollo de aplicaciones basado en descripción de lenguajes de hardware, y utilizando diagramas a bloques, permitiendo el diseño e implementación de aplicaciones de hardware en menos tiempo. El software permite también realizar la documentación del trabajo de una manera directa. El uso de este Software tiene un gran impacto en las áreas de diseño, investigación, desarrollo tecnológico y enseñanza académica, para dar solución a problemas reales. El software tiene un alcance de a partir de un Diagrama de Bloques Descriptivos de Hardware, genera el código correspondiente en VHDL basado en el estándar 1164 de la IEEE, sim embargo no cuenta con las herramientas que permitan simular y compilar el código.

Palabras clave: PLD's, Software, Bloques Descriptivos de Hardware, VHDL, Java, HDL

1. Introducción

En la actualidad, el nivel de integración alcanzado con el desarrollo de la microelectrónica permite desarrollar sistemas completos dentro de un solo circuito integrado SOC (System On Chip), con lo cual se han mejorado características como lo son velocidad, confiabilidad, consumo de potencia y tamaño de diseño. El desarrollo que ha habido en los últimos años en el desarrollo de Diseño, ha permitido que las nuevas tecnologías sean implementadas en un espacio menor, lo cual se observa en aplicaciones militares, industriales, y de consumo en las cuales el tamaño es considerablemente menor a comparación de los diseños de hace varios años. Los teléfonos celulares, computadoras personales, agendas relojes digitales sistemas de audio, sistemas de telecomunicaciones entre otros dispositivos en los cuales se ve la evolución de los circuitos integrados también conocidos como chips. Los chips son programados mediante Lenguaje Descriptivo de Hardware (HDL) haciendo posible programar desde un diagrama a bloques nivel básico como una compuerta, hasta los más elaborados como las máquinas secuenciales.

Es importante definir conceptos clave mencionados a lo largo del artículo. Dichos conceptos son presentados a continuación:

1.1 Dispositivos Lógicos Programables (PLD).

Son dispositivos con característica de conectividad programable, mediante el firmware desarrollado por los fabricantes. Los PLD tienen una arquitectura interna regular y flexible, y pueden ser programados mediante una estructura de interruptores [1].

Existe una gran diversidad de empresas que han desarrollado sus propias herramientas de desarrollo para PLD's, como lo son Altera Corporation, Aldec Inc., Xilinx Inc, entre otros, y se

ponen a disposición de los desarrolladores de sistemas digitales.

1.2 Sistema Embebido.

Un Sistema Embebido es un sistema de cómputo "empotrado" en un componente o producto, diseñado generalmente para realizar una tarea específica, a menudo con restricciones de tiempo real.

Un ejemplo de Sistema Embebido son las cámaras inteligentes, donde la cámara no sólo captura la imagen, sino que también procesa para extraer información como es requerido por la aplicación [2].

1.3 Lógica Programable.

La lógica programable representa la funcionalidad de un circuito, donde un sistema digital particular puede ser programado a partir de las funciones lógicas necesarias, para cumplir con los requisitos de una aplicación. La funcionalidad se implementa como un sistema en paralelo, en lugar de un sistema secuencial. Trabajan con una lógica de dos niveles, como pueden ser circuitos ROM (Read Only Memory), circuitos PAL (Programable Array Logic) y PLA (Programable Logic Array) [2].

1.4 Lenguaje Descriptivo de Hardware.

El Lenguaje Descriptivo de Hardware (HDL) describe el comportamiento de un sistema digital, el cual es implementado en Hardware en un PLD, para implementar las funciones lógicas necesarias para resolver un problema específico. Debido a que es hardware, todo lo que se implementa en HDL se ejecuta en paralelo, es decir, al mismo tiempo. Los HDL más utilizados son el Lenguaje Descriptivo de Hardware para Dispositivos de Muy alta velocidad (VHDL) y Verilog.

El VHDL fue diseñado para el desarrollo de sistemas digitales de tipo combinacional y/o secuencial, abarcando el desarrollo de la Unidad Central de Procesamiento (CPU), utilizado en el diseño de los controladores. El lenguaje VHDL se convirtió en estándar de la IEEE en 1987 (std 1076-1987) [3].

1.5 Bloques Descriptivos de Hardware (BDH).

Se refiere al diseño gráfico de los bloques descriptivos de hardware (BDH) que se implementarán en el software. Se basa en el símbolo con el que éstos los identifica [3].

1.6 Diagrama a Bloques.

Un diagrama a bloques es una representación descriptiva gráfica de los elementos que contiene un sistema, es común emplear diagramas a bloques debido a la simplicidad de análisis del sistema [4].

1.7 Field Programmable Gate Array, FPGA.

Los dispositivos FPGA son circuitos lógicos programables directamente por el usuario, para lo cual se requiere de herramientas de desarrollo. Son circuitos de silicón reprogramables muy utilizados por fabricantes que producen tecnología a baja escala. Al utilizar bloques de lógica pre-construidos y recursos de ruteo programables, el FPGA se puede configurar para implementar funcionalidades de hardware personalizadas. Estos dispositivos se basan en arreglos de compuertas teniendo una arquitectura de tres elementos configurables: Bloques de Lógicos Configurables (CLB), Bloques de Entrada y Salida (IOB) y canales de comunicación [1]

2. Estado del Arte

En la actualidad, en el mercado existen múltiples herramientas de diseño de aplicaciones de hardware, y existen compañías internacionales que fabrican PLD's, así como herramientas de diseño EDA (Electronic Design Automation) para la síntesis, implementación y simulación de circuitos digitales. A continuación se presenta una relación de algunos fabricantes, así como sus principales herramientas de desarrollo [5].

2.1 Altera Corporation.

La capacidad de integración de las familias de Altera varía desde 300 hasta un millón de compuertas utilizables por dispositivo, además de que todas tienen la capacidad de integrar sistemas complejos. Cuenta con la herramienta MAX + PLUS II que soporta toda la familia de dispositivos Altera, así como el Software estándar compatible con VHDL. Ofrece al mercado el software **Quartus** que apoya el desarrollo de sistemas complejos a través de un conjunto de todas las funciones. Los entornos de diseño de alto nivel que incluyen, se basan en C. Se tiene la flexibilidad para cambiar la funcionalidad básica sobre la marcha. La compra del Software incluye subscripción a actualizaciones por un año por el precio de 2,995 USD [w1].

2.2 Aldec.

Aldec Inc es una propiedad privada de automatización de diseño electrónico. Ofrece el software y hardware utilizados en la creación y verificación de diseños digitales con FPGA y tecnología ASIC. Ofrece el entorno de desarrollo **Active–HDL** que es una creación integrada para el diseño y simulación de lenguajes descriptivos de hardware VHDL y Verilog. Cuenta con interfaz MATLAB/Simulink, aunque el código generado está basado en librerías no estándar. El precio de la licencia de Active-HDL es de 2,495 USD por un año. Cuenta con un tipo de licencia para Estudiantes que ofrece sin costo a los estudiantes el uso del software necesario para su trabajo de curso [w2].

2.3 Xilinx.

Es una de las compañías líder en soluciones de lógica programable, incluyendo circuitos integrados avanzados, herramientas de software para diseño, funciones predefinidas y soporte de ingeniería. Creadora de FPGA´s, cuenta con herramientas importantes como es Foundation Series que soporta diseños estándares basados en ABEL-HDL y en VHDL. Esta herramienta se ofrece en versión estudiantil y profesional. Y la segunda pero no menos importante es **ISE Design Suite** que proporciona un conjunto completo de entorno de desarrollo integrado, herramientas de software, asistentes de configuración e IP que facilita el diseño y utiliza toda la flexibilidad que ofrece una plataforma programable. Recientemente Xilinx liberó una Suite de Diseño llamada **Vivado**, que soporta flujos jerárquicos de diseño para el diseño con la reutilización de código y reconfiguración parcial, con un tiempo de ejecución hasta 4 veces más rápido. Su costo es de 2,995 USD [w3]. Es posible obtener una versión gratuita.

3. Justificación

Aunque en la actualidad existen herramientas de diseño de lenguaje VHDL, la herramienta de software desarrollada en la Universidad Autónoma de Querétaro, a la cual se le ha dado el nombre de SHDL (Sistema de Lenguaje Descriptivo de Hardware) versión 1.0, permite el desarrollo de descripciones de hardware de una manera gráfica y en menos tiempo, impactando en áreas de diseño, investigación y académicas.

Un diseño digital puede ser creado en cualquier editor que cuente con símbolos gráficos de circuitos o usando un lenguaje de descripción de Hardware, sin embargo, en la mayoría de los casos está limitado a la compra de licencias o el uso de librerías específicas del fabricante, por lo que el desarrollo pierde portabilidad al funcionar exclusivamente para los dispositivos del fabricante [3].

La herramienta SHDL 1.0 ofrece algo diferente, permitiendo generar, a partir de un diagrama a bloques descriptivo de hardware diseñado por el usuario, el código correspondiente en el lenguaje VHDL, ahorrando tiempo y realizando una documentación de manera directa. El código generado puedes ser utilizado en cualquier software de simulación del lenguaje VHDL, por ejemplo, Active-HDL, ISE, entre otros. Se está utilizando el estándar 1164 del IEEE, para garantizar la portabilidad del diseño a cualquier plataforma de programación de PLD's.

4. Metodología

El desarrollo del software se basa en la metodología que se muestra en la Figura 1.

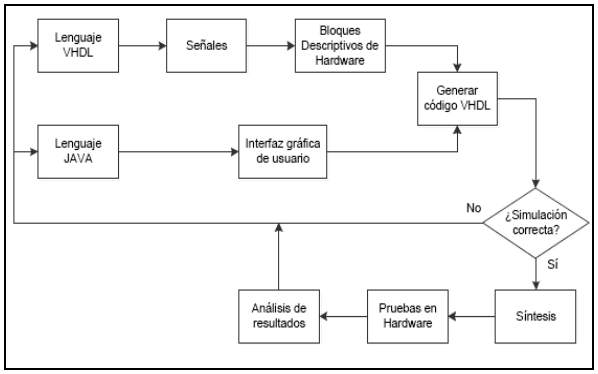

Fig. 1. Metodología de diseño del SHLD.

4.1 Lenguaje VHDL.

Es necesario estudiar su sintaxis, herramientas de diseño, aplicaciones y todo lo necesario para diseñar, sintetizar y programar un PLD. Se está utilizando el estándar del IEEE para garantizar la portabilidad del diseño a cualquier plataforma de programación de este tipo de dispositivos.

4.2 Señales.

Después de que son creados los BDH, deben ser conectados entre sí por medio de señales, permitiendo que exista una relación entre éstos.

4.3 Bloques Descriptivos de Hardware.

Se refiere al símbolo con el que se identifica los BDH, ejemplos de ellos pueden ser las compuertas lógicas, flip flop, bloques aritméticos, bloques combinacionales y bloques relacionales.

4.4 Lenguaje de Programación Java.

El software se desarrolla bajo la plataforma de Java, debido a que éste es un lenguaje orientado a objetos y es de uso libre. Permite interactuar con gráficos que simulan los BDH al igual que la iteración entre bloques y señales, por medio de clases, instancias, objetos, herencia y polimorfismo.

En la Figura 2 se muestra el diagrama de clases para la base del diseño de software, posteriormente se explicará cada una de las clases generalizadas para el desarrollo del mismo.

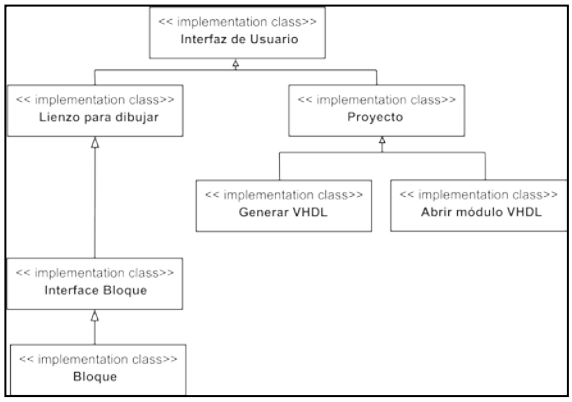

Fig. 2. Diagrama de clase.

4.5 Clase Interfaz de usuario.

Es la clase principal del software, en ésta se definirán las propiedades de la ventana así como la distribución y creación de los métodos para la barra de menús y submenús para la creación de los diagramas.

4.6 Clase Lienzo para dibujar.

Se crearán y se dibujarán objetos gráficos que representen la simbología de todos los BDH, las señales, la caja negra, entre otros. Es posible manipular los bloques en cuanto a su posición como a sus propiedades por medio de los eventos del mouse.

4.7 Clase Interface Bloque.

Esta clase implementará los métodos que pertenecen a la clase Bloque, de tal manera que pueda accesar a todos los BDH de manera genérica.

4.8 Clase Bloque.

Esta clase se establecerá para cada uno de los BDH con un nombre único. Implementará los atributos necesarios para la construcción del objeto BDH, todos éstos serán declarados como *private*.

4.9 Clase Proyecto.

Corresponde a las acciones que se podrán hacer en el proyecto, tales como generar un nuevo archivo, guardar y cerrar.

4.10 Clase Generar Código VHDL.

Esta clase contiene atributos necesarios para la creación del código en VHDL, esta clase se invocará una vez que el usuario haya terminado de realizar el diagrama a bloques.

4.11 Clase Abrir Módulo VHDL.

Permitirá al usuario abrir un archivo de tipo .vhd, con el objetivo de cargar módulos prediseñados para que sean implementados en un nuevo diseño.

4.12 Simulación.

Una forma de comprobar que nuestro código ha sido implementado de forma adecuada, es utilizando cualquier herramienta de simulación, para poder comprobar la funcionalidad de la descripción diseñada.

4.13 Pruebas en hardware.

Una vez que el código ha sido generado, una forma de implementarlo en hardware es a través de una tarjeta FPGA para complementar el proceso y utilidad del código generado. Debido a que el código generado se basa en el estándar del IEEE, la descripción tiene una total portabilidad a las herramientas de cualquier fabricante.

5. Pruebas y Resultados

Se realizaron las pruebas necesarias para comprobar la implementación del código VHDL generado por la herramienta SHDL. Para poder llevar a cabo la simulación fue necesario hacer uso de la herramienta Active-HDL, obtenido gracias al programa universitario de Aldec, en el cual se carga el código en VHDL generado por el SHDL y luego se compila con el objetivo de analizar y corregir posibles errores en la arquitectura diseñada.

A continuación se presentan soluciones a problemas reales mediante el uso del Software, desarrollando un Diagrama a Bloques, Código en VHDL y la Simulación, respectivamente.

En la Figura 2 se muestra los componentes y señales que en conjunto conforman el diagrama a bloques de un Sumador. Donde se han generado dos señales de entrada A y B y una señal de Salida Y.

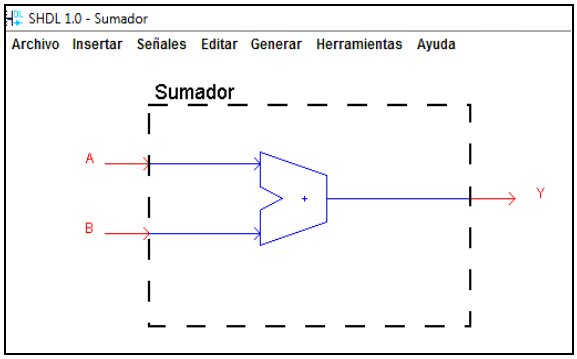

Fig. 2. Diagrama a Bloques de un sumador.

La Figura 3 muestra el código generado después de la implementación del diagrama a Bloques del Sumador presentado en la Figura 2.

Codigo

Sumador.vhd

```
1   library IEEE;
2   use IEEE.std_logic_1164.all; -- Libreri estandar
3   use IEEE.std_logic_arith.all;
4   use IEEE.std_logic_unsigned.all;
5   --Descripcion de caja negra
6   entity Sumador is
7   port(
8       Y: out std_logic_vector(3 downto 0); -- Salidas de un bit
9       B: in std_logic_vector(3 downto 0); -- Entradas de un bit
10      A: in std_logic_vector(3 downto 0) -- Entradas de un bit
11  );
12  end Sumador;
13  -- Descripcion del circuito
14  architecture Sumador of Sumador is
15  begin
16      Operacion: process(A, B)
17      variable RU: unsigned(3 downto 0);
18      begin
19          RU := unsigned(A) + unsigned(B);
20          Y <= std_logic_vector(RU);
21      end process;
22  end Sumador;
23
```

Fig. 3. Código VHDL del Sumador.

Finalmente, en la Figura 4 se muestra la simulación del código generado en la Figura 2, utilizando la herramienta Active-HDL, y se pudo comprobar que el código ha tenido los resultados esperados. Es importante mencionar que el código generado para un sumador puede ser realizado de muy diversas maneras, y en este caso, decidimos utilizar este código pues es el que más hemos utilizado y que nos ha dado muy buenos resultados, especialmente en el caso de que el sumador cuente con sumadores en hardware, como es el caso de los dispositivos de Xilinx, permitiendo un ahorro considerable de hardware. La flexibilidad del software es tal, que el usuario puede crear su propio bloque sumador y utilizarlo para el diseño de sus descripciones de hardware.

Fig. 4. Simulación del Sumador.

Como principal limitación, el Software SHDL 1.0 no cuenta con la herramienta necesaria para el proceso de Simulación, por lo que es necesario el uso de determinadas herramientas en el mercado que permitan Simular.

6. Conclusiones

Al término del desarrollo de este proyecto se logró como resultado el software SHDL 1.0 el cual es una herramienta que permitirá el desarrollo e implementación de sistemas digitales de manera rápida y confiable pues al estar basado en los estándares de la IEEE es una herramienta compatible en cualquier plataforma y no se limita a una sola tecnología.

Al ser un software desarrollado en un lenguaje orientado a objetos tiene la facilidad de crecer de una manera sencilla y poderosa dando oportunidad de integrar herramientas que lo hacen un importante competidor ante las herramientas de software que en la actualidad son usados en el desarrollo de aplicaciones basado en descripción de lenguajes de hardware. SHDL 1.0 es una herramienta que provee a los estudiantes el soporte necesario para fomentar el aprendizaje y desarrollar proyectos que dan solución a problemas reales, también siendo un apoyo en el ámbito profesional en el área de Ingeniería.

El lenguaje VHDL dado que es un lenguaje versátil que puede ser usado para modelar y simular, es muy práctico a la hora de desarrollar aplicaciones tales como un Sistema de Detección PD usando FPGA [5], Diseño en FPGA para procesamiento de imagen utilizando la interfaz gráfica de usuario de un Web-Based Generador de código VHDL [6], Enfoque de diseño para FPGA implementación de Controlador Flexray utilizando VHDL para Intra comunicación de la aplicación Vehicular[7], Diseño de un controlador de luz FPGA basado en la inteligencia de tráfico con VHDL [8].

El alcance del Software radica en que hasta el momento no cuenta con la herramienta que permita los procesos de Simulación, Compilación, e interconexión de señales acorde a un dispositivo en particular; éstas herramientas están planeadas en trabajo a futuro en el desarrollo del Software.

Referencias

[1] Bozich, E. *"Introducción a los Dispositivos FPGA. Análisis y ejemplos de diseño", 2005.*

[2] Bailey, D. *"Design For Embedded Image Processing On FPGAs"*, John Wiley & Sons, (Asia) Pte Ltd, Massey University, New Zealand, 2011.

[3] Aceves, M., Ramos J. *"Fundamentos de Sistemas Embebidos Mediante Lenguajes Descriptivos de Hardware"*, Asociación Mexicana de Mecatrónica, México, 2010.

[4] Reyes, F., Cid J., Vargas E. *"Mecatrónica. Control y Automatización"*, Alfaomega, México, *2005.*

[5] E., Kumar C., Basri A., Agileswari K. *"VHDL Simulation of Peak Detector, 64 Bit BCD Counter and Reset Automatic Block for PD Detection System Using FPGA"*, 6th International Colloquium on Signal Processing & Its Applications (CSPA), Department of Electronics and Communication Engineering, College of Engineering (COE), pp. 149 – 155, *2010.*

[6] Schumann T., Dewi A., *"FPGA Design for Image Processing Using a GUI of a Web-Based VHDL Code Generator"*, Faculty of Electrical Engineering and Information Technology, University of Applied Sciences, Conference Visual Communications and Image Processing (VCIP), 2011.

[7] Gaidhanel A., Khanapurkar M., Bajaj P., *"Design Approach for FPGA Implementation of Flexray Controller Using VHDL for Intra Vehicular Communication Application"*, First International Conference on Emerging Trends in Engineering and Technology, pp. 100 2-1006, 2008.

[8] Sourav N. Chandrajit P., Suman S., Sreyanka M., Abhishek R., Arghya G., Debdatta K. *"Design of an FPGA Based Intelligence Traffic Light Controller with VHDL"*, International Conference on Radar, Communication and Computing (ICRCC), SKP Engineering College, pp. 92- 97, 2012.

[w1] www.altera.com, fecha de consulta: 12 Agosto, 2010.

[w2] www.aldec.com, fecha de consulta: 12 Agosto, 2010.

[w3] www.xilinx.com, fecha de consulta: 12 Agosto, 2010.