Revista Facultad de Ingeniería N.° 34. pp. 72-85. Septiembre, 2005

Analizador lógico de tiempos implementado en arquitectura digital reprogramable

Eugenio Duque, José Édinson Aedo*, Julián Correa**, Alexis Alberto Ramírez**, Camilo Torres**, Rubén Darío Nieto**, Álvaro Bernal***

(Recibido el 2 de julio de 2004. Aceptado el 7 de abril de 2005)

Resumen

En este artículo se describe la concepción, diseño, simulación e implementación de un analizador lógico de tiempos implementado sobre una arquitectura digital reprogramable. El sistema fue especificado en VHDL [1] e implementado en una plataforma basada en una FPGA (Field Programmable Gate Array) Spartan II. El uso de esta metodología para la implementación del analizador, permite obtener un sistema flexible, económico y eficiente en cuanto a capacidad de procesamiento, ya que su característica modular hace posible escalar el sistema cuando sea necesario utilizando varios de los subsistemas desarrollados.

---------- *Palabras clave:* analizador lógico de tiempos, generador de estímulos, analizador lógico programable vía Internet.

Timing logic analyzer implemented in reprogrammable digital architecture

Abstract

The conception, design, simulation, and implementation of a timing logic analyzer implemented on a reprogrammable digital architecture are described in this paper. The system was specified in VHDL [1] and implemented in a platform based on a FPGA (Field Programmable Gate Array) Spartan II. This methodology for analyzer implementation, allows obtaining a flexible, economic an efficient system in regards to processing capacity, since its modular characteristics make possible, through the use several of the developed subsystems, to scale the system when necessary.

---------- *Key words:* timing logic analyzer, stimulus generator, programmable logic analyzer using Internet.

* Grupo de Microelectrónica y Control. Universidad de Antioquia. Calle 67 N.º 53-108, oficina 18-310. joseaedo@udea.edu.co.

** Grupo de Arquitecturas Digitales y Microelectrónica. Universidad del Valle. Calle 13 carrera 100, A. A. 25360. Teléfono: (572) 339 17 80. Fax: (572) 339 23 61. Cali, Colombia. alvaro@univalle.edu.co.

Introducción

En el diseño y desarrollo de sistemas electrónicos se utiliza una metodología de diseño, en la cual una de sus etapas principales consiste en las pruebas funcionales del sistema. Para este propósito, se utilizan instrumentos adecuados tales como los analizadores lógicos, los cuales permiten censar simultáneamente diferentes canales de información digital almacenando las muestras tomadas para ser posteriormente desplegadas y analizadas. Una dificultad para realizar este proceso es el elevado costo que tienen los analizadores lógicos. Por esta razón en este trabajo se presenta el diseño e implementación de un analizador lógico de bajo costo implementado usando FPGA [2].

Los principios funcionales de un analizador lógico se pueden dividir en cinco subsistemas a saber: el bloque de muestreo y captura, el bloque de disparo, el bloque de almacenamiento, el bloque de despliegue y el bloque de control. Cada uno de estos subsistemas fue modelado y simulado utilizando VHDL.

Arquitectura general

La arquitectura adoptada para este analizador se muestra en la figura 1, en ella se detallan los subsistemas principales y sus interconexiones.

Debido a la arquitectura del analizador y a los requerimientos en las características del mismo, los bloques funcionales: de muestreo y captura, el bloque de disparo y control, se implementaron de la manera indicada en figura 1, sobre un dispositivo reprogramable (Xilinx Spartan II [2]).

Se utilizaron memorias FIFO SN74v293 (Texas Instruments Inc.) [3], para el almacenamiento de los datos de muestreo y los *glitches*. Se diseñó una interfaz de comunicación por el puerto paralelo, la cual también fue implementada en la FPGA. Esta interfaz permite el intercambio de datos con un PC.

Bloque de muestreo y captura

Este bloque se encarga de muestrear las señales correspondientes a los canales de entrada del

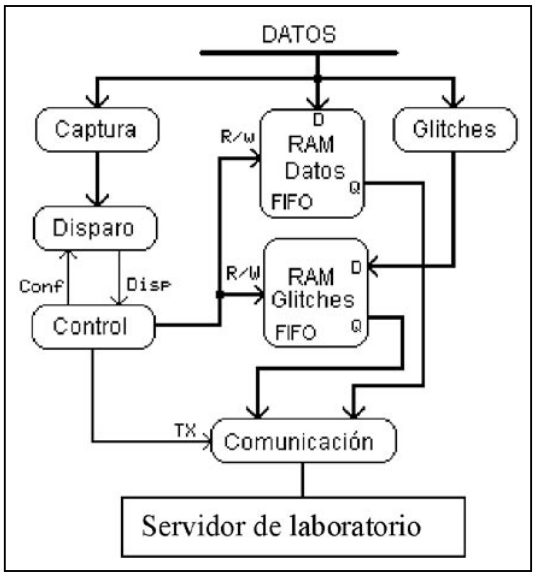

Figura 1 Esquema general del analizador lógico

analizador y de su captura una vez ésta sea requerida. Tiene como función escribir los datos en las memorias FIFO dependiendo de las condiciones en que se encuentre tanto el circuito de disparo como las memorias. Estas condiciones son los modos de captura que pueden ser configurados en el analizador y el estado de almacenamiento en las FIFO. Para las primeras condiciones (condiciones de disparo como *pre-trigger*, *pos-trigger*, etc.) este bloque tiene como objetivo primordial asegurarse de que el estado de almacenamiento de la FIFO (memoria medio llena, medio vacía, memoria llena, memoria vacía, memoria a la mitad) coincida con los requerimientos de la configuración seleccionada. En la figura 2 se muestra el diagrama de este bloque.

Módulo de detección de glitches

Este módulo requiere de un tratamiento especial, diferente al utilizado en los otros bloques. Primero se necesita definir apropiadamente el concepto de *glitch* o señal espuria. Una señal espuria es un cambio espontáneo, indeseado e inesperado de una señal que hace parte de un sistema; este tipo de señales tiene duración temporal y podría asumirse como una señal con determinada fre-

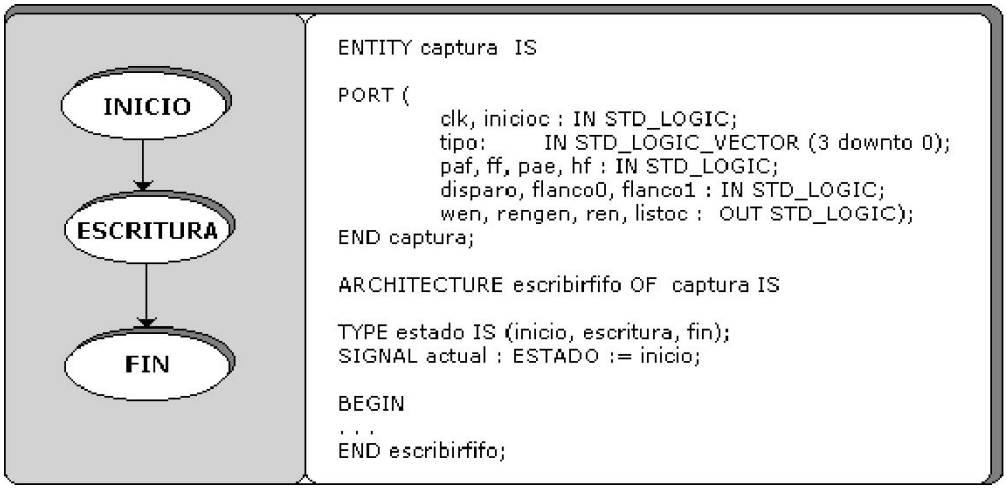

Figura 2 Diagrama de bloque de captura

cuencia, evidentemente más alta que la frecuencia de funcionamiento del sistema. En consecuencia, para detectar una señal espuria se requiere un circuito que pueda muestrear a una frecuencia más alta que la frecuencia de funcionamiento del sistema, por lo cual si el analizador cumple con esta condición, entonces actuaría como detector de señales espurias.

En la figura 3 se muestra el diagrama esquemático del circuito de este bloque.

Bloque de disparo

Este bloque se encarga de comparar los datos capturados en los canales del analizador con la palabra de control determinada por el usuario en la configuración del analizador. La detección de la palabra de disparo, en este bloque, se efectúa por nivel. La comparación depende del número de *bits* que se utilicen en la palabra de control.

En la figura 4 se muestra el diagrama de estados de este bloque.

Bloque de almacenamiento y configuración de las FIFO

El elemento principal de este bloque corresponde a las memorias FIFO, las cuales tienen un ancho de palabra de 18 bits y una profundidad de 32 K palabras; el manejo de estos dispositivos se efectúa por medio de un componente de control de FIFO dentro del bloque de control.

El analizador lógico cuenta con 2 memorias FIFO, las cuales almacenarán tanto los datos muestreados, como los *glitches* que se presenten en cada canal. Estas memorias tienen la particularidad de informar al usuario cuando están vacías, llenas o cuando se ha alcanzado un porcentaje de almacenamiento, el cual puede ser configurado por el usuario.

Debido a la correspondencia directa que tiene este dispositivo con las funciones de inicialización, se hace necesario que un bloque del analizador configure las memorias FIFO. En la figura 5 se presenta el diagrama de bloques de la máquina de inicialización de las memorias FIFO.

Bloque de control

Este bloque se encarga de efectuar tres tareas generales:

- Comunicación. En esta tarea se efectúa la inicialización del analizador lógico recibiendo los vectores de configuración que serán provistos por la interfaz paralela y además suministra las banderas de estado del analizador (configuración, captura, espera, listo).

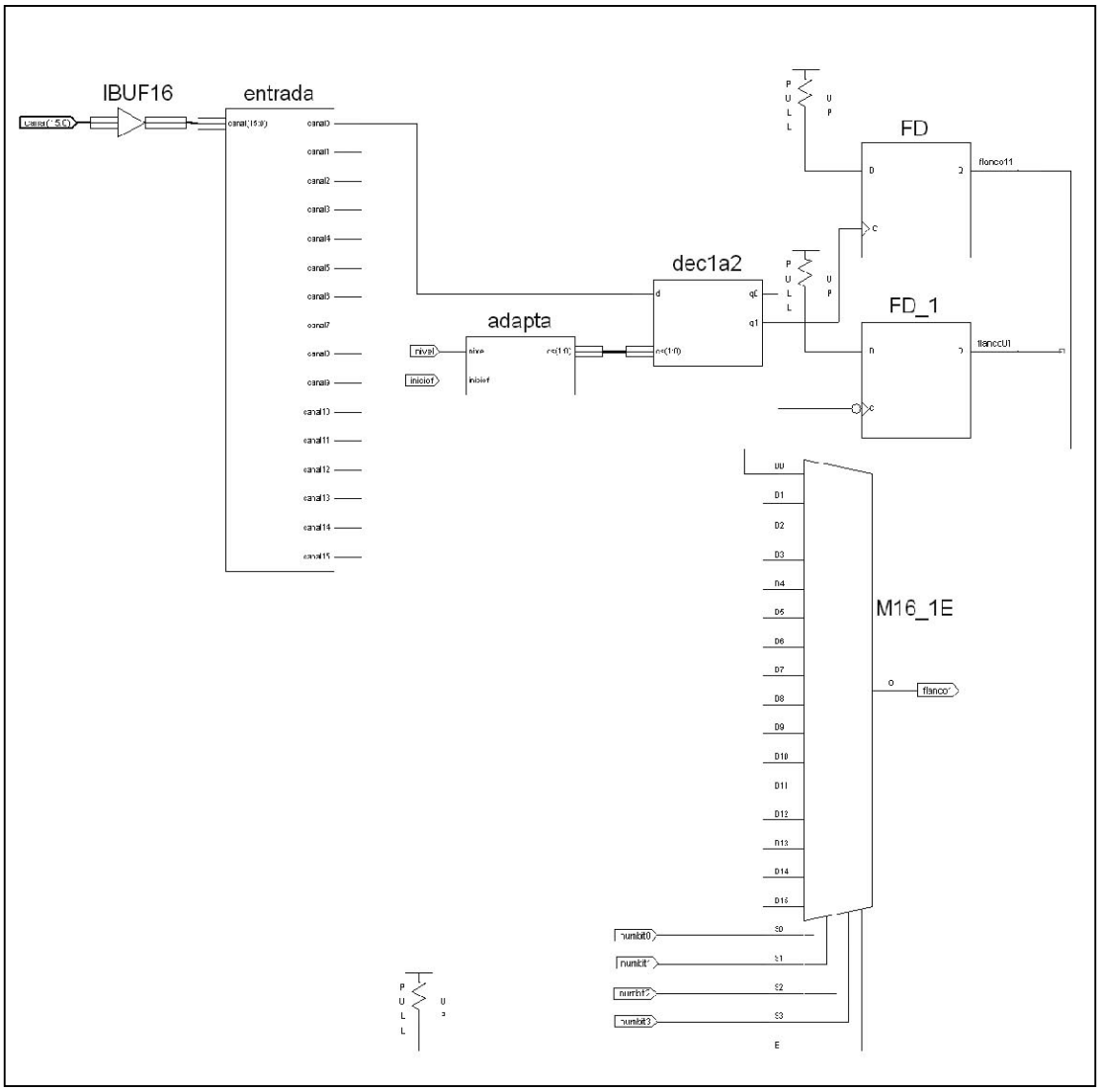

Figura 3 Diagrama esquemático del circuito de *glitches*

- Configuración de disparo. En esta tarea se establece el modo de captura que tendrá el analizador de acuerdo con la configuración recibida y el disparo. Se recibe una palabra de disparo (1 a 16 bits) que será entregada al bloque de disparo, además de unas señales de configuración para este último, a su vez se fija el modo de captura de *trigger*, ya sea *pos- trigger* o *pre-trigger*, para determinar la manera como la FIFO almacenará los datos [4, 5].
- Control de FIFO. Inicialmente esta tarea se encarga de administrar el modo como se escribe en la FIFO. Este modo depende del tipo de disparo, es decir, para un disparo *pos-trigger* el bloque de control espera que la señal de disparo (proveniente del bloque de disparo) se active para iniciar la escritura en la RAM hasta que ésta se llene; para un disparo *pre-trigger* se inicia el proceso de escritura en la RAM hasta que se detecte la activación de la

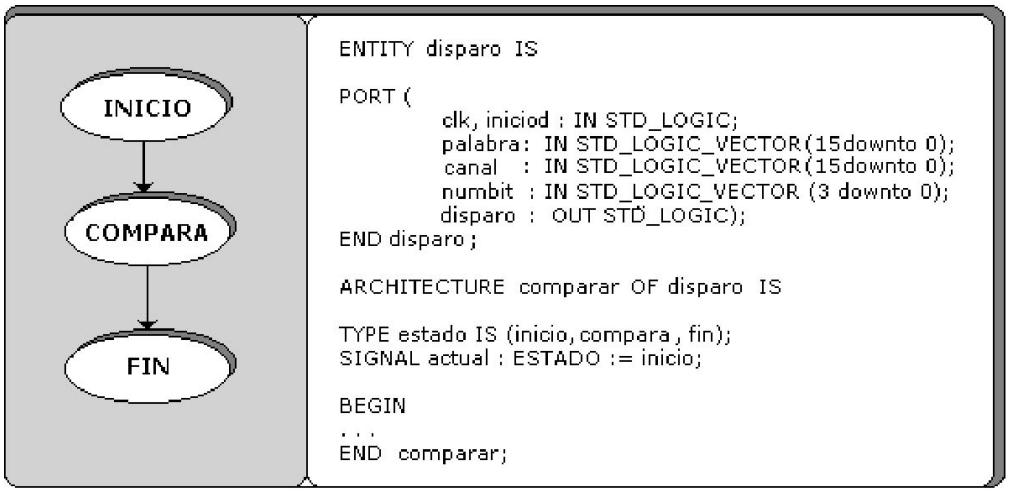

Figura 4 Diagrama de bloque de disparo

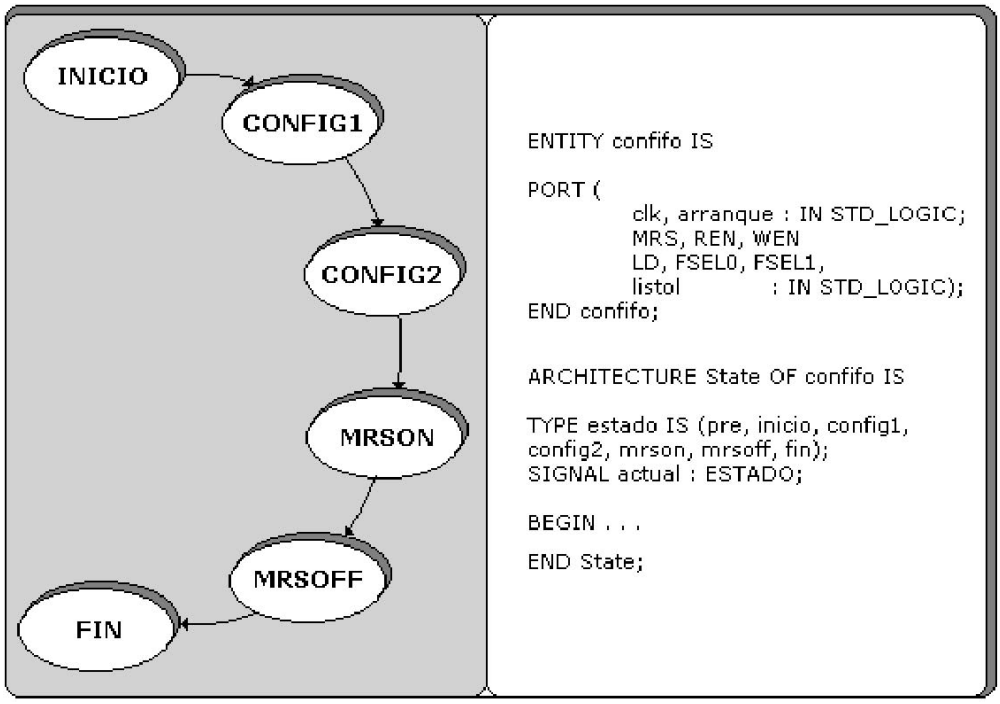

Figura 5 Diagrama de la configuración de las FIFO

señal de disparo, en este caso, para evitar que la RAM se llene sin haberse activado la señal de disparo, se da inicio a un proceso de escritura/lectura simultánea sobre la FIFO por una cantidad de palabras antes del llenado total de la RAM, hasta detectar la activación de la señal de disparo. Adicional a esto se tiene un componente de inicialización de FIFO que fue

descrito en el bloque de almacenamiento. En la figura 6 se muestra el diagrama del bloque de control.

Bloque de comunicación

La comunicación con los módulos se realizará a través de una interfaz paralela, la cual tiene como característica principal que se implementa sobre una FPGA totalmente reconfigurable a través de un bloque de comunicación. Este bloque se usa para codificar y multiplexar los datos enviados a través del puerto paralelo. Este bloque toma una señal de código llamada *"decodi"* enviado a través del puerto 37AH (descrita en la tabla 1) y dependiendo de su valor puede tomar acciones sobre sus salidas o definir a donde deben ir los datos recibidos por el puerto 378H.

Las principales funciones de la interfaz paralela son:

- Establecer comunicación entre los módulos analizador lógico, generador de estímulos y matriz de interconexión.
- Comunicar los códigos que representan las palabras de disparo, activación del disparo en *post-trigger* y *pre-trigger*, la detección de *glitches*, velocidad de muestreo e inicio de operaciones (*start*).
- Vigilar las señales de estado del analizador lógico y el generador de estímulos. Estas señales son: "dispositivo activo", "dispositivo en *standby"* y "dispositivo apagado". El analizador lógico en estado activo indica que se encuentra en la etapa de adquisición de datos o esperando las señales de activación de disparo. En estado *standby* el analizador lógico se encuentra con la FPGA programada para que el módulo actúe como analizador lógico y se pueda editar la palabra de disparo y los modos

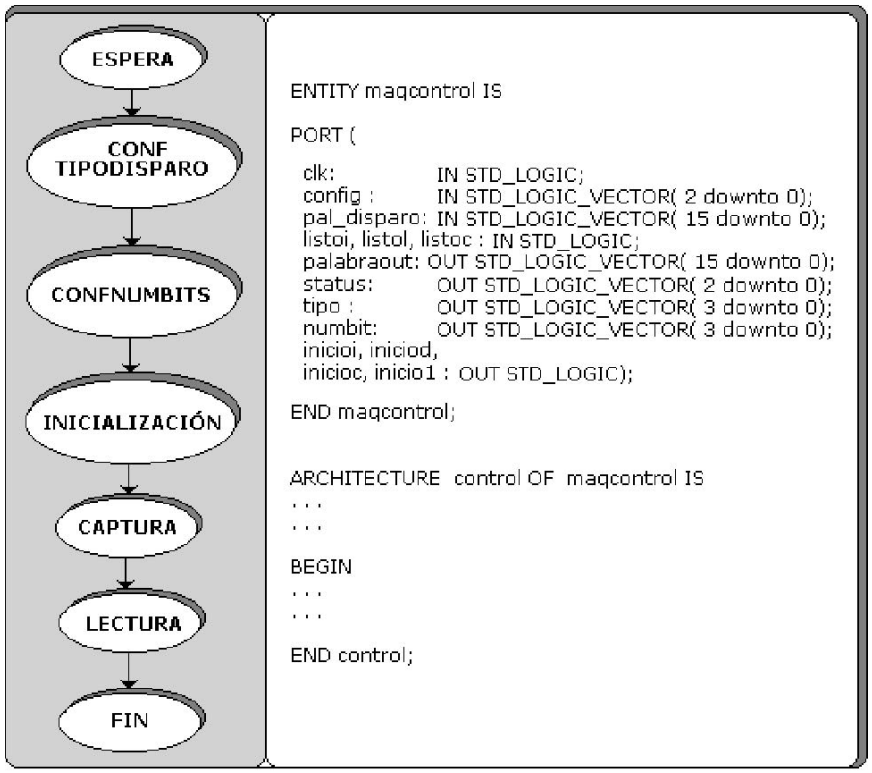

Figura 6 Diagrama del bloque de control

Tabla 1 Valores y función de la señal Decodi

Decodi 37AH	*Función*
0000	Computador ocupado procesando datos
0001	Valor en 378H es Config.1 tipo, 2 numbit, 3 palabra, etc.
0010	Valor en 378H al Byte bajo del dato
0011	Valor en 378H al Byte alto del dato
0100	Concatena el Byte alto y el bajo para la salida "*dato*"
0101	Concatena el Byte alto y el bajo para la salida "*gene*"
0110	Pasa solicitud de dato del computador al analizador
0111	Pone en 379H el "*status*" del analizador
1000	Pone en 379H el nibble 3 del dato almacenado en FIFO
1001	Pone en 379H el nibble 2 del dato almacenado en FIFO
1010	Pone en 379H el nibble 1 del dato almacenado en FIFO
1011	Pone en 379H el nibble 0 del dato almacenado en FIFO
1100	NO UTILIZADO (reservado para programación)
1101	NO UTILIZADO (reservado para programación)
1110	NO UTILIZADO (reservado para programación)
1111	NO UTILIZADO (reservado para programación)

de disparo; y el estado apagado para indicar que el módulo no se encuentra programado. El generador de estímulos tiene su equivalente de estados de tal manera que el estado activo indica que la FPGA del módulo se encuentra programada para que el módulo se comporte como generador de estímulos y se encuentra en operación. El estado *standby* indica que el generador está listo para recibir las señales que serán transmitidas hacia la FPGA de prueba y el estado apagado indica que no se encuentra programado.

- Transmitir las señales que serán utilizadas por el generador de estímulos para ser almacenados en la memoria FIFO del módulo y activar sus disparos.
- Transferir los datos almacenados en el analizador lógico hacia el PC para ser visualizados en pantalla y transmitidos a través de Internet.

Software del sistema

La interfaz gráfica de usuario debe cumplir con todas las operaciones de configuración, control y monitoreo del analizador lógico. Dentro de las funciones que debe proveer dicha interfaz se tiene:

- Configuración de los dispositivos reprogramables (FPGA).
- Configuración de los parámetros de operación del instrumento.
- Configuración de los vectores de estímulo del sistema.
- Monitoreo del estado de operación del analizador lógico.
- Descarga de los datos muestreados por el analizador para su análisis y despliegue en la interfaz gráfica.

En la figura 7 se presenta un diagrama de flujo del *software* de sistema.

Pruebas y resultados

Cada uno de los bloques fue modelado en VHDL y validado por simulación lógica. También se implementaron en un sistema de desarrollo basado en una FPGA y el *hardware* necesario

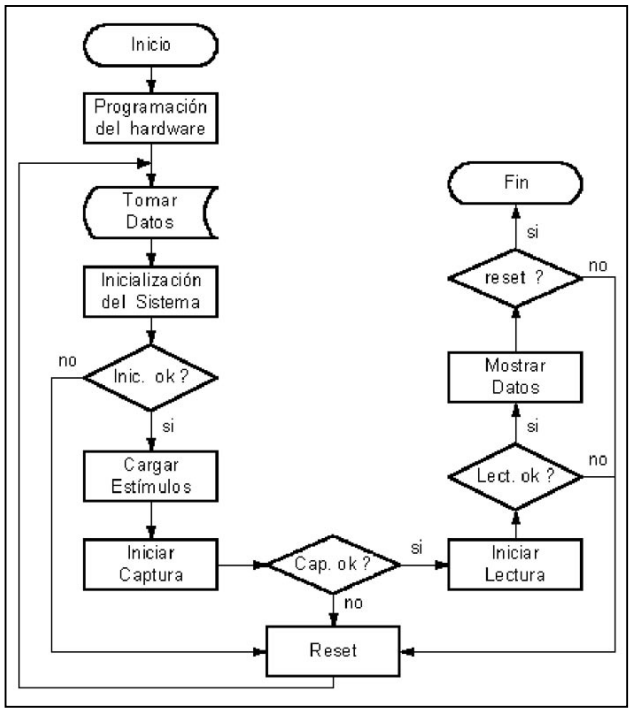

Figura 7 Diagrama de flujo general del sistema

para ello. A continuación se detalla el proceso de simulación e implementación.

Simulación e implementación de la inicialización de las FIFO

Las memorias FIFO usadas en este proyecto deben ser configuradas antes de su utilización, ya que éstas cuentan con múltiples formas de inicialización. Para la memoria FIFO se usó la siguiente secuencia de configuración *(setup):*

- Ancho del bus de entrada: 18 bits.
- Ancho del bus de salida: 9 bits.
- Modo de operación: estándar.
- Organización de los datos: *Big Endian.*
- Modo de retransmisión: normal.
- Método de programación: paralelo.
- *Offset*: 127 palabras antes del estado *full.*
- Paridad: no entremezclada.

Las características mencionadas corresponden a estados lógicos en los pines de las FIFO, los cuales son proporcionados por el bloque de configuración y almacenados por las FIFO a través de la función *Master Reset* que ellas poseen; la simulación que se muestra en la figura 8 corresponde a este ciclo de configuración; en la figura 9 se muestra el resultado de este ciclo visto por el analizador HP1661A que permite verificar que la implementación esté de acuerdo con la simulación del sistema.

Sistema de desarrollo

En esta sección se presenta la implementación en hardware en donde se programó y comprobó la funcionalidad del sistema.

Este sistema de desarrollo (figura 10) se diseñó teniendo en cuenta la máxima flexibilidad que se podía alcanzar, contando con las características

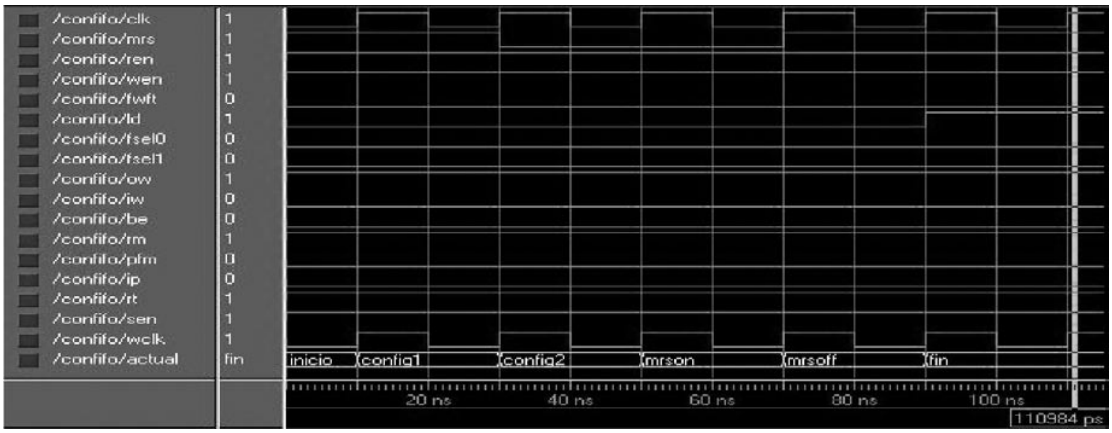

Figura 8 Simulación inicialización de las FIFO

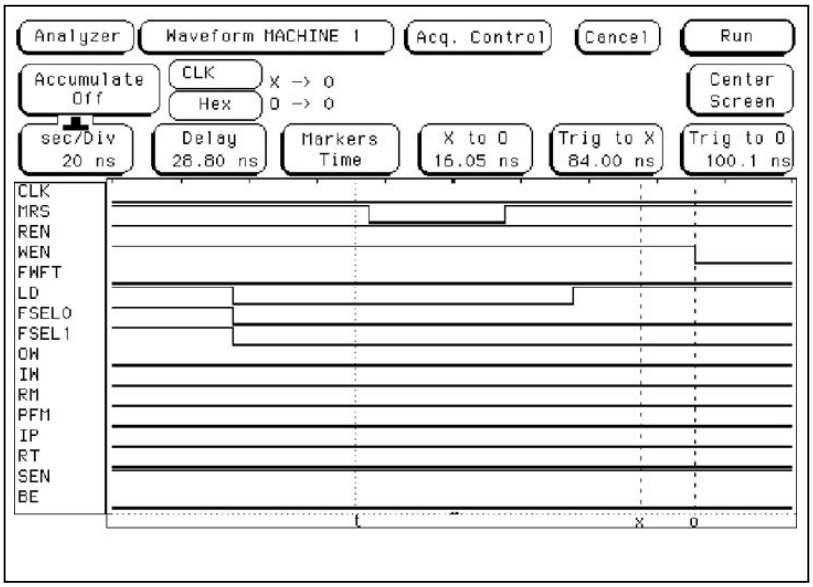

Figura 9 Implementación Inicialización de las FIFO

de cada uno de los dispositivos que lo componen, principalmente la FPGA y las memorias FIFO. Se da la alternativa al usuario del sistema de desarrollo, de poder controlar todos los dispositivos con las funciones programadas en la FPGA a través del puerto paralelo de un computador.

El componente *hardware* del sistema consta de:

- Tarjeta ALGEN (Analizador Lógico GENerador de estímulos).
- Tarjeta TMP (Tarjeta de Multiplexación del *puerto Paralelo*).
- Tarjeta de usuario (sistema *D2E*).

La tarjeta ALGEN cuenta con:

- FPGA de Xilinx, XC2S200 (200,000 compuertas).
- Memorias FIFO SN74V293.

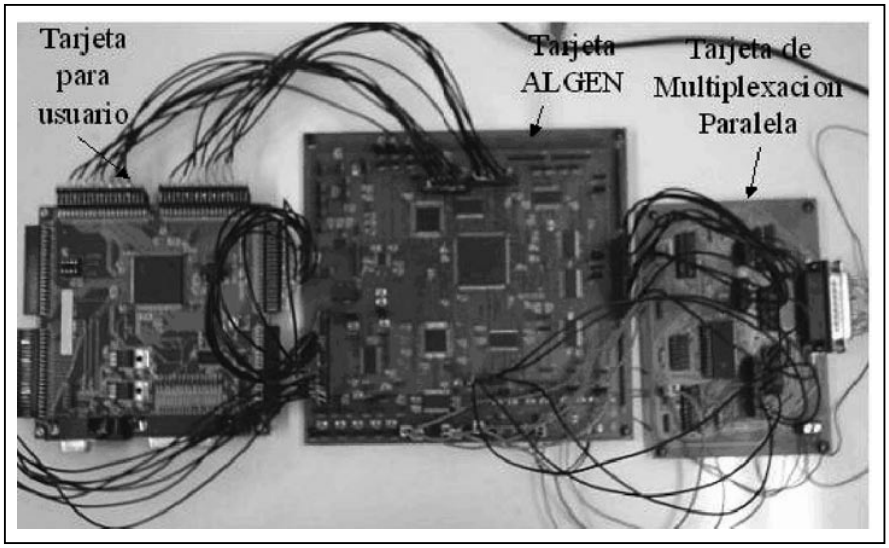

Figura 10 Sistema de desarrollo

- Oscilador de 50 Mhz.
- Clock buffer para manejar 6 señales de reloj sin retardos.
- Buffers de protección para cada canal de la tarjeta.
- Construcción del PCB a dos capas.
- Reguladores de 3,3 V y 2,5V a 1,5 A máx.
- Alimentación externa de 4,58 V.

Según las pruebas realizadas a esta tarjeta el consumo de potencia total es de 1,28 W.

La tarjeta TMP fue concebida para trabajar con un solo puerto paralelo, que debe cumplir con las siguientes funciones:

- Programación del analizador lógico/generador de estímulos.
- Programación del sistema del usuario (sistema bajo prueba).
- Configuración y control del analizador lógico en operación.
- Transmisión de los vectores de estímulos desde el computador hasta la tarjeta *ALGEN*.
- Verificación del estado y transmisión de datos capturados hacia el computador.

Prueba de respuesta del sistema para un circuito secuencial

Para esta prueba se configura en el sistema de usuario (circuito bajo prueba) un contador de 16 bits que será activado por medio de una señal de arranque que introduce el analizador a través de sus vectores de estímulo.

Este contador se implementó sobre el sistema de desarrollo D2E [6] y se operó con una frecuencia de 12,5 MHz. Luego se efectuó la captura y posteriormente la transmisión de los 64 K datos muestreados al computador para su despliegue y análisis.

El componente *software* del sistema analizador lógico/generador de estímulos está en capacidad de crear un archivo de tipo texto con todas las muestras transmitidas por el analizador lógico después de su captura, éstas son desplegadas por medio del módulo de visualización del *software.* Se presenta una gráfica con los datos capturados por el analizador lógico HP1661A y a conti-

nuación los datos capturados por el analizador ALGEN.

En la figura 11 se muestran los datos capturados por el analizador HP1661A para los primeros 12 datos.

Estos datos fueron capturados simultáneamente por el ALGEN; los resultados de esta captura se muestran en la figura 12.

Como se puede apreciar en las figuras 11 y 12 la captura de los datos generados por el sistema de desarrollo D2E son similares.

En este caso no se puede hablar de una captura idéntica de los dos instrumentos ya que la frecuencia de muestreo de ellos es diferente y no tienen la misma referencia de tiempo. La frecuencia de muestreo del analizador lógico HP1661A es de 16 ns y la del ALGEN es de 20 ns.

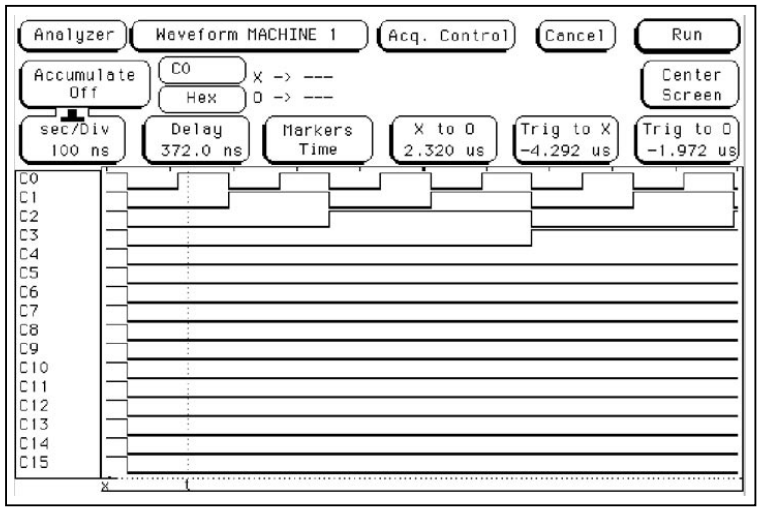

Figura 11 Captura del HP1661A para el contador

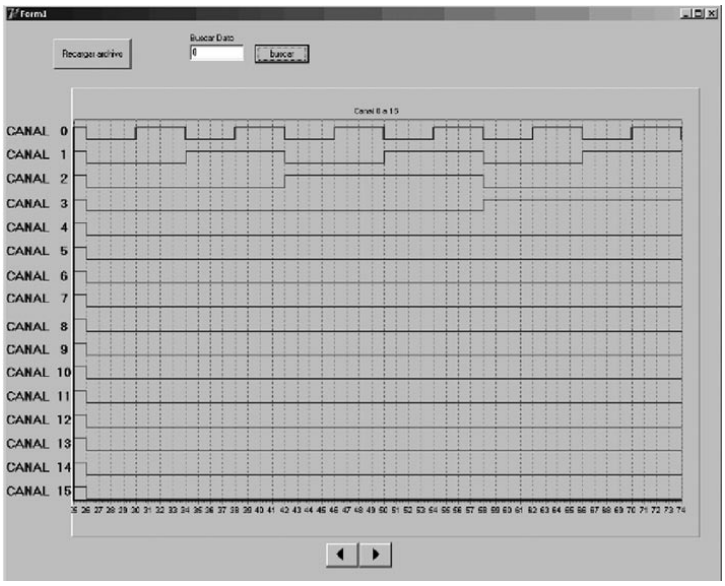

Figura 12 Captura del ALGEN para el contador

Los resultados en estas figuras verifican el correcto funcionamiento del analizador lógico ALGEN al compararlo con la captura realizada por un instrumento embebido comercial (Hewlett-Packard HP1661A). Además se comprueba la calidad de las sondas del sistema.

En esta prueba no es posible comparar todos los datos muestreados por el analizador ALGEN ya que la profundidad de memoria de éste es 16 veces la profundidad del analizador HP1661A, sin embargo para los primeros 4 K datos muestreados los resultados son iguales.

Análisis de desempeño

Bajo arquitecturas similares se pueden observar trabajos diferentes al presentado en este artículo, los cuales presentan algunas características similares y otras diferentes a las obtenidas con el sistema enunciado en este artículo, las cuales son detalladas en la tabla 2. Algunos de estos trabajos son:

- Analizador lógico de tiempos implementado en un FPGA utilizando el bus PCI como interfaz de comunicación. Centro de investigación en computación. Instituto Politécnico Nacional. México D. F. [7].
- Analizador lógico de 100 Mhz utilizando FPGA. Instituto de Ingeniería Eléctrica. Universidad de la República. Montevideo, Uruguay [8].
- Handheld pocket logic analyzer [9].

Como se puede observar ninguno de estos analizadores incluyen un sistema generador de estímulos, el cual está presente en el sistema expuesto en este artículo.

Según los resultados obtenidos en las pruebas realizadas sobre el diseño final, se pueden deducir las siguientes características técnicas del sistema ALGEN:

- Frecuencia de muestreo máxima para una captura realimentada, es decir llevando los datos del generador directamente a los canales de entrada del analizador lógico es de 50 MHz, esto es debido a la sincronización entre los datos que se generan y su captura. Esto evidencia la relevancia del fenómeno de *Skew clock* entre los diferentes dispositivos de la tarjeta gracias al *Clock Buffer* que posee la misma.
- Frecuencia máxima del circuito muestreado para una captura no realimentada, es decir capturando las salidas desde el sistema de usuario: 20 MHz (cumpliendo con el criterio de muestreo).
- La frecuencia de funcionamiento del sistema es de 50 MHz (frecuencia del reloj del sistema).
- La profundidad de muestras del sistema es de 64 K muestras.

Tabla 2 Características de otros analizadores

Modelo	***Analizador lógico México D. F.***	***Analizador lógico Montevideo***	***Handheld Pocket Logic Analyzer***
Canales	16	48	8
Generador de patrones	No	No	No
Vel. tiempo	50 Mhz	100 Mhz	20 Mhz
Memoria	256 KB	16 KB	256 KB
Tipo de dispositivo	FPGA	FPGA	CPLD
Modelo del dispositivo	Xilinx XL XCS20XL	FLEX 6000 EPF6016TC144-2	XPLA3 CoolRunner XCR3256XL-10CES
Simulado	Sí	Sí	Sí
Implementado	No	No	Sí (NowTech test board)

- La profundidad de estímulos del sistema es de 64 K estímulos.
- El número de canales para el analizador es de 16 canales.
- El número de canales para el generador es de 16 canales.
- Se cuenta con una conexión al computador a través del puerto paralelo.
- La potencia máxima consumida por el sistema ALGEN y la tarjeta TMP juntas es de 1,31 Watts.
- La corriente máxima a través del sistema ALGEN y la tarjeta TMP es de 290 mA.
- Se utiliza una alimentación externa de 4,58 Volts D. C.

Conclusiones

En este trabajo se presenta el desarrollo de analizador/generador orientado a realización de pruebas de sistemas electrónicos digitales. Uno de los objetivos que se buscó al concebir el analizador/generador fue desarrollar una arquitectura modular y una estrategia de control por medio de señales de habilitación. Esto facilita la sincronización por medio de una máquina de estados que acciona los bloques dependiendo de la función que se esté desempeñando dentro del analizador lógico en un momento determinado.

El comportamiento de cada uno de los bloques que constituyen el analizador se verificó mediante simulación lógica, determinando así, rutas críticas y áreas requeridas dentro de la FPGA. Igualmente, se realizó una verificación similar de los bloques del generador.

Gracias a la modularidad de la arquitectura, es relativamente sencillo hacer modificaciones, permitiendo la configuración del analizador y del generador, por ejemplo si se requiere aumentar el número de canales.

Con el aumento de la complejidad de los códigos VHDL para la programación de la FPGA, se observa una relación costo beneficio entre la funcionalidad y la velocidad del sistema. En la medida que la funcionalidad del sistema aumenta el costo por pagar se refleja en el aumento de los tiempos de retardo de las señales que pasan a través de la circuitería interna de la FPGA.

Se maximiza la operación de la FPGA como circuito principal al controlar todas las operaciones y verificar los estados de todos los componentes; dejando el almacenamiento de datos a otros dispositivos externos como lo son las FIFO para datos y *glitches*. Así se evita el gastar recursos de la FPGA al no incluir módulos de almacenamiento de datos en su circuitería interna.

El uso de sistemas reconfigurables en la implementación del sistema de pruebas, provee una gran flexibilidad, ya que permite realizar múltiples programaciones y verificaciones. Igualmente, debido a su arquitectura genérica, da la posibilidad de realizar diferentes tipos de pruebas, y además verificar el comportamiento de los módulos funcionales con medidas reales.

Agradecimientos

Los autores agradecen a Colciencias, a la Universidad del Valle y a la Universidad de Antioquia por el apoyo económico para llevar a cabo este proyecto.

Referencias

1. J. Bhasker. *A VHDL primer*. 2.ª ed. Prentice Hall. 1995. p. 255.
2. Altera Corp. Altera Digital Library 2001 Version 1. CD-ROM 650 MB.
3. F. Clyde et al. *Electronic Instrumentation Handbook*. McGraw-Hill. 3.th ed. New York. 1995. pp. 715-750.
4. Agilent Technologies. *Feeling Comfortable with Logic Analyzers*. Application note 1337. http://ece-www.colorado.edu/~mcclurel/agan1337.pdf. Consultado el 12 de diciembre de 2003.
5. Tektronix, Inc. The XYZs of Logic Analyzers. http://web.mit.edu/6.111/www/s2005/HANDOUTS/LA.pdf Consultado el 25 de enero de 2004.

6. Digilent, Inc. Digilab 2E. http://www.digilentinc.com/Data/Products/D2E/D2E-brochure.pdf. Consultado el 28 de enero de 2004.

7. A. Gelbukh, S. Grigori. *Analizador morfológico disponible*. Centro de Investigación en Computación. México D. F. http://www.auto03.cic.ipn.mx/Indice.html. 2003. Consultado el 2 de diciembre 2003.

8. O. De Oliveira, G. Airea. *Analizador lógico de 100 MHz utilizando FPGA*. Universidad de la República. Montevideo. http://iie.fing.edu.uy/~geirea/analiza.pdf. 2003. Consultado el 10 de febrero de 2004.

9. Xilinx Inc. *Handheld Pocket Logic Analyzer*. 2001. p. 21. http://direct.xilinx.com/bvdocs/appnotes/xapp368.pdf. Consultado el 20 de agosto de 2003.