

Metodología de diseño de sistemas digitales con reconfiguración parcial dinámica sobre *FPGA* de bajo costo

A design methodology for digital systems using partial run-time reconfiguration for low-cost FPGA

Recibido: diciembre 4 de 2012
Aprobado: diciembre 11 de 2012

Víctor Bastidas*, Sebastián Eslava**

Resumen

En este documento se presenta una metodología para el diseño de sistemas digitales usando reconfiguración parcial dinámica basada en diferencias en *FPGA* de la familia Spartan de Xilinx. Se aborda el problema de diseño en alto nivel de abstracción y se muestran las herramientas disponibles para traducir el diseño en un circuito funcional sobre una plataforma autorreconfigurable.

Palabras clave

Block RAM, *FPGA*, *hardware* reconfigurable, reconfiguración parcial dinámica, Select-MAP, sistema digital.

Abstract

This paper presents a design methodology for digital systems using difference-based run-time partial reconfiguration on low-cost Xilinx Spartan *FPGA* family. High-level abstraction design issues are described and available tools for translating design in a functional circuit using a self-reconfigurable *hardware* platform are shown.

* Ingeniero Electrónico. Estudiante de Maestría en Ingeniería. – Automatización Industrial, Departamento de Ingeniería Eléctrica y Electrónica, Universidad Nacional de Colombia Bogotá, Colombia.Email: vbastidasa@unal.edu.co

** Ingeniero Electricista, Magíster en Ingeniería Eléctrica Departamento de Ingeniería Eléctrica y Electrónica, Universidad Nacional de Colombia, Bogotá, Colombia.Email: jseslavag@unal.edu.co

Keywords

Block RAM, digital system, FPGA, reconfigurable *hardware*, run-time partial reconfiguration, SelectMAP.

I. Introducción

El *hardware* reconfigurable, usado como plataforma para el diseño de sistemas digitales, permite realizar modificaciones del comportamiento lógico de un circuito usando Lenguaje de Descripción de Hardware (en adelante, *HDL* por su acrónimo en inglés *hardware Description Language*). Esta característica, denominada reconfiguración, incrementa la flexibilidad de un sistema digital y otorga al diseñador una herramienta fuerte para modificar, mejorar y optimizar su diseño original [1].

El concepto de flexibilidad de diseño adquiere un nivel superior cuando se utiliza Reconfiguración Parcial Dinámica (en adelante, RPD). Este tipo de reconfiguración aprovecha la capacidad de ciertas familias de *FPGA* (*Field Programmable Gate Array*) de modificar la función de regiones seleccionadas, en cualquier momento después de su configuración inicial [2].

Se han hecho algunos estudios [3], [4], [5], [6], [7] que han podido traducirse exitosamente en diseños de *hardware* que son capaces de reutilizar recursos que solo son usados ocasionalmente en la operación de un sistema digital. La reutilización de estos recursos de *hardware* permite hacer mejoras importantes en términos de reducción de espacio, aumento de la velocidad de operación y disminución del consumo de potencia eléctrica [1].

No obstante, la posibilidad de mejoras en la operación de un sistema digital, la RPD no tiene un uso demasiado extendido y varias aplicaciones que podrían aprovechar sus características no hacen uso de ella, principalmente, porque exige un conocimiento profundo de la arquitectura y la tecnología del *hardware* reconfigurable que se esté usando. Es también una limitante importante el hecho de que el flujo de diseño de la RPD solo esté soportado y bien documentado en los *FPGA* de las familias Virtex, Artix y Kintex [1], [8] (en lo referente al fabricante Xilinx, únicamente), cuyos precios son más altos.

La RPD puede usarse en un sistema digital de dos formas: basada en diferencias y basada en módulos. La RPD basada en diferencias se enfoca en las estructuras primitivas de *hardware* contenidas en el *FPGA* y permite la modificación de sus atributos. La RPD basada en módulos permite modificar, insertar o retirar regiones más grandes de *hardware* con su descripción funcional.

En este documento se presenta el desarrollo de una metodología de diseño enfocada en aplicar RPD basada en diferencias, en la familia de *FPGA* de bajo costo Spartan de Xilinx. No se ha incluido la RPD basada en módulos debido a que su implementación requiere de una arquitectura diferente que solo esta disponible en los *FPGA* Spartan-6 o superiores. La presentación de esta metodología se complementa con la descripción de las herramientas disponibles y necesarias para traducir un diseño en un circuito digital funcional y, como prueba de concepto, se muestra el desarrollo completo de un ejemplo de circuito que aprovecha las propiedades de la RPD basada en diferencias.

Al no existir un flujo de diseño estándar para trabajar con RPD en *FPGA* de la familia *Spartan*, el desarrollo de la metodología de diseño que se presenta introduce la posibilidad de construir sistemas digitales para *hardware* reconfigurable con las prestaciones de la RPD y la accesibilidad de los *FPGA* de bajo costo.

La información contenida en este documento se distribuye así: en la sección dos se presenta el análisis y la metodología propuesta para establecer un flujo de diseño completo de un sistema digital usando RPD en *FPGA* de bajo costo. En la sección tres se detalla el diseño de un circuito en el que se usa la metodología mencionada anteriormente. En la sección cuatro se presentan las conclusiones y el trabajo futuro.

II. Metodología de diseño

En la Figura 1 se aprecia un diagrama del procedimiento de diseño constituido por siete etapas en alto nivel de abstracción que son la base de la metodología para construir un sistema digital con RPD basada en diferencias.

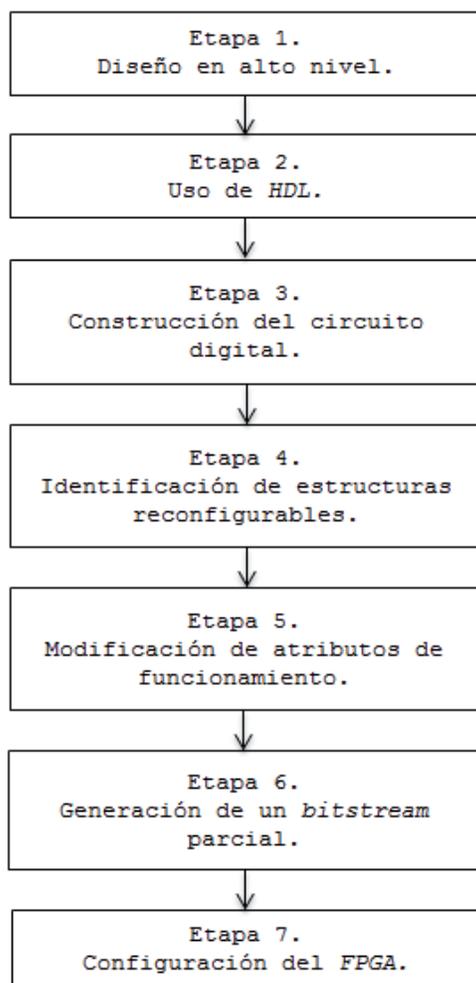


Figura 1. Siete etapas de diseño en alto nivel para construir un sistema digital con RPD basado en diferencias, sobre un *FPGA* de bajo costo.

II-A. Uso de las herramientas de diseño

A continuación se *explica* cómo se usan las herramientas de *software* en cada una de las etapas de diseño:

II-A1. Diseño en alto nivel: el proceso de diseño inicia con una idea de un sistema digital en el que se incluyen regiones reconfigurables dinámicamente, cuyas funciones pueden predefinirse con claridad. Se espera llegar al punto de obtener estructuras primitivas físicas (se debe considerar que, dependiendo de la arquitectura de *FPGA*, las modificaciones mínimas pueden estar restringidas a regiones más grandes), principalmente *slices*, que se encuentran dentro de cada celda lógica de un *FPGA*.

Los circuitos digitales que pueden aprovechar las características de la RPD en *FPGA* de bajo costo se centran generalmente en modificaciones de las funciones lógicas atribuidas a cada *slice* o al comportamiento predeterminado de una estructura primitiva, por ejemplo, un *Digital Clock Manager* (en adelante, *DCM*) o un multiplicador. Los sistemas que funcionan con base en señales de reloj variables pueden aprovechar también esta característica.

II-A2. Uso de HDL: cuando se tiene una idea clara del sistema que se va a diseñar, es conveniente realizar una primera división, muy específica del sistema: en aquellas regiones estáticas, que no van a modificarse después de la primera configuración, puede usarse cualquier tipo de descripción de *hardware*; en las regiones diseñadas para reconfigurarse dinámicamente, es altamente recomendado el uso de la descripción estructural (en bajo nivel) de cada módulo.

Las consideraciones en el tipo de descripción mencionadas recientemente son de gran ayuda porque en el proceso de Síntesis se pueden traducir directamente a *hardware*, estructuras bien definidas, con ubicación y conexionado fijo. Es entonces mucho más fácil acceder posteriormente a realizar modificaciones en los atributos lógicos de dichas estructuras.

II-A3. Construcción del circuito digital: Al finalizar la etapa de diseño *HDL*, el proceso de Síntesis permitirá observar el esquemático RTL que debe contener las estructuras reconfigurables. Si todo está bien, el siguiente proceso, la Implementación permitirá que el esquemático se traduzca en *hardware*. En este punto se debe prestar atención al proceso de *Place & Route*. Los resultados de este proceso pueden apreciarse con la herramienta de *software FPGA Editor* de *Xilinx Tools*.

II-A4. Identificación de estructuras reconfigurables: *FPGA Editor* ofrece una vista con la lista de componentes que han sido exitosamente ubicados y conectados dentro del *FPGA*; se debe buscar en esta lista aquel componente que se desee modificar. Facilita el trabajo de identificación, el hecho de que cada estructura conserve el mismo nombre con el que aparece en el esquemático RTL en el proceso de Síntesis.

II-A5. Modificación de atributos de funcionamiento: cuando se ha seleccionado la estructura que va a modificarse se puede entrar al modo de edición y revisar la información de sus atributos lógicos. Dependiendo de la estructura, se pueden modificar atributos como la función lógica, el periodo y la frecuencia de operación, el factor de multiplicación de frecuencia, la fase, las entradas activas, entre otros.

II-A6. Generación de un *bitstream* parcial: la estructura modificada debe guardarse como un diseño nuevo con un nombre diferente, con extensión *.ncd* (*Native Circuit Design*). Esto permitirá después, con la herramienta Bitgen, de *Xilinx Tools*, realizar una comparación entre el archivo de configuración *bitstream* original (*.bit*) y el diseño modificado (*.ncd*), para generar un nuevo archivo de configuración *bitstream*. Este archivo contiene solo información de las modificaciones, por lo cual es pequeño y notablemente más fácil de manejar que un *bitstream* completo.

II-A7. Configuración del *FPGA*: cuando se regenera el archivo *bitstream* completo, el archivo *bitstream* parcial queda contenido en él, lo que significa que se puede realizar la configuración del *FPGA* como se hace convencionalmente, por ejemplo vía *JTAG*. La información que se introduce en el dispositivo reconfigurable es suficiente para que en adelante se auto-reconfigure, sin necesidad de ninguna conexión externa.

El procedimiento completo mencionado anteriormente puede repetirse varias veces. La *RPD* no está limitada a una sola modificación, por lo cual cada *bitstream* parcial nuevo puede contener modificaciones aplicables a la misma estructura, o incluso, a estructuras nuevas.

II-B. Requerimientos de *hardware* para *RPD*

La *RPD* puede implementarse en dispositivos *FPGA* de bajo costo a través del modo *SelectMAP* (también conocido como *Slave Parallel*). *SelectMAP* es un modo de configuración que permite conectar hasta tres dispositivos Xilinx y posibilita su lectura y escritura en paralelo [10]. La Figura 2 muestra las conexiones entre la interfaz *SelectMAP* y los periféricos necesarios para controlar el proceso de *RPD*.

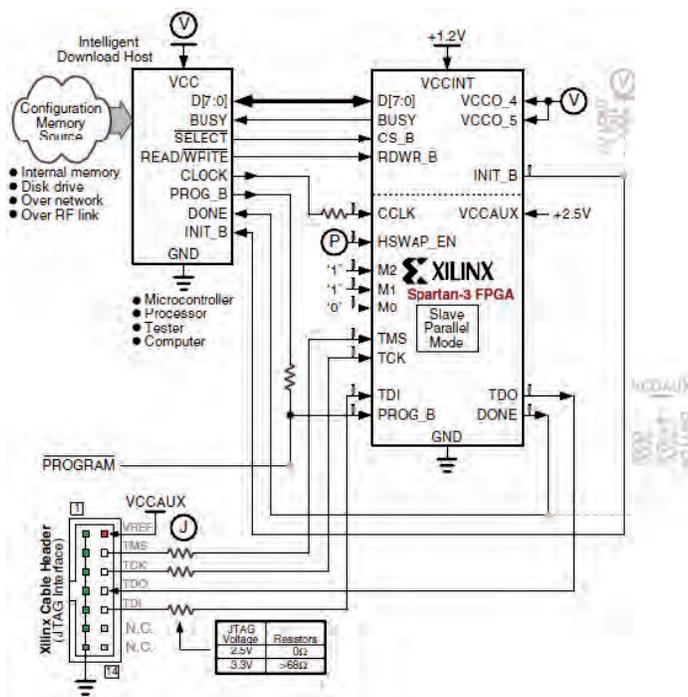


Figura 2. Esquemático del conexionado de *SelectMAP* para un *FPGA* XC3S200. Este esquemático está diseñado originalmente para ser controlado mediante periféricos. Fuente: tomado de [9].

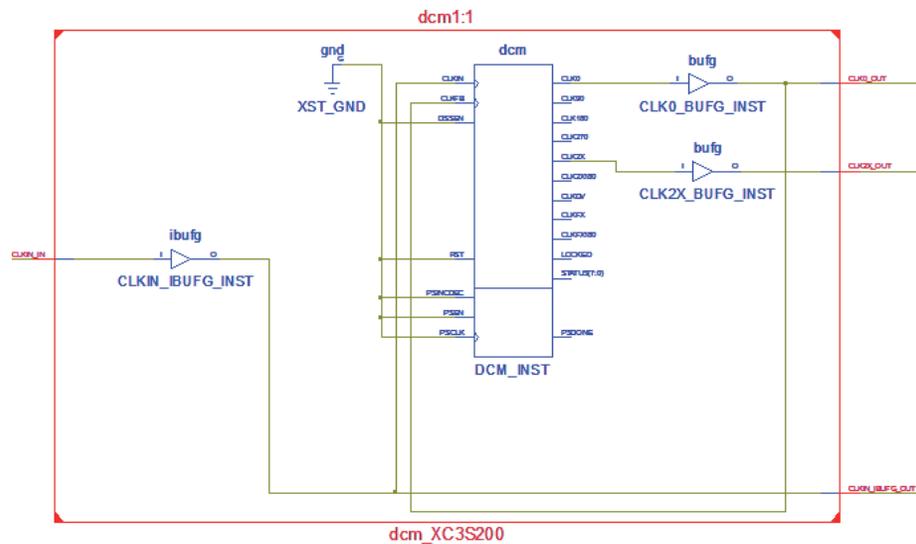
Los *bitstreams* parciales previamente generados deben convertirse a un formato adecuado, y posteriormente, deben almacenarse en una memoria. El Gestor de RPD, que puede ser un procesador, un microcontrolador, una PC o, incluso, un módulo diseñado internamente en un *FPGA*, es responsable de controlar la sincronización, la secuencia de inicio y finalizar la comunicación con la interfaz *SelectMAP*. Otras funciones del Gestor de RPD son generar señales de reloj adecuadas y transmitir cada *byte* desde la memoria hasta *SelectMAP*.

III. Utilización de la metodología

Con el fin de aplicar la metodología propuesta y como ejemplo de un módulo reconfigurable, se presenta el proceso completo del diseño de un gestor dinámico de señales de reloj (*Dynamic clock manager*), tomando como base un *FPGA Spartan-3*. La gestión dinámica de las señales de reloj es usada frecuentemente como un método para reducir el consumo de potencia en diseños de *hardware*, al cambiar la frecuencia de operación dependiendo de las necesidades.

El proceso de diseño empieza por identificar las estructuras adecuadas y disponibles en el *FPGA*. El *FPGA Spartan-3* de *Xilinx* dispone de cuatro módulos DCM. Estos módulos son estructuras fijas, con funciones claramente determinadas, entre las que se encuentra la variación de frecuencia. Es esta la parte reconfigurable del sistema; cualquier circuito puede aprovechar la variación de frecuencia de este módulo, por lo tanto, el diseño se limitará solo al módulo DCM.

Un módulo DCM puede instanciarse usando descripción estructural y el proceso de Síntesis resulta en el esquemático RTL que se muestra en la Figura 3. En este caso, se ha usado un *FPGA XC3S200*.



Después de realizar el proceso de Implementación, se usa *FPGA Editor* para apreciar los resultados de *Place & Route* y encontrar el componente *dcm_XC3S200*. Cuando se activa el modo de edición, los atributos del módulo DCM pueden visualizarse, como se muestra en la Figura 4.

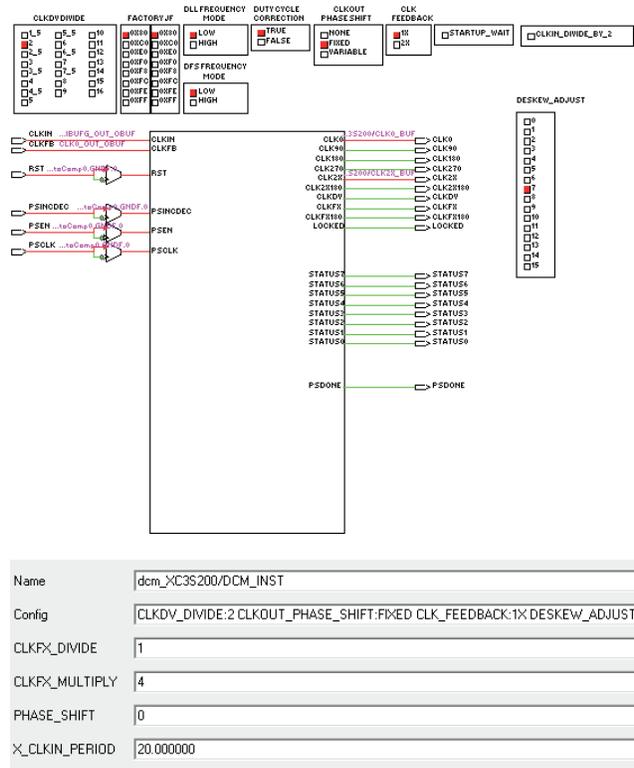


Figura 4. Atributos modificables en uno de los cuatro módulos DCM disponibles dentro del FPGA XC3S200.

Si el atributo *X_CLKIN_PERIOD*, originalmente fijado en 20 ns, se cambia, la frecuencia (o el periodo) puede modificarse a cualquier valor menor que la frecuencia inicial de 50 MHz. Cada atributo puede modificarse las veces que sea necesario y pueden generarse, también, varios *bitstreams* parciales.

III-A. Sustitución de periféricos por módulos de hardware

En la Figura 2 se muestran las conexiones de la interfaz *SelectMAP* con los periféricos necesarios para controlar el proceso de RPD. El concepto del sistema auto-reconfigurable es que ningún periférico sea necesario y que el *FPGA* tenga la posibilidad de controlar su propia reconfiguración. Dadas sus características, el *FPGA* puede ser aprovechado para sustituir a estos periféricos de la siguiente forma:

III-A1. Memoria: dentro de los *FPGA* de la familia Spartan se encuentran conectados físicamente varios bloques de memoria que no son utilizados con mucha frecuencia debido a su tamaño reducido. Esta memoria, conocida como *Block RAM*, a diferencia de una celda lógica solo tiene la función de almacenamiento, no tiene funciones adicionales. Por lo anterior, considerando el tamaño de un archivo de configuración *bitstream* parcial de un *FPGA*, la *Block RAM* puede destinarse exclusivamente para este fin.

al circuito Gestor de RPD, una *Block RAM* debidamente diseñada y dimensionada, y que dentro de esta se ubique adecuadamente la información de los archivos *bitstream* parciales.

III-B. Tratamiento de los archivos de configuración

Un archivo de configuración completo contiene la información total de las conexiones, ubicación y de la lógica que se necesitan para que cierto diseño pueda construirse dentro de un *FPGA*. Este archivo *bitstream* tiene un tamaño fijo para cada *FPGA*, independientemente de que el diseño sea una sola compuerta lógica o algo muy complejo [9]. En el caso del *FPGA XC3S200* el archivo de configuración completo tiene un tamaño de 1047616 *bits*, demasiado grande para almacenarse en la *Block RAM*. Sin embargo, al trabajar con archivos *bitstream* parciales, cuya información se limita a las diferencias con respecto a un *bitstream* original, su tamaño es mucho menor.

El primer paso consiste en obtener un *bitstream* parcial. La herramienta Bitgen genera un archivo *bitstream* parcial comparando el archivo *.bit* original y el archivo *.ncd* modificado después de la etapa de Implementación. Bitgen está disponible desde *TCL Console*, por lo tanto se hace necesario introducir el siguiente comando:

```
bitgen -w -g ActiveReconfig:Yes -g Persist:Yes -g -r original.bit modificado.ncd
```

El resultado es el archivo *bitstream* parcial *modificado.bit*, mucho más pequeño que el archivo original, pero al estar basado en este último, no afecta a la configuración inicial en regiones no modificadas.

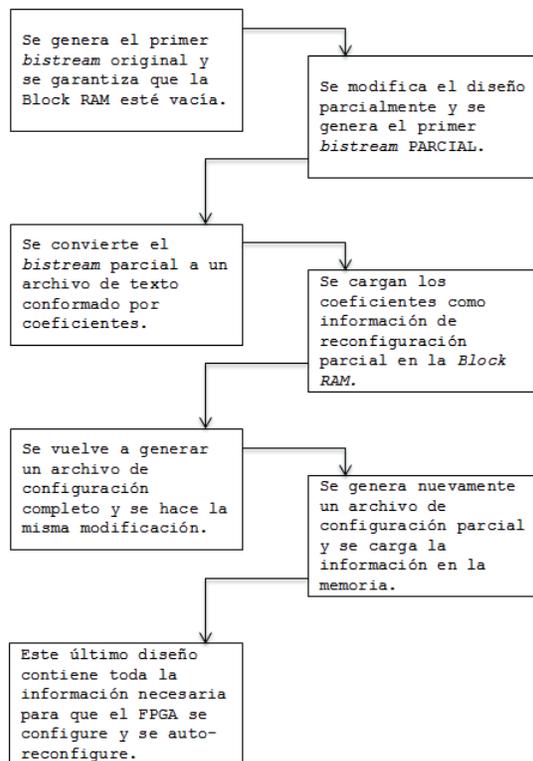


Figura 6. Procedimiento de seis etapas que debe usarse para generar un archivo *bitstream* parcial correcto.

Se requiere repetir el proceso de Implementación para evitar el problema de recurrencia que aparece por usar un *bitstream* completo que contiene un *bitstream* parcial.

Usando iMPACT de Xilinx Tools se puede convertir el archivo *bitstream* en un archivo hexadecimal con *byte-swapping*. Este procedimiento se hace necesario debido a que *SelectMAP* está diseñado para ser controlado con un procesador externo y se debe mantener coherencia entre los *bits* más y menos significativos de cada *byte*. Después, usando el *software Coegen 3.0*, desarrollado específicamente para este estudio por no existir ninguna herramienta similar disponible, se convierte el archivo *.hex* en un archivo *.coe* que puede cargarse directamente como información en *Block RAM*.

Un problema adicional que se debe solucionar antes de seguir con el diseño, es que se está tratando con un *bitstream* completo que está conteniendo un *bitstream* parcial; esto genera inicialmente un problema de recurrencia que se puede solucionar siguiendo el procedimiento de la Figura 6 que se modificó para adaptarlo a las necesidades del proyecto, con base en la información presentada en [7].

IV. Conclusiones y trabajo futuro

La metodología de diseño que se muestra en este documento permite estandarizar el proceso de construcción de un circuito digital con RPD basada en diferencias en *FPGA* de bajo costo. Aunque el diseño digital en alto nivel de abstracción sigue dependiendo de las habilidades del diseñador y exige que este tenga conocimientos profundos, tanto de sistemas digitales como de arquitectura de *FPGA*, la metodología aquí presentada ayuda a solucionar un problema crítico del diseño: definir las regiones reconfigurables y realizar modificaciones dinámicas con seguridad en *hardware* que no fue diseñado originalmente para usar RPD.

La plataforma autorreconfigurable requiere el uso de la memoria por bloques interna del *FPGA*. En este aspecto la mayor limitación es, por mucho, la cantidad de los *bitstreams* que pueden almacenarse internamente. Dependiendo del *FPGA* que se seleccione, es probable que su memoria interna no tenga capacidad mayor que la de almacenar un único archivo de configuración. Este no es un problema que pueda resolverse seleccionando un *FPGA* con una *Block RAM* más grande, puesto que un *FPGA* de mayor tamaño requiere de un *bitstream* de mayor tamaño también.

La etapa que presenta mayor dificultad al aplicar la metodología de diseño es la generación de *bitstreams* parciales. Fue necesario desarrollar *software* que completara el procedimiento y elaborar un flujo de diseño que se adaptara al tratamiento que se le estaba dando a los archivos de configuración. La solución a los inconvenientes de esta etapa, que se describe en la sección III-B garantiza la integridad de los archivos *bitstream* parciales.

El trabajo futuro se enfoca en mejorar el proceso de modificación de las estructuras primitivas específicas de cada arquitectura. Adicionalmente, se desarrollarán sistemas digitales más complejos para establecer en un espectro más amplio, las ventajas,

desventajas y limitaciones de la RPD basada en diferencias en dispositivos de bajo costo. Se estudiarán también las consideraciones de diseño de la RPD basada en módulos que permite diseñar regiones reconfigurables más grandes y complejas, usando principalmente su descripción funcional.

Referencias

- [1] Xilinx Corporation, *Partial reconfiguration user guide*, UG702, v12.1. Webpage: http://www.xilinx.comsupportdocumentation-sw_manualsxilinx12_1ug702.pdf. 2010.
- [2] L. Wang and F. Wu, Dynamic partial reconfiguration in FPGAs, vol. 2, pp. 445-448. *Third international symposium on intelligent information technology application*, 2009.
- [3] C. Inabla and K. Arshak, Using dynamic partial reconfiguration approach to read sensors with different bus protocol. *SAS IEEE Sensors Applications Symposium*. New Orleans, LA, USA, 2009.
- [4] F. Ghaffari, B. Miramond and F. Verdier, Dynamic adaptation of *hardware-software* scheduling for reconfigurable System-on-Chip. *The 19th IEEE/IFIP International Symposium on Rapid System Prototyping*. Monterrey, California, USA, 2009.
- [5] J. Cardoso, J. Basilio Simoes, C. Correia, A. Combo, R. Pereira, J. Sousa, N. Cruz, P. Carvalho and C. Varandas, A high performance reconfigurable *hardware* platform for digital pulse processing, vol. 51, No. 3. *IEEE Transactions on nuclear science*, 2004.
- [6] H. Kopka and A. Yurdakul, A self-reconfigurable platform for general purpose image processing systems on low-cost Spartan-6 FPGAs. Disponible en: <http://www.cmpe.boun.edu.tr>.
- [7] H. Kopka and A. Yurdakul, Dynamic partial self-reconfiguration on Spartan-iii FPGAs via a parallel configuration access port (PCAP). Disponible en: <http://www.cmpe.boun.edu.tr>.
- [8] D. Dye, Partial reconfiguration of Xilinx FPGAs using ISE Design Suite, WP3742, v. 1.2. Disponible en: http://www.xilinx.comsupportdocumentationwhite_paperswp374_partial_Reconfig_Xilinx_FPGAs.pdf, 2012.
- [9] Xilinx Corporation, Spartan-3 generation configuration user guide. Extended Spartan-3A, Spartan-3E and Spartan-3 FPGA families, UG332, v. 1.6. Disponible en: http://www.xilinx.comsupportdocumentationuser_guidesug332.pdf, 2009.
- [10] Xilinx Corporation, Disponible en: http://www.xilinx.comitpxilinx10isehelpm_c_introduction_SelectMAP.htm.



UNIVERSIDAD DE
SAN BUENAVENTURA
BOGOTÁ

CUPÓN DE SUSCRIPCIÓN

INGENIUM

Revista de la Facultad de Ingeniería

Fecha: Día Mes Año

Suscriptor

Apellidos

Nombres

Residencia

Dirección

Ciudad

País

Teléfono

Fax

Celular

E-mail

Suscripción anual

Colombia

\$ 120.000,00

Exterior

US \$ 70,00

Número suelto

Colombia

\$ 60.000,00

Exterior

US \$ 35,00

Suma enviada:

Suscripción a partir del número:

- Consigne en la cuenta de ahorros número 605-07721-3 del Banco de Bogotá, a nombre de la Universidad de San Buenaventura, Bogotá, D. C.
- Envíe copia del recibo de consignación, junto con este cupón, a la *Editorial Bonaventuriana* (Fax: 677 3003).
- Si lo prefiere presente el cupón y el recibo de consignación directamente en la *Editorial Bonaventuriana* (Cra. 8 H 172-20, Bogotá, D. C., Colombia).