Implementación de algoritmos de compresión de imágenes en FPGAs

Federico García Crespí[(1)], Otoniel M. López Granado[(1)], Juan José Abellán Pérez[(2)]

fedeg@umh.es, otoniel@umh.es, juan.abellan@graduado.umh.es

[(1)]Departamento de Física y Arquitectura de Computadores de la Universidad Miguel Hernández de Elche

[(2)]Ingeniero de Telecomunicación por la Universidad Miguel Hernández de Elche

Resumen

En aplicaciones tales como edición de vídeo profesional, cine digital, imágenes médicas, etc, la capacidad de almacenamiento, el ancho de banda en las comunicaciones, el tiempo de procesamiento y el consumo son parámetros críticos. Para el desarrollo de codificadores de imagen y/o video en hardware, se han venido utilizando generalmente los circuitos integrados específicos (ASIC) aunque han aparecido desarrollos de los algoritmos JPEG2000 y SPIHT en FPGA. En este artículo se presenta una comparativa de la implementación en FPGA, de varios algoritmos de compresión de imágenes: JPEG , JPEG 2000 y un nuevo algoritmo *Lower Tree Wavelet* (LTW).

1. Motivación

Dado que JPEG [1] y JPEG 2000 [2] son estándares, es interesante su implementación en dispositivos FPGAs para compararlos entre sí y conseguir mejorar sus rendimientos, el primero por ser uno de los algoritmos de compresión más usados hoy día y el segundo por ofrecer diversas mejoras, entre ellas una mayor calidad de imagen. Antes de JPEG 2000, aparecieron otros algoritmos de compresión basados en la DWT como, EZW [3] y, posteriormente una versión mejorada de éste, SPITH [4], pero finalmente fue JPEG 2000 el estándar adoptado. El mayor problema de JPEG 2000 es que es un compresor demasiado costoso computacionalmente y hace un uso demasiado alto de memoria. De hecho, se han realizado algunas optimizaciones hardware para acelerar el proceso de Post Compression Rate Dsitortion (PCRD) [5]. Con el objetivo de mantener la eficiencia en compresión pero reduciendo el coste computacional que necesita JPEG 2000, aparece LTW [6].

Para mejorar el rendimiento de JPEG y JPEG 2000 existen diversas implementaciones desarrolladas en FPGAs mediante lenguajes de descripción hardware como VHDL y Verilog. Algunas de ellas se explican en las siguientes citas bibliográficas: [7], [8], [9], [10]. En este trabajo se utiliza una herramienta de alto nivel, *CoDeveloper* [11] para realizar un diseño mixto hardware-software de los algoritmos JPEG, JPEG2000 y LTW.

2. Introducción

En los últimos años, con la aparición y expansión de Internet, ha sido necesario el desarrollo de sistemas computacionales que tengan un mejor desempeño en el tratamiento multimedia de imágenes y vídeo, siendo necesarios sistemas de alto rendimiento para procesar la información multimedia. El problema principal del tratamiento de la información multimedia es el tamaño y la lentitud de transferencia de dicha información por la red. Por todo esto, la compresión de imágenes adquiere una gran importancia, ya que permite reducir tanto el ancho de banda para transmitir imágenes como la memoria necesaria para almacenarlas.

Los algoritmos más usados para la compresión de imágenes requieren cada vez de un menor número de cálculos para su implementa-

ción, pero para los microprocesadores usados en computadores tradicionales, la implementación de estos algoritmos resulta una gran carga computacional. Una solución es el procesamiento en paralelo, por ello o se usa un procesador muy rápido o uno con varios núcleos, con el elevado coste que esto supone.

Para aplicaciones como la compresión y el procesado digital de imágenes existen dispositivos como las FPGAs, que han ido incrementando con los años sus capacidades en cuanto a velocidad, periféricos embebidos, número de componentes lógicos, memoria, DSPs e incluso procesadores embebidos, disminuyendo a la vez su tamaño y coste. Estos dispositivos tienen diversas ventajas respecto a otras soluciones como el software o los ASICs y son un compromiso entre estos dos. Una de las grandes ventajas es que son reprogramables, lo que añade una enorme flexibilidad, haciendo que los costes y tiempo de desarrollo sean menores como se muestra en la comparativa de la figura 1. Por tanto, las FPGAs proporcionan los medios adecuados para el procesado de imágenes, puesto que son dispositivos que pueden desarrollar un gran poder de cálculo sin necesidad de utilizar un computador. Las las FPGAs se presentan como una solución intermedia entre el uso de microprocesadores de propo?sito general y el disen?o de circuitos especi?ficos (ASIC). En una situacio?n ideal las FPGAs combinan lo mejor de las dos soluciones anteriores: la flexibilidad del software que se ejecuta en un microprocesador de propo?sito general y la velo- cidad del hardware especi?fico [12].

Se pueden encontrar ejemplos de codiseño hardware-software del algoritmos de compresión como [13] y [14]. En el primero, se implementa el algoritmo JPEG2000 en un sistema mixto DSP-FPGA, aunque se limita el ancho de banda y tiene un consumo mayor que un circuito integrado. En el segundo, no se procesa el primer nivel de descomposición en la wavelet lo que eleva el ratio el ratio (R/D).

Para la realización de este trabajo, se han desarrollado en software los tres algoritmos de compresión de imágenes, utilizando para ello el lenguaje de programación `C`, el procesador software MicroBlaze [15] y el procesador hardware PowerPC 440 [16]. Posteriormente se han analizado los códigos fuente buscando qué partes de los mismos consumen más tiempo de ejecución para implementarlas en hardware sobre una FPGA. Así, mediante el codiseño hardware-software se han diseñado distintas arquitecturas y coprocesadores para mejorar el rendimiento de los algoritmos JPEG y LTW.

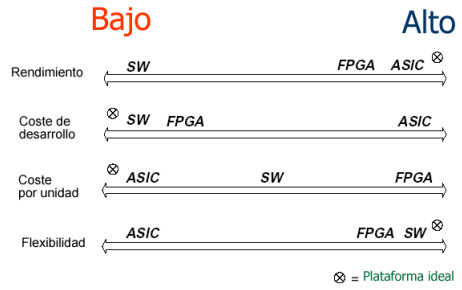

Figura 1: FPGAs vs ASICs vs Software

3. Herramientas

Las arquitecturas propuestas para ejecutar los algoritmos de compresión se han realizado en una placa de desarrollo *Digilent Genesys Virtex V* [17]. Para comprobar el rendimiento de los algoritmos se ha utilizado, en principio, el procesador *MicroBlaze*. Posteriormente, se utilizó de la placa *Avnet AES Virtex V FXT* [18] y se comprobó el rendimiento en el procesador *PowerPC*, incorporado en dicha placa.

Una vez se implementaron los diseños software de los algoritmos, se realizó una comparación entre ellos en cuanto a tiempo de computación (tanto en codificación como en decodificación) y en calidad (PSNR) para distintos tamaños de imágenes y tasas de bits. Además, se realizó un profiling (estudio) de los algoritmos con las herramientas proporcionadas por Xilinx, *EDK 12.1* e *ISE 12.1*, para averiguar cuáles eran los cuellos de botella de los mismos. Posteriormente, mediante el uso del lenguaje de programación *Impulse* `C` [19] y la herramienta de desarrollo *CoDeveloper* [11], se

implementaron en hardware las partes críticas del código, con lo que se consiguió aumentar el rendimiento de estos algoritmos.

4. Arquitecturas Propuestas

A continuación veremos las distintas arquitecturas que se han implementado para realizar las pruebas con los distintos algoritmos. Estas arquitecturas podemos clasificarlas como: software, hardware-software y duales.

4.1. Arquitecturas Software

Arquitectura MicroBlaze A

Esta arquitectura es la principal que se ha usado para implementar el algoritmo JPEG, así como JPEG 2000 (mediante el software de referencia JasPer) y LTW. A partir de esta arquitectura se proponen modificaciones para intentar conseguir un mejor rendimiento de los algoritmos. El diagrama de bloques simplificado la arquitectura básica es el de la figura 2a. Las características y los periféricos básicos utilizados que comparten todas las arquitecturas son los siguientes:

- Procesador MicroBlaze con una frecuencia de 125 MHz.
- Memoria DDR2 SDRAM de 256 MB conectada mediante el bus XCL.
- Caché de datos e instrucciones de 8 KB ubicada en la memoria SDRAM.
- Memoria BRAM de 8 KB para almacenar el bootloader que permite la carga de los diferentes programas en la memoria externa. Esta memoria está conectada al procesador mediante el bus LMB.
- Módulo MDM para poder realizar debug hardware.
- Memoria Flash de 32 MB para almacenar imágenes.
- Xps_timer para obtener los ciclos de reloj que tarda un programa en ejecutarse.
- Puerto RS232 para poder ver mediante un programa Hyperterminal la salida del programa.

Arquitectura MicroBlaze B

A partir de la arquitectura *MicroBlaze A* y mediante unas plantillas predefinidas en EDK, se propone la siguiente modificación. Las plantillas permiten mejorar el rendimiento del procesador MicroBlaze según las necesidades requeridas. En nuestro caso se ha utilizado la plantilla *Maximum Performance* para comprobar si se consigue aumentar el rendimiento de los algoritmos. Esta plantilla activa las siguientes mejoras hardware en MicroBlaze: Integer Multiplier, Float Point unit, Barrel Shifter, Integer Divider, I-Cache y D-Cache. El diagrama para esta arquitectura es el de la figura 2b.

Arquitectura PowerPC

La arquitectura *PowerPC* propuesta, utiliza una memoria BRAM para almacenar el *bootloader* y periféricos como el módulo RS232 UART, timer y la memoria DDR2 SDRAM. Para la implementación de esta arquitectura, en vez de usar la memoria flash, se usa el módulo Compact FLASH para leer datos del exterior, ya que se usa una placa distinta (placa Avnet) y ésta dispone de este módulo. El PowerPC no presenta ninguna característica de configurabilidad. Al contrario, dispone de un conjunto de unidades operacionales en hardware como multiplicadores, divisor etc., además de la caché de datos y la caché de instrucciones. En el caso de las cachés es necesario habilitarlas para que el procesador haga uso de ellas, éstas por defecto tienen un tamaño de 32 KB, y han sido configuradas para cachear la región de memoria que comprende la memoria SDRAM externa. Además se ha configurado la frecuencia de reloj al máximo permitido por este procesador, 400 MHz.

La arquitectura realizada mediante XPS (Xilinx Platform Studio) para este caso es la siguiente:

- Procesador PowerPC a 400 MHz, con una caché de datos e instrucciones de 32 KB con la unidad FPU activada.

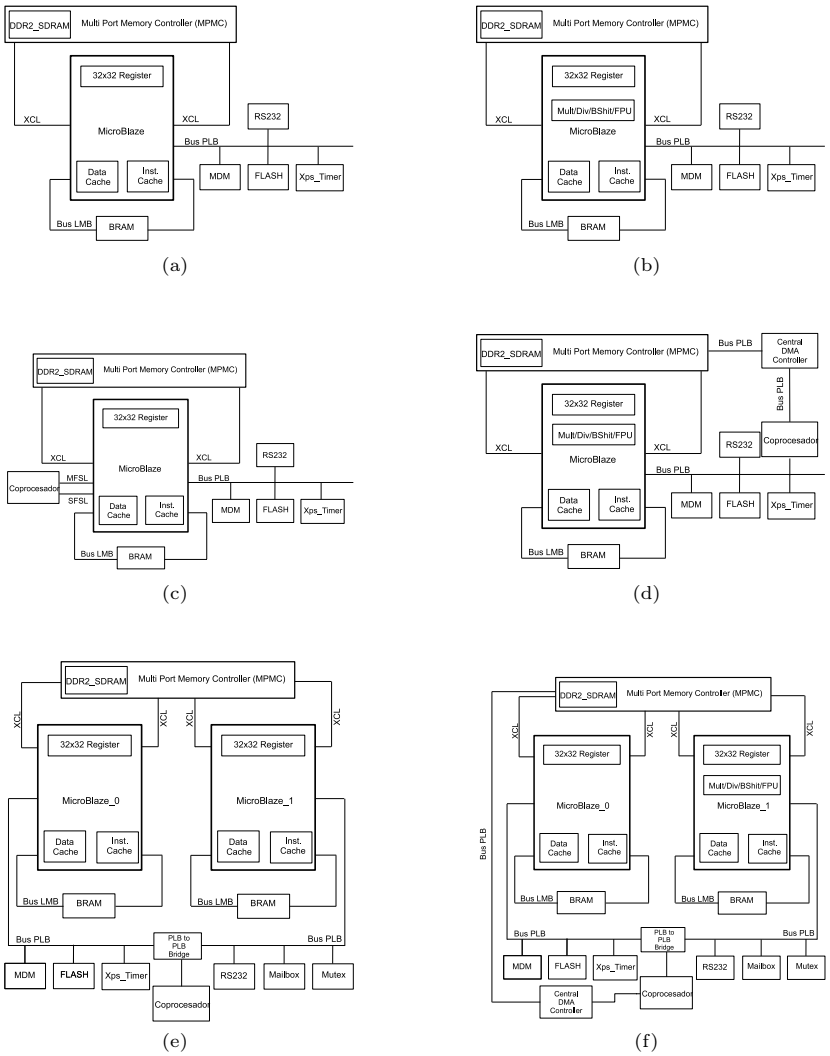

Figura 2: Diagramas de bloques de las arquitecturas implementadas: **(a)** MicroBlaze A, **(b)** MicroBlaze B. **(c)** MicroBlaze C. **(d)** MicroBlaze C'. **(e)** Dual MicroBlaze D. **(f)** Dual MicroBlaze D'

- Reloj de bus y reloj de referencia del sistema 100 MHz.
- Memoria SDRAM DDR2 de 64 MB.
- Módulo System ACE Compact Flash para almacenar las imágenes.
- Puerto RS232 para la salida de datos estándar.
- XPS Timer para tomar los ciclos de ejecución

4.2. Arquitecturas Hardware-Software

Arquitecturas MicroBlaze C y C'

Para intentar mejorar el rendimiento de los algoritmos se implementan dos arquitecturas. La arquitectura *MicroBlaze C* está pensada para el algoritmo JPEG. Ésta será muy similar a *MicroBlaze A* pero incluyendo un coprocesador hardware encargado de realizar la Transformada Discreta del Coseno (DCT). El diagrama de bloques obtenido tras realizar el diseño en CoDeveloper es el de la figura 3. *MicroBlaze C'* incluye un coprocesador encargado de realizar la Transformada Wavelet Discreta (DWT) para el algoritmo LTW, podemos ver el diagrama de bloques en la figura 4. Además incluye unas pequeñas modificaciones respecto a la arquitectura *C* que son:

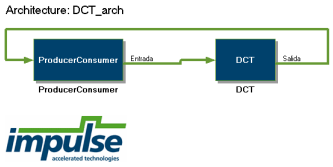

Figura 3: Diagrama de bloques del coprocesador hardware encargado de realizar la DCT

- Uso del perfil *Maximum Performance* en el procesador.
- El coprocesador se conecta mediante el bus PLB. El coprocesador leerá los datos directamente desde la memoria externa y CoDeveloper no permite generar este tipo de coprocesadores usando el bus FSL.
- Se ha incluido un nuevo periférico, *Central DMA controller*. Este dispositivo es un controlador de acceso directo a memoria que permite al coprocesador acceder directamente a la memoria del sistema.

En las figuras 2c y 2d vemos el diagrama de bloques para las arquitecturas *C* y *C'* respectivamente.

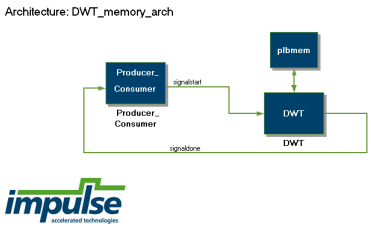

Figura 4: Diagrama de bloques del coprocesador hardware encargado de realizar la DWT

4.3. Arquitecturas Duales

Arquitecturas MicroBlaze D y D'

Se intenta mejorar al máximo el rendimiento de cada uno de los algoritmos. Por esto, se opta por usar una configuración con dos procesadores MicroBlaze.

Dual MicroBlaze D incluye dos procesadores similares a la arquitectura *A*, es decir, no incluyen ninguna de las mejoras que proporciona el perfil *Maximum Performance*. Además, esta arquitectura incluye un coprocesador específico para el algoritmo JPEG. Podemos ver el esquema de esta arquitectura en la figura 2e

En la arquitectura *D'* se han realizado unas pequeñas modificaciones respecto a la *D*. La primera de ellas es el uso del perfil *Maximum Performance* en el procesador *MicroBlaze_1*. La segunda es la de incluir el periférico, *Central DMA controller*. Esta arquitectura está pensada específicamente para el algoritmo LTW. En la figura 2f se muestra el diagrama de bloques.

5. Resultados obtenidos

5.1. Comparativa de tiempos

Con el fin de mostrar el rendimiento de cada arquitectura, se realiza una comparativa utilizado tres imágenes de prueba cuyos tamaños en pixeles son: 512×512 píxeles para la imagen Lena, 1024×1024 píxeles para la imagen Earth y 2048×2560 para la imagen Café. En la figura 5 se muestra la aceleración (Speed-Up) para cada arquitectura propuesta y para cada codificador en comparación con la arqui-

tectura MicroBlaze A. Aunque los resultados se presentan para la imagen Lena, un comportamiento similar se obtiene para el resto de imágenes. Como se muestra, se logra una gran mejora de rendimiento para las arquitecturas PowerPC y Dual MicroBlaze D/D', especialmente para JPEG2000 y LTW. Para la arquitectura dual MicroBlaze D' se obtiene una aceleración cercana a 50 para el codificador LTW.

Además en la tabla se muestra el promedio de aceleración obtenido para todas las imágenes de prueba. De nuevo, LTW supera al resto de codificadores evaluados, duplicando la aceleración según aumenta la complejidad de la arquitectura.

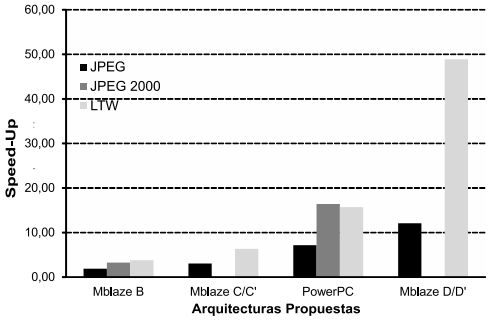

Figura 5: Speed-Up para la imagen Lena

Arquitectura/ Codificador	MBlaze B	MBlaze C/C'	PowerPC	MBlaze D/D'
JPEG	1,4	2,55	5,07	6,95
JPEG2000	2,19	-	14,77	-
LTW	2,63	4,02	10,24	29,59

Tabla 1: Aceleración media para todas las imágenes de prueba

Para ver de forma más clara las diferencias de tiempos entre algoritmos y arquitecturas, se ha creado la gráfica de la figura 6. En esta gráfica sólo se muestran los tiempos de codificación para la imagen *Lena* con una tasa de bit de 0,5 bpp, pero sirven para hacerse una idea general de todos los resultados comentados anteriormente.

Si hacemos una comparativa agrupando por arquitecturas:

- **Dual MicroBlaze D/D'**: Con estas arquitecturas sólo se han obtenido resultados para LTW y JPEG, siendo más rápido el primero.
- **PowerPC**: Para PowerPC sigue siendo más rápido LTW, en segundo lugar se encuentra JPEG y por último JPEG 2000.
- **MicroBlaze C/C'**: Usando un solo procesador y la ayuda del coprocesador sigue siendo más rápido LTW.
- **MicroBlaze B**: En este procesador se repite el mismo orden que en PowerPC, pero obteniendo peores tiempos que en él.
- **MicroBlaze A** Es el procesador más simple con el que se obtienen los peores tiempos, pero una vez más el algoritmo más eficiente es LTW, mientras que el más lento es JPEG 2000.

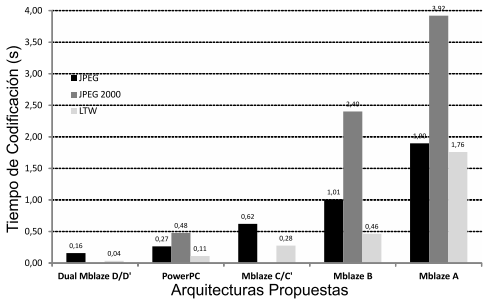

Figura 6: Tiempos de codificación para la imagen Lena

5.2. Comparativa de calidad

La calidad obtenida por cada algoritmo es independiente de la arquitectura en la que se ejecute, así, se ha realizado la tabla 2 en la que se encuentran los valores de la relación señal a ruido de pico (PSNR). Estos valores se han obtenido mediante la aplicación *imgcmp*, que se encuentra en las librerías JasPer. Los valores representados corresponden a una compresión con una tasa de bit fija de 0,5 bpp. Además, se han incluido en esta comparativa dos nuevas imágenes: Bárbara y Bike.

	PSNR(dB)				
	Lena	**Bárbara**	**Earth**	**Café**	**Bike**
LTW	37,35	32,49	38,43	26,86	33,28
JPEG 2000	37,22	32,83	38,23	26,79	33,5
JPEG	28,88	26,78	24,66	23,8	24,05

Tabla 2: Comparativa de la calidad de compresión

El algoritmo JPEG, incluso teniendo unas tasas de bit superiores, es el que peor calidad ofrece.

En el caso de JPEG 2000 y LTW, los valores del PSNR obtenidos son similares. Para las imágenes Lena, Earth y Café se obtiene una mayor calidad con LTW, mientras que para Bárbara y Bike, JPEG 2000 es algo superior. En cualquier caso, estos dos compresores están a la par en cuanto a calidad se refiere.

5.3. Comparativa de ocupación en las FPGAs

En cuanto al área ocupada, en la tabla 3 se encuentran los valores más característicos obtenidos tras sintetizar las arquitecturas en la placa Genesys. En la tabla 4 se muestra el tamaño ocupado por la arquitectura *PowerPC* en la FPGA de Avnet. La arquitectura que menos área ocupa es *MicroBlaze A*, siendo ésta la más sencilla, mientras que *MicroBlaze D'* es la que ocupa un mayor tamaño dentro de la FPGA. Se puede comprobar que cuanto mayor es la complejidad, el área ocupada por éstas es mayor.

	Slices	**BRAMs(36K)**	**DSP48E**
Tamaño Total	7200	60	48
MBlaze A	2298	19	3
MBlaze C(JPEG)	2736	47	9
MBlaze B	3419	34	6
MBlaze C'(LTW)	3786	43	16
DMBlaze D(JPEG)	5005	46	12
DMBlaze D'(LTW)	5671	52	22

Tabla 3: Áreas ocupadas en la FPGA Genesys por las arquitecturas MicroBlaze

	Slices	**BRAMs(36K)**	**DSP48E**
Tamaño Total	5120	68	64
PowerPC	2395	5	13

Tabla 4: Área ocupada por la arquitectura PowerPC en la FPGA Avnet

5.4. Discusión de los resultados

La arquitectura *MicroBlaze A* es la que menos área ocupa, por contra, es la que peor rendimiento ofrece. Los dos casos (para JPEG y LTW) con los dos procesadores MicroBlaze y el coprocesador hardware (*Dual MicroBlaze D* y *D'*), son los de mayor ocupación en placa, pero también los que ofrecen mayor rendimiento.

Si se desea implementar cualquiera de los algoritmos de compresión de imágenes vistos en este proyecto, es mucho más barato adquirir una FPGA con los periféricos necesarios para el funcionamiento de los mismos que un ordenador para realizar estos procesos específicos. Además, si se desease producir en serie alguna de las arquitecturas, cuanto menor sea el área ocupada por éstos, menor será el coste. Así que compararemos el área ocupada con el rendimiento ofrecido.

El procesador MicroBlaze con la configuración básica es el que menor área ocupa, pero ofrece el peor rendimiento. La ventaja de usar este procesador es que sirve para cualquier algoritmo. Si le añadimos las mejoras que ofrece el perfil *Maximum Performance* (arquitectura *MicroBlaze B*), el rendimiento del mismo aumenta, aumentando también el área ocupada por éste.

Si hablamos de las arquitecturas donde se usa un coprocesador, para JPEG (arquitectura *MicroBlaze C*) tenemos que la arquitectura implementada es menos compleja en cuanto a número de slices que *MicroBlaze B*, pero es mayor en cuanto al uso de BRAMS y DSP48E, obteniéndose además un mayor rendimiento. Para el caso de LTW (arquitectura *MicroBlaze C'*), a parte de usarse la versión de MicroBlaze más compleja, el coprocesador también aumenta el número de componentes lógicos. En estos dos casos se obtiene una mejora de rendimiento con respecto al procesador

básico.

Si hacemos uso de dos procesadores MicroBlaze y la ayuda de un coprocesador, se obtiene el mejor redimiento, pero los componentes lógicos utilizados para implementar estas arquitecturas son elevados, llegando a ocupar gran parte de la FPGA.

Por último, tenemos la implementación realizada en la placa Avnet usando el procesador PowerPC. Este procesador ofrece un compromiso entre área ocupada y rendimiento.

Hay que tener en cuenta que las arquitecturas *PowerPC*, *MicroBlaze A* y *MicroBlaze B* serían validas para implementar cualquier algoritmo. *MicroBlaze C/C'* y *D/D'* sólo pueden beneficiarse de la aceleración hardware cuando ejecutan los algoritmos para los que han sido diseñados, por lo que son menos versátiles.

Para según que aplicaciones y necesidades en cuanto a velocidad y coste, sería fácil seleccionar la arquitectura idónea basándonos en todos los resultados obtenidos y en todo el abanico de posibilidades que se han mostrado.

6. Conclusiones

En este trabajo hemos presentado el desarrollo en FPGA de tres tipos de codificadores de imágenes (JPEG, JPEG200 y LTW). Hemos seguido un diseño top-down para realizar la arquitectura de implementación, desde un diseño simple con un microprocesador (Microblaze A) hasta diseños mas complejos con dos procesadores y coprocesadores hardware (Microblaze D and D').

La arquitectura MicroBlaze A es la que menos área ocupa en la FPGA pero es la que peor rendimiento ofrece. Por otro lado, las arquitecturas MicroBlaze D y D', son las que mejor rendimiento ofrecen. Los sistemas mixtos hardware-software, MicroBlaze C y C', requieren entre un 6 % y un 20 % más de área extra respecto a MicroBlaze A para implementar los aceleradores hardware.

Las arquitecturas basadas en PowerPC ofrecen una buena relación entre el área ocupada y la velocidad. Con un número similar de Slices que el la arquitectura MicroBlaze A pero utilizando menos módulos BlockRam, la arquitectura con PowerPC ofrece una mejora en los tiempos de ejecución.

Utilizando esta comparativa, dependiendo de la aplicación final y de los requerimientos (velocidad de codificación, coste de la plataforma y área ocupada) se puede elegir entre los diversos sistemas presentados en este trabajo.

Referencias

[1] JPEG Standard. *JPEG ISO/IEC 10918-1 ITU-T Recommendation T.81.*

[2] *JPEG2000 part I final int. std.*, September 2000.

[3] J.M. Shapiro. A fast technique for identifying zerotrees in the ezw algorithm. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Vol. 3:pags. 1455–1458, 1995.

[4] A. Said and A. Pearlman. A new, fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits, Systems and Video Technology*, vol. 6, no. 3:pags. 243–250, 1996.

[5] Y.M. Yeung and O.C. Au. Efficient rate control for jpeg2000 image coding. *Circuits and Systems for Video Technology, IEEE Transactions on*, 15(3):335 – 344, march 2005.

[6] J. Oliver and M.P. Malumbres. Low-complexity multiresolution image compression using wavelet lower trees. *Circuits and Systems for Video Technology, IEEE Transactions on*, 16(7):1437 – 1443, november 2006.

[7] J. Rosenthal. *JPEG image Compression using an FPGA*. PhD thesis, University of California, 2006.

[8] J. Ahmad and M. Ebrahim. Fpga based implementation of baseline jpeg decoder. *International Journal of Electrical & Computer Sciences*, Vol:9 No:9:371–377, 2009.

[9] Arun Kumar. *Implementation of Image Compression Algorithm using Verilog with Area, Power and Timing Constraints*. Department of Electronics and Communication Engineering National Institute Of Technology Rourkela, 2009.

[10] A. Descampe, Devaux F., Rouvroy G., Macq B., and Legat J.D. *An Efficient FPGA Implementation of a Flexible JPEG2000 Decoder for Digital Cinema*. PhD thesis, Université catholique de Louvain, 2002.

[11] Impulse Accelerated Technologies. *C to FPGA Tools. CoDeveloper*. `http://www.impulseaccelerated.com`.

[12] Jim Turley. Soft computing reconfigures designer options. In *Embedded Systems*, April 1997.

[13] Korkmaz I. Durna M. Kolçak T. Ismailoglu N., Benderli O. and Tekmen Y. A real time image processing subsystem: Gezgin. In *AIAA/USU Conference on Small Satellites*, 2002.

[14] Pasquale Corsonello, Stefania Perri, Paolo Zicari, and Giuseppe Cocorullo. Microprocessor-based fpga implementation of spiht image compression subsystems. *Microprocessors and Microsystems*, 29(6):299–305, 2005.

[15] Xilinx. *MicroBlaze processor reference Guide*, April 2010. UG081 (v11.0).

[16] AMCC. *PPC 440 Processor User's Manual*, March 2008. Rev. 1.09.

[17] Digilent. *Genesys Board Reference Manual*, February 2012. Rev. C.

[18] Avnet Electronics. *Xilinx Virtex 5 FXT Evaluation Kit - User Guide*, May 2008. Rev. 1.0.

[19] David Pellerin and Scott Thibault. *Practical fpga programming in c*. Prentice Hall Press, Upper Saddle River, NJ, USA, first edition, 2005.