DESARROLLO DE PROYECTOS HARDWARE DE VISIÓN Y PROCESAMIENTO DE SEÑAL DIGITAL SOBRE FPGAs

Sergio A. Cuenca Asensi, Jose M. Guerrero Romero, Emilio Tendero Esteve, Abel Gonzalez Onrubia

Universidad de Alicante, Escuela Politécnica Superior, Dpt. Tecnología Informática y Computación.
e-mail: sergio@dtic.ua.es

RESUMEN: En el presente trabajo se exponen una serie de herramientas y metodologías para la implementación en hardware de algoritmos relacionados con visión y procesamiento digital de la señal. Este tipo de prácticas resultan altamente motivadoras tanto en las asignaturas de visión y procesamiento de señal como en las relacionadas con el diseño de arquitecturas avanzadas, diseño VLSI o lenguajes de descripción hardware. Se proponen los dispositivos reconfigurables (FPGAs) como herramienta principal de implementación. Gracias a su gran versatilidad y facilidad de uso, esta tecnología esta desplazando cada vez más a los tradicionales ASIC, tanto en la industria como en los temarios de las asignaturas relacionadas.

1.- INTRODUCCIÓN.

Las tecnologías reconfigurables (principalmente FPGAs y CPLDs) se están convirtiendo en una alternativa real a las implementaciones tradicionales de algoritmos computacionalmente intensivos. En algunos sistemas las FPGAs pueden sustituir eficientemente a microprocesadores de propósito general o procesadores DSP [1], en otros la lógica programable trabaja junto con estos a modo de coprocesador liberándolos de ciertas tareas masivamente paralelas para que puedan atender otras tareas de control [2]. Por otro lado la tendencia a la baja del precio de estos dispositivos así como la aparición de nuevas familias de dispositivos de bajo coste (p. ej: serie SpartanII de Xilinx), ha convertido a las FPGAs en una seria alternativa al tradicional diseño de circuitos full-custom y semi-custom.

Los estudios de informática no son ajenos a esta tendencia de la industria, y en los últimos años se están incorporando estas tecnologías a las prácticas de muchas asignaturas [3] relacionadas con arquitecturas avanzadas, diseño de circuitos, visión o procesamiento digital de la señal. Otro factor que juega a favor es el hecho de que las herramientas de trabajo y las metodologías son cada vez más parecidas a las utilizadas en el diseño de ASIC, por lo que las asignaturas de diseño VLSI y lenguajes de descripción hardware (HDL) también empiezan a incorporarlas a sus temarios [4].

En este trabajo se describen una serie de herramientas y metodologías para el diseño de prácticas y pequeños proyectos hardware relacionados con visión y procesamiento de señal. La implementación de estos proyectos, mediante lógica reconfigurable, generalmente queda vedada a la mayor parte de los alumnos, ya que supone el empleo de costosas tarjetas tipo PCI, cuyo uso suele reservarse a la investigación (por ejemplo Aristotle de Mirotech [5], o RC1000-PP de Embedded Solutions [6], con precios por encima de los 2.500$). Gracias a la utilización de las herramientas que proponemos, este tipo de proyectos se pueden realizar utilizando tarjetas de bajo coste lo que supone una alternativa realmente asequible, máxime pensando en un laboratorio de 10 a 20 puestos. En nuestro caso disponemos de la placa XS40 de la compañía Xess [7] (<100$), que actualmente se utiliza en la asignatura de Diseño de Circuitos Asistido por Computador en los estudios de informática de la Universidad de Alicante.

Con el desarrollo de estas herramientas se pretende pues, aumentar las posibilidades de esta plataforma en varios sentidos. Primero, suministrando un interfaz amigable que permita de forma sencilla y transparente llevar a cabo las tareas de configuración y comunicación. En segundo facilitando el control, la depuración y la verificación de los algoritmos desde el host. Y por último proporcionando una librería de controladores hardware que realice la comunicación entre los tres elementos básicos de todo el sistema, esto es, el host, la memoria de la tarjeta y el dispositivo reconfigurable. De esta forma el alumno podrá concentrarse en el diseño del núcleo del algoritmo, ahorrará tiempo en el desarrollo y se ampliarán notablemente las posibilidades de éxito de los proyectos.

2.- EL ENTORNO DE DESARROLLO.

La placa XS40 está basada en la FPGA XC4010 de Xilinx [8] y el microcontrolador 8031 de Intel. Dispone además de una pequeña memoria SRAM de 32Kx8, accesible tanto desde la FPGA como desde el micro. Su arquitectura es muy simple como muestra la figura 1; la mayoría de los pines de los puertos P1 y P3 del micro están conectados a la FPGA y pueden ser utilizados como entradas/salidas de propósito general. La placa incluye un oscilador de 12MHz conectado directamente a una entrada de la FPGA que controla a su vez la señal de reloj que recibirá la patilla XTAL1 del micro. Además dispone de un display de 7 segmentos y un conector VGA accesibles directamente desde la FPGA. La mayor parte de las patillas de la FPGA están accesibles mediante un par de arrays de pines en la parte inferior de la placa, por lo que resulta relativamente sencillo la conexión de circuitería externa.

La comunicación con el host y la configuración de la placa se realiza a través del puerto paralelo. Desde el host se pueden enviar datos utilizando los ocho bits del registro de datos (D0..D7) y cuatro bits del registro de control (C0..C3) de este puerto. La comunicación desde la placa se realizará utilizando únicamente cuatro bits del registro de estado (S3..S6).

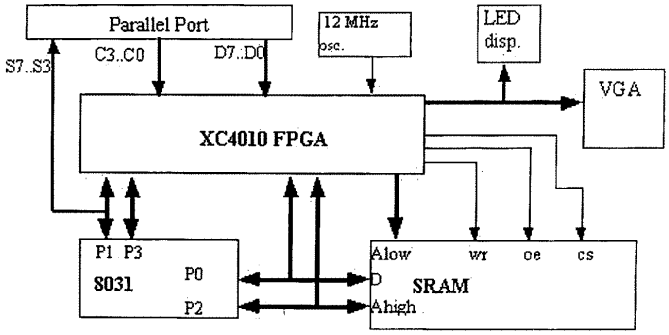

fig 1. Arquitectura de la placa XS40.

3.- DESARROLLO DE UN INTERFAZ DE PROTOTIPADO .

El objetivo principal de este interfaz es el prototipado rápido y la verificación funcional de los algoritmos mapeados en el hardware. Para conseguir esto las distintas tareas se reparten como se detalla a continuación. El host se encarga, por un lado, de las tareas de control y configuración del hardware, y por otro de la adquisición de los datos (imágenes y/o señales) y su visualización para la verificación funcional del procesamiento. Estas imágenes y/o señales pueden estar almacenadas en la memoria del host como ficheros de distintos formatos (bmp, tiff, pgm, wav,...). La FPGA, por su parte, se encarga del procesamiento del algoritmo utilizando los datos que le va transfiriendo el host o los datos almacenados previamente en la memoria SRAM de la placa. Conforme se van procesando estos, se envían de vuelta los resultados o bien se almacenan en la SRAM para su posterior volcado y visualización en el host. Esta metodología de trabajo permite al alumno concentrarse en el diseño y mapeado del algoritmo dejando para una etapa posterior opcional la incorporación de circuitería externa que permita la adquisición de los datos.

En esta línea de trabajo, la utilización del puerto paralelo para la comunicación con la placa supone una importante limitación a la hora de implementar aplicaciones con cierto grado de complejidad y en las que se necesite un trasiego masivo de datos entre el host y la FPGA. Esta limitación podría subsanarse utilizando el modo mejorado del puerto paralelo (EPP), sin embargo las líneas de datos D0 y D1 (fig.1) están conectadas a sendos inversores, lo que hace imposible su utilización en modo bidireccional. Es por tanto imprescindible desarrollar un interfaz a medida que permita la transferencia eficiente de datos en ambos sentidos. Este interfaz constará de una parte software, que implementa el protocolo de comunicación correspondiente al host, y una parte hardware que implementa la parte del protocolo correspondiente a la XS40.

a) Software

Se han diseñado una serie de funciones empaquetadas en una librería dinámica (DLL) que permite, mediante simples llamadas desde un programa C, realizar las principales tareas de comunicación bidireccional con la placa:

- test de los componentes.
- configuración de la FPGA.
- Control del procesamiento de la FPGA.
- Escritura/Lectura en la memoria SRAM de la XS40.
- Transferencia de bloques de datos hacia/desde la XS40.

Mediante esta librería el usuario puede crear un entorno amigable de comunicación e intercambio de datos con la placa.

b) Hardware

En este caso se trata de una librería de controladores hardware escrita en lenguaje VHDL, que permite fácilmente incorporar a los algoritmos los protocolos de comunicación y otras funcionalidades adicionales. Estos módulos ocupan muy pocos recursos de la FPGA (entre el 5% y 15% dependiendo de su complejidad), por lo que la funcionalidad de esta apenas se ve mermada.

Existen dos tipos de controladores (MC):

- Los "básicos" que implementan el protocolo mínimo de comunicación que permite controlar el procesamiento de la FPGA y la escritura y lectura de la memoria.
- Los de tipo "scan" que añaden a las funcionalidades básicas la posibilidad de controlar de forma transparente los accesos a la memoria SRAM siguiendo patrones típicos de los procesamientos de imagen y señal (acceso por filas, columnas, ventanas 3x3, etc..). Estos controladores permiten además trabajar con diferentes tamaños y resoluciones de imágenes y/o señales.

Utilizando esta librería, el usuario incorpora al núcleo del algoritmo o módulo de procesamiento (MP) el controlador apropiado y crea un circuito ("XS40 Core") totalmente preparado para comunicarse con el host (fig.2). Esta conexión entre el MC y el MP se lleva a cabo mediante una serie de líneas de control (control "residual") que especifican el tipo de operación a realizar: procesamiento, espera, lectura, escritura, etc... Este esquema de trabajo soporta cualquier tipo de arquitectura para la implementación del MP (multiciclo, segmentada, etc...).

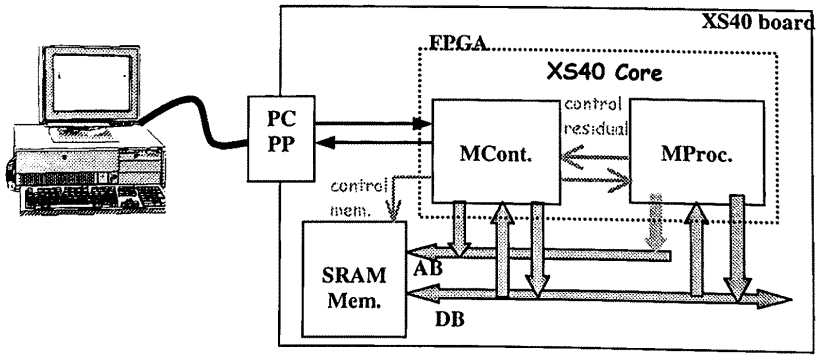

fig. 2. Esquema de trabajo.

4.- METODOLOGÍA Y FLUJO DE DISEÑO

En la figura 3 se puede apreciar el flujo de diseño a seguir para el desarrollo de un proyecto. Por un lado contamos con el software Foundation de Xilinx, que consta de una serie de herramientas que permiten realizar el ciclo completo de diseño de la FPGA. Incorpora, además de la típica captura de esquemáticos, varios de los lenguajes de descripción hardware más extendidos (VHDL, Verilog y Abel). Utilizando cualquiera de estos se lleva a cabo la entrada del circuito que implementará el algoritmo. A continuación se incorpora uno de los controladores (especificado en VHDL) y se llevan a cabo las simulaciones funcionales y temporales del circuito resultante. Una vez depurado el diseño se pasa a la fase de implementación (colocación, encaminamiento, etc...) que produce un fichero de configuración para la FPGA. Utilizando el entorno desarrollado mediante la librería software, se configura y verifica el circuito y por último se realiza una verificación funcional del algoritmo mediante el envío de imágenes y/o señales desde el host y la posterior visualización de los resultados.

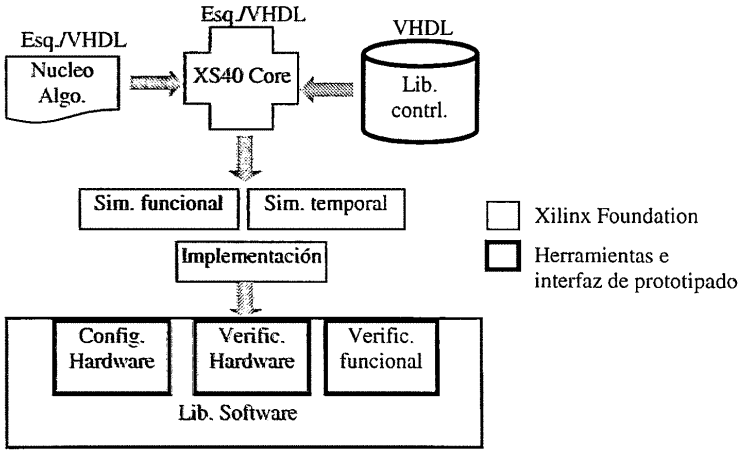

fig 3. Flujo de diseño.

5.- IPXS: ENTORNO PARA EL DESARROLLO DE PROYECTOS DE PROCESAMIENTO DE IMAGEN.

Siguiendo esta filosofía de trabajo y utilizando las librerías de prototipado, se ha desarrollado un entorno para el desarrollo de proyectos de procesamiento de imagen (IPXS). Utilizando este entorno visual (fig. 4) el usuario puede configurar la FPGA y comprobar su funcionamiento mediante un sencillo interfaz gráfico de generación de señales del puerto paralelo. Del mismo modo, puede abrir ficheros de imagen, visualizarlos, cambiar la resolución de los pixels y enviar la imagen a la memoria SRAM de la XS40 para ser procesada por la FPGA. El procesamiento se inicia desde el mismo entorno y una vez terminado este automáticamente se lee la imagen resultado y se visualiza. Existe además la posibilidad de comprobar la funcionalidad del algoritmo repitiendo localmente el procesamiento en el host mediante un programa C y comparando los resultados.

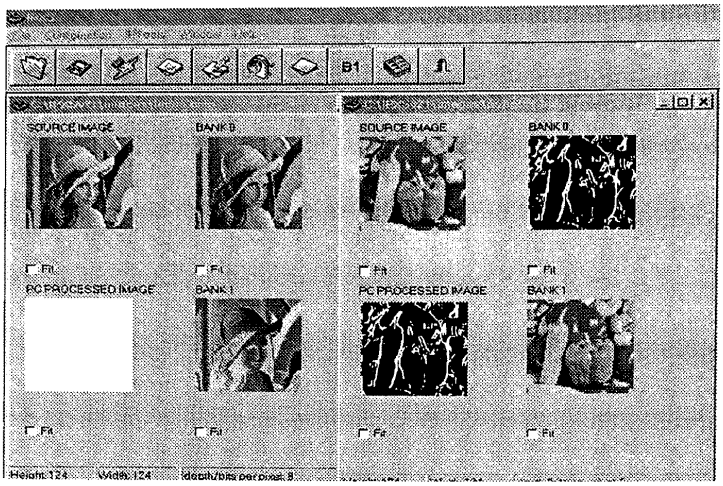

fig 4. Entorno IPXS

Al mismo tiempo se ha desarrollado una librería de controladores que realizan de forma transparente los accesos a la imagen siguiendo patrones típicos de visión: acceso por filas, acceso mediante ventanas 2x2, 3x3,4x4, 5x5, etc... (fig. 5). Pueden además, soportar distintas resoluciones de los pixels (1bit, 2bits, 4bits, ...). Para este tipo de proyectos se considera la memoria SRAM (32Kx8) dividida en dos bancos, B0 y B1 de 16Kx8, con lo que es posible trabajar con imágenes de hasta 124x124 pixels en 256 niveles de gris. Ambos bancos se utilizan indistintamente para almacenar la imagen original o el resultado del procesamiento, pudiéndose realizar procesamientos iterativos simplemente intercambiando los papeles.

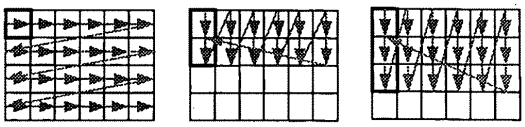

fig 5. Patrones de acceso: por filas, ventana 2x2 y 3x3

En la figura 4 se puede apreciar el resultado de los procesamientos mediante las operaciones NOT y SOBEL , mostrándose en cada caso el contenido de ambos bancos. En el último se muestra además el resultado del procesamiento local.

El entorno y la librería de controladores se encuentran actualmente disponibles en la dirección: www.dtic.ua.es/dtic.local/asignaturas/DCAC/project/IPXS21.htm

6.- CONCLUSIONES

Se ha presentado una serie de herramientas y métodos de trabajo para facilitar la labor de los estudiantes en el desarrollado de prácticas y proyectos de visión y procesamiento de imagen basados en dispositivos programables (FPGAs). Gracias a estas herramientas se hace posible la utilización de una tarjeta de bajo coste para este tipo de proyectos, facilitando el acceso de los alumnos a prácticas más atractivas y motivadoras. Por último se presenta un entorno desarrollado sobre la base de esta filosofía para el prototipado de circuitos de procesamiento de imagen.

7.- AGRADECIMIENTOS

Este proyecto ha sido parcialmente financiado por la empresa Xess Corp.

8.- REFERENCIAS

[1] Goslin, G. R. "Using Xilinx FPGAs to Design Custom digital Signal Processing Devices". Proceedings of the DSP^X 1995.
[2] Nassif S. C. & Capson D.W. "A Flexible DSP-based Network for Real-Time Co-operative Windowing Applications". Real-Time Imaging. pp 283-293. 1997.
[3] Gómez Pulido, J. A. et. al " Actualización, Investigación y Desarrollo en el Entorno Docente de Arquitectura de Computadores en la Universidad de Extremadura". IV Jornadas sobre la enseñanza universitaria de la informática, Andorra 1998.
[4] Albadalejo, J. et al "Uso del VHDL en la enseñanza de Arquitectura de Computadores". IV Jornadas sobre la enseñanza universitaria de la informática, Andorra 1998.
[5] http://www.mirotech.com
[6] http://www.embeddedsol.com
[7] http://www.xess.com
[8] "The Programmable logic data book". Xilinx Inc. 1998.