LINUX SOBRE UNA FPGA

Linux running on FPGA

RESUMEN
En la actualidad, los sistemas embebidos han logrado un gran auge gracias a sus diferentes campos de aplicación y a sus bajos costos comparados con sistemas de cómputo tradicionales. En el presente documento se mostrará como montar un sistema operativo, denominado uClinux sobre un dispositivo FPGA (Field Programable Gate Array), en la cual debe implementarse un soft processor, en este caso Microblaze, un microprocesador de 32 bits creado por Xilinx. Todo esto con el fin de lograr optimizar los recursos de la FPGA y permitir un control más sencillo de los periféricos que sean conectados a la misma.

PALABRAS CLAVES: FPGA, Microblaze, sistema embebido, uClinux.

ABSTRACT
Currently embedded systems have obtained great heights thanks to their ample fields of application and, at it´s lower costs comparated with the tradicional computer systems. This document will show how to port an operative system called uClinux on an FPGA (Field Programable Gate Array), in which a soft processor must be designed, in this case; Microblaze microprocessor of 32 bits created by Xilinx. All this to optimize the resources in the FPGA and permit a simple control over the peripherals that been plug.

KEYWORDS: *Embedded system, FPGA, Microblaze, uClinux.*

JOSE ALFREDO JARAMILLO VILLEGAS
Ingeniero Electrónico
Pontificia Universidad Javeriana
jj@sirius.utp.edu.co

LINA MARIA PEREZ PEREZ
Estudiante Ingeniería en Sistemas y Computación
Universidad Tecnológica de Pereira
lp@sirius.utp.edu.co

JOHN HAIBER OSORIO RIOS
Estudiante Ingeniería en Sistemas y Computación
Universidad Tecnológica de Pereira
jo@sirius.utp.edu.co

ANDRES GUILLERMO VELASQUEZ GOMEZ
Estudiante Ingeniería en Sistemas y Computación
Universidad Tecnológica de Pereira
agvg@sirius.utp.edu.co

1. INTRODUCCIÓN

Los sistemas embebidos son sistemas especializados dedicados a una sola tarea, los cuales podemos llamar sistemas de propósito específico.

Por esto, en la actualidad, se ha adoptado un nuevo paradigma de diseño de bajo precio el cual ha mostrado gran eficiencia al ser dedicado a una tarea específica, dado que solo se diseñan e implementan los módulos que se van a utilizar y por tanto se usa el hardware estrictamente necesario. Además puede ser optimizado en cualquier momento ya que en la mayoría de las ocasiones, es implementado sobre dispositivos que pueden ser reprogramados, son portables y de tamaño reducido. Es así como los sistemas embebidos son la primera opción en el campo de la ingeniería para la solución de problemas específicos.

Si un sistema embebido ofrece grandes ventajas, estas serán mayores si se cuenta con un sistema operativo que le brinde al usuario una mayor facilidad a la hora de

trabajar y crear un sistema especializado mucho más robusto. Esto se ha logrado gracias a que se cuenta con uno de los sistemas operativos más estables, de mayor fuerza y en la actualidad más difundidos como lo es Linux. Este Linux (uClinux) es adaptado especialmente para ser portado a la arquitectura del procesador Microblaze de Xilinx.

2. MICROBLAZE

[1]Microblaze es un procesador con un set de instrucciones reducido (RISC) optimizado para la implementación en FPGAs. Este microprocesador es un IP core que debe instanciarse en el proyecto de hardware, sólo puede estudiarse a partir de la documentación del fabricante, de los documentos de resultados experimentales publicados en la red o de la experiencia propia del usuario. Aún así, es una herramienta muy potente para desarrollar proyectos relacionados con arquitecturas en paralelo, diseño, control y en general, cualquier investigación sobre software, ya que permite una comparación inmediata con otras arquitecturas y metodologías de desarrollo. MicroBlaze puede ejecutar diferentes sistemas operativos como Nucleus, ThreadX, uC/OS-II y uClinux.

2.1 ARQUITECTURA DEL MICROPROCESADOR

[2]MicroBlaze es un procesador de 32 bits el cual ha sido desarrollado con unos requisitos muy rígidos de

Fecha de Recepción: 07 Septiembre de 2007
Fecha de Aceptación: 19 Noviembre de 2007

ocupaciónes y prestaciones, debido a la limitación impuesta por los recursos disponibles en una FPGA. Sin embargo, esta catalogado [3] como uno de los soft processors más rápidos del mundo, según la revista de electrónica española Dialnet.

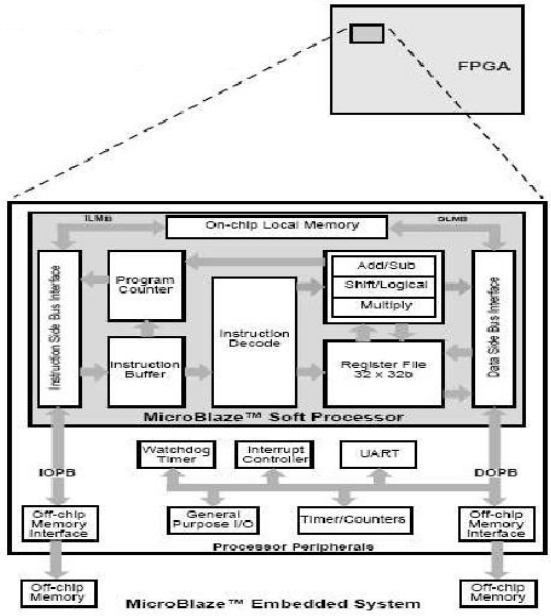

Figura 1. Diagrama de Bloques del MicroBlaze

Microblaze tiene una serie de instrucciones, en total 87, con las que puede hacer todo el procesamiento, pero si se requiere de procesos más complejos, es necesario el desarrollo de hardware específico utilizando los recursos disponibles de la FPGA. Dado que es un procesador tipo RISC, sus prestaciones se ven mejoradas gracias a la implementación de 3 etapas pipeline, ejecutando una instrucción por ciclo de reloj.

3. uCLINUX

uCLinux es un sistema operativo creado específicamente para microcontroladores que carecen de unidad de manejo de memoria (MMU). Esto implica que no hay protección de memoria; los procesos pueden escribir en cualquier parte y la multitarea es compleja. En la actualidad existe un proyecto "open source" denominado[4] ¨**Petalinux**¨ y liderado por el profesor John Williams en la universidad de Queensland (Australia). Logró el porte de este SO a la arquitectura del procesador Microblaze. La última versión se basa en el kernel 2.6 de Linux, del cual mantiene su estabilidad, capacidad de red superior y su excelente manejo del sistema de ficheros. Fue necesario reescribir el kernel para lograr un tamaño adecuado para ser portado a estos dispositivos, logrando una versión operativa de aproximadamente 900KB, pero su tamaño puede variar.

Existen otras distribuciones de linux para FPGA como VxWorks, BlueCat, MontaVista linux, ThreadX, uC/OS-II, entre otros, pero su único inconveniente es que son propietarios.

4. INSTALACION DE uCLINUX SOBRE MICROBLAZE

Para llevar a cabo una correcta instalación es necesario disponer de herramientas adecuadas para ello, se debe contar con:

1) El entorno de desarrollo de Xilinx ISE 8.2i Foundation y el EDK 8.2i con sus respectivos archivos XBD para mayor soporte sobre las tarjetas de desarrollo. Es necesario versiones superiores a la 8.2i para obtener el soporte de las tarjetas más recientes, preferiblemente sobre windows ya que con la instalación en Linux, se presentan algunos conflictos a la hora de instalar el driver del cable paralelo JTAG.
2) Descargar de la página de Petalinux la distribución completa de uCLinux.
3) Se puede trabajar con cualquier programa para comunicación serial (para el proyecto desarrollado en el Laboratorio Sirius de la Universidad Tecnológica de Pereira, se utilizó el Kermit, el cual es un proyecto open source de la universidad de Columbia en New Cork. este brinda una interfaz confiable para la descarga de archivos binarios y de texto) [5].
4) Opcionalmente se puede tener un servidor TFTP, donde se ubique la imagen del kernel para posteriormente descargarla a la tarjeta pero para el presente proyecto, no se utilizó.

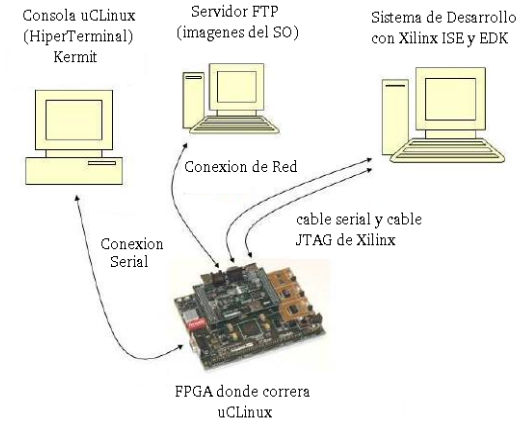

Figura 2. Herramientas de Trabajo

4.1 Creación del Proyecto de Hardware

Dado que la descripción en hardware del Microblaze en VHDL o Verilog no esta disponible para el usuario de FPGAs, el diseño debe hacerse a través de una herramienta específica que proporciona el fabricante. En este caso, dicha herramienta es el EDK (Embedded Development Kit), que permite la configuración del hardware y el desarrollo de los periféricos, y el ISE (Integrated Software Environment) Foundation que proporciona las herramientas necesarias para llevar a cabo las tareas de sintesis, implementacion y programacion de la FPGA. Ambas distribuciones son propias de Xilinx, con el único inconveniente que son propietarias.

4.1.1 Archivo de descripción del hardware

Cuando se hace una configuración de hardware, se crea un archivo .mhs (microprocessor hardware specification). En este archivo se localizan las especificaciones de los puertos del sistema, la memoria o memorias externas a la FPGA, los buses, los recursos del hardware, las instancias y configuraciones de los periféricos del sistema, tamaños de caché y habilitación de las mismas, así como el rango de memoria en el cual está el mapeado de los periféricos. Todo esto mediante una herramienta del EDK de manejo muy sencillo e intuitivo.

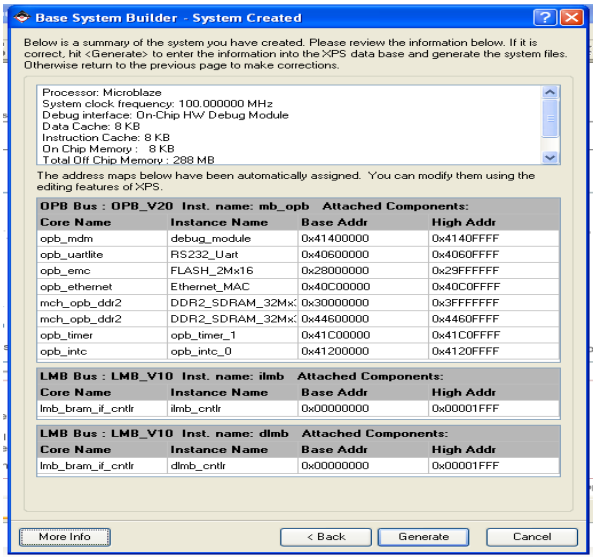

Figura 3. Imagen de la pantalla del editor con la descripción de hardware en la herramienta EDK 8.2i.

4.1.2 Archivo de descripción del software

En este archivo con extensión .mss (microprocessor software specification), está la especificación del software necesario para que el hardware funcione correctamente. Aquí se instancian los drivers necesarios para cada uno de los periféricos que se determinaron en el archivo de descripción del hardware, así como las opciones de compilación.

4.1.3 Configuración final del MicroBlaze

Después de haber generado el proyecto de hardware, se deben hacer varios cambios que se verán reflejados en los archivos .mhs y .mss. Es necesario cambiar el sistema operativo por el de petalinux, y se debe adicionar un proyecto de software nuevo, el cual viene en la distribución de petalinux y es llamado FS-Boot (First Stage Bootloader). Este permitirá cargar un sistema más grande llamado U-Boot, que en principio no cabe en la caché del microprocesador, por lo tanto, debe existir una manera de cargarlo a una memoria mas grande de la tarjeta de desarrollo. De esto se encarga el FS-Boot. Así como otros cambios que sean requeridos.

4.2 Preparación del Kernel

Una vez se ha configurado totalmente el Microblaze y éste se encuentra sobre la FPGA, se continúa con la compilación del kernel. Para esto se deben tener en cuenta las siguientes consideraciones:

1) Tener las fuentes del kernel
2) Conocer cuales dispositivos de hardware se han creado sobre la tarjeta de desarrollo. Esto con el fin de compilar un kernel que se adecue perfectamente a las necesidades que se tengan.

Estas dos consideraciones son vitales para obtener un kernel totalmente funcional e intentar reducir al mínimo la cantidad de errores que se puedan presentar al momento de compilar.

El SO que se compilará contará con la ventaja de utilizar un archivo de texto plano, denominado auto-config.in. Este contiene todas las definiciones de hardware que se hicieron sobre el microprocesador y permite disminuir los errores de compilación; este archivo se encuentra en la carpeta en la cual se hizo el diseño de hardware. Allí se debe buscar para posteriormente ubicarlo en un lugar específico dentro de la estructura de directorios que se crea al instalar las fuentes del kernel. Cabe anotar que la máquina sobre la que se compila el kernel, deberá tener SO basado en Linux, aunque es posible compilar el kernel desde windows utilizando las herramientas adecuadas.

Lo primero que se debe hacer es indicar todos los dispositivos que se usarán, el nombre de la tarjeta para la cual se esta compilando el kernel, la forma en que se quiere que la tarjeta se comunique con el exterior y en que lugar del sistema se desea que se guarden las imágenes del kernel. Todo esto se logra gracias al

comando "**make menuconfig**", el cual lanzará una serie de ventanas que permitirán configurar el kernel.

Posteriormente se procede a ejecutar el comando "**yes "" | make oldconfig dep**" que construye todas las dependencias del proyecto y permite tomar todos los valores por defecto.

Como paso final para la compilación del kernel, se ejecutará el comando **"make all"**; si el proceso es exitoso, las imágenes del kernel quedarán ubicadas en la siguiente carpeta: **$PETALINUX/software/petalinux-dist/images",** compiladas y listas para ser descargadas.

4.3 Descarga de la imagen del kernel.

Cuando todos los pasos anteriormente mencionados están completos, se puede iniciar la descarga de la imagen del kernel. Para esto, lo siguiente que se hace es conectar la tarjeta de desarrollo al puerto serial del computador, y lanzar un programa que permita comunicación por este puerto como el kermit. El programa se configura de la misma manera que el hardware, bits por segundo, bits de datos, paridad, bit de parada, control de flujo, etc.

Cuando la configuración esté completa, se descarga el microblaze desde el EDK. Si el resultado es exitoso, aparecerá en el computador a través del programa para comunicación serial lo siguiente:

==========
FS-BOOT First Stage Bootloader (c) 2006 PetaLogix
==========

FS-BOOT: System initialisation completed.

FS-BOOT: Booting from FLASH. Press 's' for image download.

FS-BOOT: Waiting for SREC image....

Ahora se procede a la descarga de la imagen del U-Boot (Universal Bootloader) el cual permitirá cargar un archivo mucho más grande que es el de la imagen del kernel.

Luego y dependiendo del programa de comunicación utilizado, se debe enviar al puerto serial el archivo **u-boot.srec.** Este se encuentra en la carpeta de las imágenes del kernel, el cual contiene el Universal Bootloader. Posteriormente se observará como resultado lo siguiente:

FS-BOOT: Waiting for SREC image....
FS-BOOT: Image download successful.

FS-BOOT: Warning image location differ from default boot location. Image will not boot automatically after POR.

FS-BOOT: Press 'n' to boot old image.

FS-BOOT: Use new image.

FS-BOOT: Booting image...

SDRAM:

U-Boot Start:0x13fc0000

Malloc Start:0x13f60000

Board Info Start:0x13f5ffd0

Boot Parameters Start:0x13f4ffd0

FLASH: 8 MB

ETHERNET: MAC:<NULL>

*** Warning - bad CRC, using default environment

U-Boot>

Hasta el momento se tiene corriendo el U-Boot, después de esto, se debe enviar la configuración inicial del U-Boot y la imagen del kernel previamente compilado:

loadb $(clobstart)

Desde el programa de conexión serial, se envían los archivos ub.config.img e image.ub, localizados en la carpeta de imágenes; hasta aquí se tiene el sistema cargado en la RAM lo que significa que se perderá al apagar la tarjeta de desarrollo; para correr la imagen del sistema operativo se ejecuta:

autoscr $(fileaddr) – Ejecuta el archivo de configuración
bootm ${kernel_addr} Arranca el SO

Con lo cual se completa el proceso y se tendrá un kernel completamente funcional sobre una FPGA. Si se quiere obtener un sistema en el que no se pierdan todos los datos al reiniciar, simplemente se cargan la imagen del U-Boot y del kernel a una memoria flash en la cual la información no será volátil. Los pasos completos para llevar a cabo lo anterior se encuentran detallados en la pagina de Petalinux.

5. CONCLUSIONES

Se obtuvo un sistema operativo funcionando sobre una FPGA.

En el Laboratorio Sirius, se portó este sistema operativo sobre dos tarjetas, una Spartan 3E Starter Kit revisión D con una FPGA Spartan 3E XC3S500E [18] y una sistema de desarrollo ML501 que posee una FPGA Virtex 5 XC5VLX50 [19] (para las cuales no había soporte, los

resultados fueron enviados al profesor John Williams para que fueran publicados en la pagina oficial).

Se dió a conocer la versatilidad de un sistema operativo como Linux, el gran potencial que existe en las FPGAs y que además de contener al SO, puede ejecutar diversos algoritmos en los recursos que queden disponibles en la FPGA, aprovechando al máximo el paralelismo que ofrecen estos dispositivos.

6. BIBLIOGRAFÍA

[1] Xilinx Inc., "MicroBlaze Processor Reference Guide", UG081 (V 6.3), Ago. 2006. www.xilinx.com/ise/embedded/mb_ref_guide.pdf

[2] Aguayo, E; Gonzáles, I. Et al. "Tutorial Xilinx Microblaze-uClinux". Universidad Autónoma de Madrid, ES. http://arantxa.ii.uam.es/~igonzale/recursos/Tutorial_MicroBlaze_uCLinux_jcra2004.pdf

[3] (2001) MicroBlaze: El procesador soft de las altas prestaciones. Dialnet [Online], 584, pp 74-76. Available: http://dialnet.unirioja.es/servlet/articulo?codigo=140701

[4] Sitio web oficial de Petalinux http://developer.petalogix.com/

[5] Sitio web oficial "Proyecto Kermit". http://www.columbia.edu/kermit/

[6] Sitio web oficial de Xilinx Inc. http://www.xilinx.com/

[7] Xilinx Inc, "Embedded System Tool Guide" www.xilinx.com/ise/embedded/est_guide.pdf

[8] Xilinx Inc, "User Cores Templates Reference Guide". www1.cs.columbia.edu/~sedwards/classes/2005/emsys-summer/user_core_templates_ref_guide.pdf

[9] Xilinx Inc, "Microblaze Hardware Reference Guide". www.xilinx.com/ise/embedded/mb_ref_guide.pdf

[10] Xilinx Inc, "Microblaze Software Reference Guide". www.xilinx.com/ipcenter/processor_central/microblaze/doc/swref.pdf

[11] Sitio web oficial "Linux Devices". http://www.linuxdevices.com/

[12] Sitio Web oficial "Guia de referencia Make". http://www.gnu.org/software/make/manual/make.html

[13] Sitio web oficial de Petalinux "Petalinux Platform From Scratch" http://developer.petalogix.com/wiki/BoardGuides/Custom/Tutorial

[14] Sitio web oficial UBoot "U-Boot-The Universal Boot Loader ". http://www.denx.de/wiki/UBoot

[15] Sitio web oficial Wind River "Linux VxWorks". http://www.windriver.com/vxworks/

[16] Sitio web oficial LinuxWorks "Embedded Linux: BlueCat Linux". http://www.lynuxworks.com/embedded-linux/embedded-linux.php

[17] Sitio web oficial montavista linux "Embedded montavista linux". http://www.mvista.com/

[18] Xilinx UG230 Spartan 3E Starter Kit Board User Guide. http://www.xilinx.com/bvdocs/userguides/ug230.pdf

[19] Xilinx UG226 ML501 Evaluation Platform User Guide. http://www.xilinx.com/bvdocs/userguides/ug226.pdf