6º Jornadas de Desarrollo e Innovación Tecnológica | 2007
Instituto Nacional de Tecnología Industrial
ELECTRÓNICA
Desarrollo e Innovación

Controlador de interrupciones, descripción de hardware

Tropea, S. E.[(i)]

[(i)]INTI-Electrónica e Informática

Introducción

En este trabajo presentamos el desarrollo de un controlador de interrupciones para ser usado con microcontroladores compatibles con el estándar de interconexión WISHBONE[1].

Los microcontroladores son similares a los microprocesadores o *CPUs* utilizados en computadoras. Las principales diferencias radican en su capacidad, mucho más modesta, su consumo, mucho menor, y el hecho de que todos sus componentes periféricos se encuentran incluidos en el mismo circuito integrado, por ejemplo: memoria de programas.

Las interrupciones son el mecanismo más eficiente para que un microcontrolador preste atención a un evento en particular. Este desarrollo se orienta al uso de microcontroladores embebidos en *FPGAs* (*Field Programmable Gate Arrays* = Arreglo de Compuertas Programable en Campo). En un microcontrolador común el número de entradas que permiten interrumpir el procesamiento de datos es limitado y fijado al momento de diseñar el microcontrolador. Esta limitación se debe a que casi todos los periféricos se encuentran integrados en el mismo circuito. Por otro lado en los microprocesadores, como los usados por las computadoras personales, existen mecanismos para extender el número de entradas de interrupción. Usualmente esto se logra utilizando controladores como el expuesto en este trabajo. En nuestro caso la utilización de *FPGAs* nos permite integrar los periféricos que sean necesarios a nuestros propios microcontroladores. Es por esta razón que se hace necesario contar con un controlador de interrupciones.

El objetivo principal de este desarrollo fue el de disponer de un controlador de interrupciones que fuera configurable y extensible, y que pudiera ser utilizado en conjunto con el microcontrolador desarrollado por nuestro laboratorio[2][3]. Dicho controlador de interrupciones debía poder ser sintetizable en *FPGAs* de cualquier fabricante ya que nuestro microcontrolador ya gozaba de esta característica.

Metodología

Las *FPGAs* son circuitos integrados reconfigurables y son los miembros más avanzados de la familia de circuitos lógicos programables. Una *FPGA* está compuesta por lógica combinacional, registros y mecanismos de interconexión para unir estos dos últimos. Todos estos recursos son reconfigurables de manera tal que se puede modificar el funcionamiento para que se ajuste a nuestras necesidades. Estos elementos se complementan con celdas de entrada y salida que permiten conectar nuestro circuito con el exterior.

Las *FPGAs* presentan una alternativa muy interesante cuando es necesario desarrollar un circuito integrado a medida. Los circuitos integrados realizados a medida poseen costos muy elevados y sólo se justifican cuando el volumen de producción supera las decenas de miles. Esto es válido aún para los *ASIC* (*Application Specific Integrated Circuit* = Circuito Integrado de Aplicación Específica) que utilizan técnicas de fabricación de las conocidas como *semi-custom* en las que sólo una parte del proceso es específico del cliente y el resto son estándares. Cuando los volúmenes de producción son pequeños las *FPGAs* aparecen como una excelente alternativa. Las mismas no poseen grandes gastos iniciales o del tipo *NRE* (*Non-Recurring Engineering*) como en el caso de los *ASICs*. Por otra parte si se detectara un error en el diseño, o fuera necesario introducir otro tipo de cambio o mejora, las *FPGAs* pueden reconfigurarse sin ser necesario reemplazar el circuito integrado. Como contrapartida el costo por unidad de las *FPGAs* es superior, su velocidad es inferior y el consumo de energía es mayor cuando se las compara con los *ASICs*.

Dentro de una *FPGA* se puede incluir la funcionalidad de varios circuitos integrados. Esta funcionalidad puede ser desarrollada por el mismo equipo de trabajo o adquirida a través de un tercero. Debido a que estas funcionalidades son como componentes electrónicos, pero sin su parte

física, se los suele llamar componentes virtuales. En la industria se los conoce como bloques de propiedad intelectual o *IP cores*.

El desarrollo de un *IP core* para una *FPGA* es muy similar al de un *ASIC*. En el caso de la *FPGA* el proceso se encuentra simplificado gracias a que nuestro circuito utilizará recursos ya probados y algo sobredimensionados. Este diseño se realiza utilizando lo que se conoce como lenguajes de descripción de hardware. Los mismos tienen cierto parecido con los lenguajes de programación de computadoras pero poseen diferencias conceptuales muy importantes. No se trata de programar un dispositivo sino de describir el comportamiento del mismo. Luego la descripción se convierte en una configuración para la *FPGA* utilizando herramientas de síntesis.

En nuestro desarrollo utilizamos el lenguaje de descripción de hardware *VHDL* (*Very high speed integrated circuit Hardware Description Language* = Lenguaje de Descripción de Hardware para Circuitos Integrados de Muy Alta Velocidad). Se trata de un lenguaje muy utilizado por agencias gubernamentales y en el área aeroespacial. El mismo fue originalmente desarrollado por el Departamento de Defensa Norteamericano y hoy se encuentra estandarizado por la IEEE (normas 1076, 1164 y accesorias).

El estándar de interconexión WISHBONE es el equivalente a un *bus* de interconexión, pero fue específicamente diseñado para ser usado dentro de circuitos integrados. A diferencia de un *bus* normal, como podría ser el PCI o el ISA, WISHBONE se define de una manera más flexible que permite adoptar distintas topologías de interconexión *(ver Fig. 1)*. Un dispositivo que implementa el estándar WISHBONE puede ser conectado con cualquiera de estas topologías. Esto es muy importante en el caso de las *FPGAs* en donde es posible reconfigurar y expandir la funcionalidad de un sistema de una manera mucho más simple que en un circuito físico.

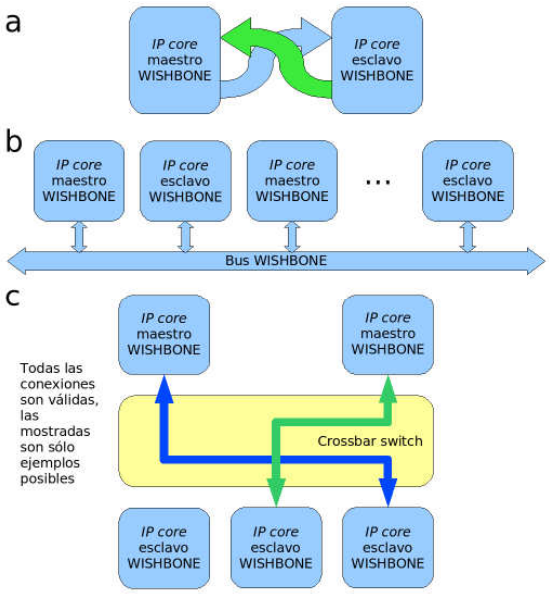

Fig. 1: Ejemplos de topologías posibles; (a) es una conexión punto a punto, (b) es un *bus* compartido y (c) una conexión tipo *crossbar*.

Las características del estándar de interconexión WISHBONE que impulsaron su selección fueron:

—Es un estándar abierto, por lo que cualquier persona o entidad puede acceder al mismo.

—No posee *royalties*, por lo que no es necesario pagar por su uso.

—Es ampliamente usado por el proyecto OpenCores[4], esto permite utilizar componentes desarrollados por dicho proyecto.

—Los recursos consumidos para su implementación son modestos y sólo crecen cuando la complejidad de la interconexión crece. De esta manera, si la interconexión es muy simple, por ejemplo punto a punto, no se desperdician recursos. Por otro lado esto no limita el tipo de interconexión pudiendo implementarse interconexiones de alta complejidad, por ejemplo *crossbar* con varios maestros y varios esclavos.

Las herramientas de desarrollo utilizadas fueron las recomendadas por el proyecto FPGALibre[5][6][7] impulsado por nuestro laboratorio. Para este desarrollo se utilizaron estaciones de trabajo que corren Debian[8] GNU[9]/Linux.

Se procedió al diseño del controlador utilizando el lenguaje *VHDL*, implementando el estándar de interconexión WISHBONE y poniendo especial énfasis en la posibilidad de poder configurar el controlador para adaptarlo a los distintos escenarios de uso. En particular se buscó que se pudieran manejar tantas señales de interrupción como fueran necesarias, y que los recursos

insumidos fueran proporcionales al número de señales necesarias para cada caso en particular, evitando desperdiciar recursos en los casos en que sólo unas pocas señales fueran necesarias.

Resultados

Se obtuvo una descripción del controlador de interrupciones escrita en *VHDL* estándar útil para ser utilizada con cualquier *FPGA*.

Dicho controlador de interrupciones puede ser configurado para soportar entre 1 y 8 señales de interrupción. Cuando se necesitan más de 8 señales basta con conectar otro controlador en paralelo *(ver Fig. 2).* El tamaño del controlador depende de la cantidad de señales de interrupción usadas.

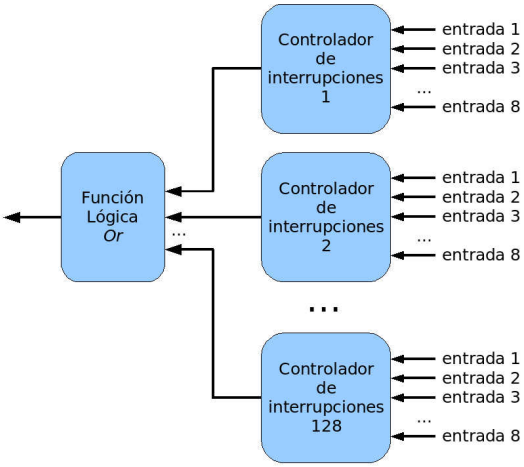

Fig. 2: Conexión de más de un controlador en paralelo.

El controlador posee dos registros, el primero es usado para determinar que entradas de interrupción se encuentran habilitadas y el segundo para determinar que interrupciones se encuentran pendientes, así como para indicar que interrupciones ya fueron atendidas.

El controlador se verificó utilizando *FPGAs Spartan* II[10] y *Spartan* 3[11] de *Xilinx*. El área ocupada por el controlador es de aproximadamente 4,5 *slices* por cada entrada de interrupción. Cada una de ellas insume aproximadamente 3 *flip/flops* y 4 tablas de *look-up*. Para el caso de una pequeña *Spartan* II 100 un controlador de 8 entradas insume aproximadamente el 3 % de la *FPGA*, en el caso de una *Spartan* 3 1500 esto representa aproximadamente el 0,25 %.

Conclusiones

El periférico realizado logró cumplir con los objetivos de ser configurable y extensible. El mismo puede configurarse para utilizar entre 1 y 8 entradas y extenderse para poder utilizar entre 1 y 128 controladores en paralelo, permitiendo así un límite teórico de 1024 fuentes de interrupción.

La utilización de *VHDL* estándar permitió que la descripción de hardware pudiera ser sintetizada para *FPGAs* de otros fabricantes aún cuando se desarrolló utilizando *FPGAs* de *Xilinx*.

La utilización de las herramientas propuestas por el proyecto FPGALibre mostró ser adecuada para este desarrollo.

Referencias

[1] Silicore and OpenCores.Org, "WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores", http://prdownloads.sourceforge.net/fpgalibre/wbspec_b3-2.pdf?download
[2] S. E. Tropea, J. P. D. Borgna, "Microcontrolador compatible con PIC16C84, bus Wishbone y video", *FPGA* Based Systems, ISBN 84-609-8998-4, 2006.
[3] S. E. Tropea, "Microntrolador compatible con *PIC* 16C84, descripción de hardware", 6° Jornadas de Desarrollo e Innovación Tecnológica, Instituto Nacional de Tecnología Industrial, Argentina, 2007.
[4] OpenCores.Org, http://www.opencores.org/
[5] S. E. Tropea, D. J. Brengi, J. P. D. Borgna, "FPGALibre: Herramientas de Software Libre para diseño con *FPGAs*", *FPGA* Based Systems, ISBN 84-609-8998-4, pp 173-180, 2006.
[6] S. E. Tropea, D. J. Brengi, J. P. D. Borgna, S. N. Gwirc, "FPGALibre: Herramientas de Software Libre para diseño con *FPGAs*", 6° Jornadas de Desarrollo e Innovación Tecnológica, Instituto Nacional de Tecnología Industrial, Argentina, 2007.
[7] Proyecto *FPGA* Libre, http://fpgalibre.sourceforge.net/
[8] Debian project, http://www.debian.org/
[9] GNU project, http://www.gnu.org/
[10] *Xilinx Spartan* II *FPGA*, http://www.xilinx.com/products/silicon_solutions/fpgas/spartan_series/spartan2_fpgas/index.htm
[11] *Xilinx Spartan* 3 *FPGA*, http://www.xilinx.com/products/silicon_solutions/fpgas/spartan_series/spartan3_fpgas/

Para mayor información contactarse con:

Ing. Salvador E. Tropea - salvador@inti.gov.ar