Diseño de SoC mínimo para reloj despertador

Malcolm Davis
Computer Engineering
mdavis.cr@ieee.org

David Monestel
Computer Engineering
p.david06.p@gmail.com

Fabian Solano
Computer Engineering
fasm2296@gmail.com

***Resumen*—En este proyecto se demuestra la utilidad de los sistemas a la medida, y además, se pone en practica los conceptos vistos en clase como los modelos de programación "Bare Metal"que serán descritos posteriormente en este documento. Esto, con el desarrollo de un prototipo del SoC(System on Chip) que utilizaría un reloj despertador. El prototipo se modela para su ejecución sobre un procesador NIOS II de Altera.**

Palabras clave: Reloj Despertador, Altera NIOS II, Platform Designer(QSYS), SoC, Servidor Web, Bare Metal, HAL, Sistemas Embebidos, TEC.

I. Introducción

El desarrollo de sistemas embebidos se ha popularizado en los últimos años. El poder tener un dispositivo especializado realizando una función que le facilite la vida a las personas es muy común en estos tiempos. Tan común que existen técnicas para el desarrollo de estos elementos.

Un reloj despertador podría ser un dispositivo sencillo de simular mediante un programa de computación, pero uno de los objetivos de este proyecto es desarrollarlo sobre un dispositivo bare metal; que es un dispositivo que carece de sistema operativo y/o aplicaciones que corren sobre él[1]. De esta forma se puede tomar control de todo el hardware en un bajo nivel, sin tener al sistema operativo como un intermediario. A este nivel se trabajan con interrupciones, de manera que, se realiza una acción de acuerdo con la entrada proveída.

En cuanto al diseño e implementación de un elemento de un sistema, o un sistema. Se puede hacer de diversas formas, desde el más alto nivel, dónde el lenguaje que se utiliza es similar al lenguaje natural, hasta el nivel más bajo el cual es más complejo pero se tiene más control sobre el sistema y se puede realizar cualquier modificación sin inconvenientes.

El sistema debe de ejecutarse de manera continua, realizando cada una de las instrucciones que se encuentran en memoria según el orden establecido. En la mayoría de los casos, este sistema debe de interactuar además con otros dispositivos o un usuario el cual cambia el flujo de instrucciones, a esta alteración del flujo se le conoce como una interrupción[2].

Al utilizar las interrupciones en un sistema bare metal, se logra crear un sistema embebido capaz de llevar el tiempo, detenerse en un momento, y que su flujo de ejecución puede ser modificado mediante interrupciones, en otras palabras un reloj despertador, que además puede mostrar la hora en los displays de siete segmentos que provee la placa de desarrollo.

II. Sistema Desarrollado

Para proponer el prototipo del sistema desarrollado en proyecto se utilizó la metodología modular, se decidió de este modo para así facilitar el diseño individual de cada uno de los componentes o soluciones de subproblemas con menor complejidad que al final se unen para componer el sistema completo [**?**]. La subdivisión de estos problemas se hace en subsistemas individuales que cumplen con los criterios de desarrollo y pueden ser utilizados para crear el sistema completo, más adelante se explica cada uno de ellos. Para conocer más sobre la implementación de cada modulo, puede referirse al documento de diseño que se entregó junto con este proyecto.

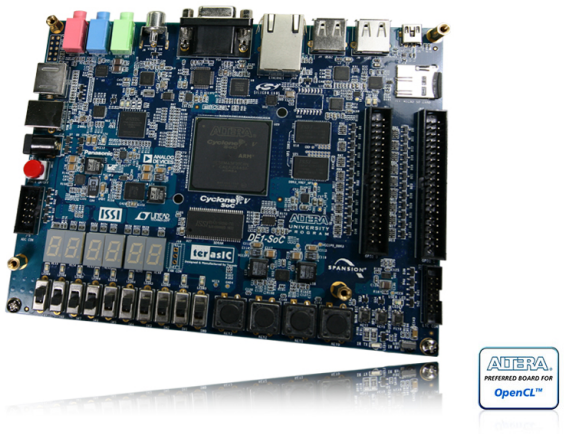

Figura 1. Placa de desarrollo DE1-SoC (recuperado de [3])

II-A. Modelo Físico

Para el modelo físico del prototipo, es decir el reloj despertador con el cual el usuario interactuará, se utilizó la placa de Altera DE1-SoC(ver figura 1). Esta placa provee puertos de entrada y salida programables (GPIO), además cuenta con una serie de núcleos de procesamiento capaces de ser instanciados y utilizados para el desarrollo del proyecto. No se realizó ninguna modificación física a la placa para poder simular el sistema.

II-B. Componentes de Plataform Designer(QSys)

Para diseñar los componentes de hardware que posteriormente iban a ser sintetizados y ejecutados en la tarjeta de desarrollo de altera, se utilizó el programa Plataform Designer. Los módulos utilizados fueron los siguientes:

- **Nios II:** Procesador del sistema que cumple la función de atender las interrupciones que generan los dispositivos de entrada y salida, además de ejecutar el código que se encuentra en la memoria de instrucciones.
- **Memoria:** Para la memoria, se plantea una arquitectura Von Neuman con una sola memoria tanto para instrucciones como datos. Ya que la simplicidad del sistema no requiere una mayor preocupación en los cuellos de botella que se podrían armar en este punto en un sistema más complejo.
- **Jtag UART:** Este modulo se utiliza para poder realizar la comunicación entre el procesador y la tarjeta de desarrollo.
- **Timer:** Este módulo es uno de los más importantes, si no el más importante, del sistema, ya que es el encargado de generar las interrupciones de reloj cada cierto tiempo. Esto genera que se puedan actualizar las variables respectivas para lograr el conteo del tiempo.
- **Botones:** Este módulo es para poder utilizar la entrada de datos del usuario, al igual que otros de entrada y salida, este genera interrupciones al procesador.
- **Interruptores:** Este módulo es para poder utilizar la entrada de datos del usuario, al igual que otros de entrada y salida, este genera interrupciones al procesador.
- **Siete Segmentos:** Este módulo es para poder utilizar la salida de datos del usuario, al igual que otros de entrada y salida, este genera interrupciones al procesador.

II-C. Integración con Quartus

Luego de generar los módulos con Plataform Designer, se puede utilizar Quartus para integrarlos(conectarlos físicamente) y planear la integración del sistema. Por último se sintetiza el sistema y se programa la FPGA.

II-D. Biblioteca/Aplicación

El último paso de la generación del reloj despertador, es programar una aplicación que va a acceder a los registros a las cuales están conectados los dispositivos para terminar de definir el comportamiento y los estados o acciones que cada módulo podrá tomar.

III. Resultados

Esta sección contiene los resultados obtenidos en los diferentes procesos de desarrollo del proyecto.

Se logró crear un SoC mínimo que sirve como unidad principal de un reloj despertador simulado en la FPGA DE1-SoC de altera con las herramientas básicas disponibles al momento de la elaboración del proyecto.

Se logró entender el funcionamiento de las interrupciones que se generan en los diferentes dispositivos de entrada y salida,

así como el temporizador, que el procesador debe atender para así lograr modelar el sistema del reloj despertador.

Se logró crear una interfaz de usuario sencilla con la cual el mismo puede interactuar con el sistema para configurarlo o para conocer el tiempo actual o el de la alarma.

Se realizaron pruebas simples de las diferentes propuestas, para observar los tiempos de reacción y poder estimar el consumo de energía y espacio de cada solución, dónde se evidenció que las primeras dos propuestas no son adecuadas debido al tiempo de respuesta y consumo de energía(ver el documento de diseño para conocer más del tema).

IV. Conclusiones

El uso de herramientas como Qsys y Quartus para el diseño e integración de módulos de hardware simplifica la creación de sistemas simples como el propuesto en este proyecto. Además, es importante hacer una buena elección del diseño del sistema, ya que un mal diseño puede causar resultados mediocres en cuanto a consumo de energía y tiempo de respuesta. Por último, con este proyecto se demostró que la implementación de un sistema de reloj despertador con alarma antiguo se puede realizar con facilidad con las herramientas modernas.

El método mediante interrupciones permite el ahorro de recursos comparado con el método de polling. Acceder directamente a memoria mediante punteros es un mecanismo mas transparente para asegurar el control sobre el comportamiento del sistema. Para utilizar los gpio, se puede realizar de la misma forma que los leds o los botones mediante un módulo PIO para agregar funcionalidad adicional como un buzzer. El control por medio del NIOS II facilita la interacción con el hardware y es una herramienta importante que permite mejorar el tiempo de desarrollo.

Referencias

1 *What is Bare Metal? - Definition from Techopedia"*, Techopedia.com. [Online]. Available: https://www.techopedia.com/definition/2153/bare-metal. [Accessed: 18- Oct- 2018].

2 J. Leiva, *"What is a CPU Interrupt Code (CIC)? - Definition from Techopedia"*, Techopedia.com. [Online]. Available: https://www.techopedia.com/definition/5653/cpu-interrupt-code-cic. [Accessed: 18- Oct- 2018].

3 Altera, DE1-Soc. 2018.

4 *raspberrypi/userland"*, GitHub. [Online]. Available: https://github.com/raspberrypi/userland. [Accessed: 21- Sep- 2018]. 2008.