SALVADOR: UNA HERRAMIENTA PARA EL APRENDIZAJE DE ARQUITECTURA DE ORDENADORES Y TEORÍA DE COMPILADORES

ISRAEL LÓPEZ, IGNACIO DE MIGUEL, JUAN CARLOS AGUADO, FERNANDO GONZÁLEZ, PATRICIA FERNÁNDEZ, RUBÉN M. LORENZO, EVARISTO J. ABRIL Y MIGUEL LÓPEZ

Departamento de Teoría de la Señal y Comunicaciones e Ingeniería Telemática. Escuela Técnica Superior de Ingenieros de Telecomunicación. Universidad de Valladolid. 47011-Valladolid. España.

En este artículo presentamos un prototipo de una aplicación educativa para apoyar el proceso de enseñanza-aprendizaje en las áreas de arquitectura de ordenadores, VHDL y teoría de compiladores. El sistema ofrece la posibilidad de editar programas, compilarlos, ensamblarlos y ejecutarlos en un microprocesador previamente diseñado e implementado físicamente, por lo que es útil para realizar demostraciones. Por otra parte, también permite realizar variaciones sobre cada uno de los módulos que lo integran: compilador, ensamblador y arquitectura del microprocesador, por lo que puede emplearse en la realización de prácticas docentes.

1. Introducción

Actualmente se considera de gran importancia sustituir algunas clases magistrales por otras actividades de aprendizaje en las que el alumno juegue un papel más activo. Debido a esto, cada vez es más habitual encontrar asignaturas teóricas con un alto contenido práctico o centradas en trabajos de laboratorio.

En las asignaturas relacionadas con la arquitectura de ordenadores y/o diseño VHDL suele ser común plantear a los alumnos un ejercicio de diseño de un microprocesador y su implementación en un dispositivo lógico programable como una FPGA [1,2]. El problema al que se enfrentan los alumnos es cómo verificar el funcionamiento del microprocesador implementado. Lo habitual suele ser que el alumno genere manualmente un programa en código máquina que realice operaciones sencillas como sumas o restas, y lo pruebe [2]. Es obvio que si quiere una verificación más exhaustiva realizando operaciones más complejas (programas con múltiples operaciones, saltos, etc.) el ensamblado manual resulta tedioso. Por lo tanto, resulta de gran ayuda disponer de un ensamblador (e incluso de un compilador) que simplifique esta tarea.

Por otro lado, en una asignatura de teoría de compiladores una práctica típica es plantear al alumno la realización de un compilador. Para comprobar si el código en ensamblador generado por su compilador es correcto, el alumno tiene dos opciones: verificarlo manualmente o utilizar un simulador, bien implementado por él o bien ya disponible (realizado por el profesor, por ejemplo). En nuestra opinión, lo ideal es que el alumno pueda verificarlo mediante la ejecución de los programas compilados en un microprocesador real que además esté presente en su puesto de trabajo.

SALVADOR es un sistema modular que trata de dar una solución a estos problemas. Permite proporcionar la parte software a un laboratorio de Arquitectura de Ordenadores (compilador y ensamblador para facilitar la comprobación del diseño del microprocesador) y la parte hardware a un laboratorio de Teoría de Compiladores (microprocesador con un repertorio de instrucciones dado).

2. Arquitectura del sistema

La Figura 1 muestra un esquema del sistema completo. Como puede observarse, consta de dos partes, una hardware y otra software.

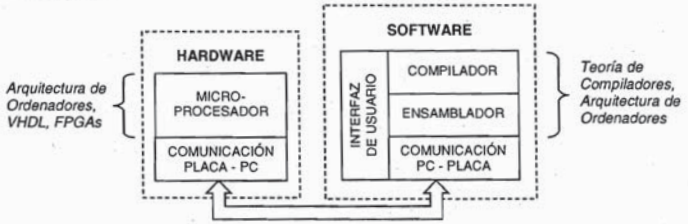

Figura 1: *Arquitectura de SALVADOR. Módulos de los que consta, y asignaturas o temas en los que puede emplearse.*

Parte Hardware: Se trata de una máquina RISC segmentada basada en el microprocesador propuesto en [3] al cual le hemos añadido nuevas características que lo hacen más adecuado para los fines pedagógicos que perseguimos. Es una máquina de carga-almacenamiento de 16 bits y arquitectura Harvard. Cuenta con un repertorio de 16 instrucciones (aritmético-lógicas, de operando inmediato, saltos condicionales y de carga-almacenamiento) y trabaja con aritmética de punto fijo y complemento a 2. Además, dispone de ocho registros internos. El diseño se realizó mediante esquemáticos y código VHDL y se implementó en una FPGA XC4010XL de Xilinx [4] incluida en una placa de demostración XS40 de XESS Corp. [5].

Parte Software: Consta de cuatro módulos:

1) *Interfaz de usuario*: Permite abrir y/o editar programas, compilarlos, ensamblarlos, y cargar el código máquina en la memoria del microprocesador (Figura 2). También permite realizar una ejecución completa o ciclo a ciclo de los programas (Figura 3).
2) *Compilador*: Si bien un proceso de compilación real abarca todas las etapas desde el análisis del programa en alto nivel hasta su conversión en código binario ejecutable, hemos considerado que desde el punto de vista pedagógico es más conveniente separar dicho proceso en una etapa de compilación a alto nivel y una etapa de ensamblado, de modo que el alumno pueda observar las distintas representaciones que adopta el programa según se baja de nivel. Así pues, este módulo traduce un programa escrito en un lenguaje de alto nivel (LIANIS) a código en ensamblador. Para su realización se emplearon las herramientas LEX y YACC [6,7]. LIANIS es un Lenguaje de Instrucciones de Alto NIvel Sencillo, que se creó específicamente para la aplicación.
3) *Ensamblador*: También está desarrollado mediante LEX y YACC. Convierte el código en ensamblador a código máquina entendible por el procesador implementado en la FPGA.
4) *Comunicación PC-placa*: Carga en la memoria asociada al microprocesador el programa a

ejecutar y se encarga del intercambio de datos de entrada/salida entre el microprocesador y el usuario. La comunicación se realiza a través del puerto paralelo del PC.

Figura 2: *Interfaz de usuario de SALVADOR.*

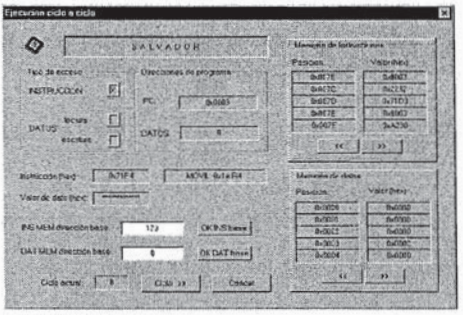

Figura 3: *Ventana para la ejecución ciclo a ciclo de un programa.*

3. Funcionalidades educativas

Desde el punto de vista educativo, SALVADOR puede emplearse de dos formas: para realizar demostraciones, o para que el alumno lleve a cabo variaciones e incluso implemente en su totalidad alguno de los módulos de los que consta el sistema.

- *Demostración de la compilación, ensamblado y ejecución de un programa*: La aplicación permite escribir programas en el lenguaje de alto nivel LIANIS. Teniendo en cuenta que el compilador genera los comentarios adecuados para cada sentencia en código ensamblador, el alumno puede ver con todo detalle el paso de un programa escrito en alto nivel a un programa en código ensamblador, observando así diferencias en tamaño de código, correspondencias entre sentencias de alto nivel e instrucciones (o grupos de instrucciones) en ensamblador, ubicación y manejo de variables, memoria, registros, etc... Tras la fase de ensamblado, el alumno puede ver la equivalencia entre instrucciones en ensamblador y sus códigos de operación en el microprocesador en el que va a ser ejecutado el programa. Por supuesto, podrá ejecutar ese código en el microprocesador y ver los resultados, y así obtener esa idea global que buscamos. Además, tras cada ejecución, la aplicación genera un completo informe de estadísticas sobre las instrucciones ejecutadas, el uso de los registros y los ciclos empleados en la ejecución.
- *Realización de variaciones sobre alguno de los módulos*: El alumno dispone de la información sobre la arquitectura del microprocesador (código VHDL) así como de los ficheros fuente de los programas compilador y ensamblador (ficheros de especificaciones para YACC y LEX, entendibles a alto nivel y fácilmente modificables). De este modo, el alumno puede introducir sus propias ideas modificando cualquiera de estos módulos y comprobar su efecto, o incluso rediseñarlos por completo a partir de especificaciones acordes a las interfaces ofrecidas por los otros módulos del sistema. De esta forma, se puede pedir al alumno que diseñe un microprocesador (o una parte del mismo, como pueda ser la ALU) y que compruebe su funcionamiento utilizando el ensamblador proporcionado con SALVADOR en lugar de ensamblar manualmente. En asignaturas de

Teoría de Compiladores, el alumno puede recibir como especificaciones el lenguaje ensamblador del microprocesador implementado en SALVADOR. A partir de ellas puede programar su propia versión del compilador y comprobar su funcionamiento compilando programas que luego serán ejecutados en un procesador real.

Por otra parte, el alumno puede trabajar en distintas asignaturas con distintas partes del sistema (como puede observarse en la Figura 1), y por tanto ser capaz de relacionar conceptos de arquitectura de ordenadores con los de teoría de compiladores.

4. Conclusiones

SALVADOR es un prototipo para facilitar al alumno el aprendizaje en las áreas de arquitectura de ordenadores, VHDL y/o teoría de compiladores. Es un sistema *completo* que permite editar programas, compilarlos, ensamblarlos y ejecutarlos en una máquina implementada físicamente, por lo que es útil para realizar demostraciones. Además es *modular*, por lo que el alumno puede modificar la arquitectura del microprocesador, así como la programación del compilador y/o del ensamblador y comprobar su funcionamiento tras dichas variaciones.

Actualmente, la conexión entre el PC y la placa de demostración XS40 se realiza de forma manual. Para ello es necesario conectar varios cables desde el puerto paralelo del PC a las patillas adecuadas de la placa. Por esto, una siguiente fase será facilitar la conexión mediante la realización de una placa de circuito impreso en la que se inserte la placa de demostración, y que conste de un conector apropiado. Además, se comenzará a realizar pruebas del sistema con grupos reducidos de alumnos.

Agradecimientos

Los autores agradecen los comentarios de Ioannis Dimitriadis sobre este artículo. Este proyecto está parcialmente financiado por la Consejería de Educación y Cultura de la Junta de Castilla y León dentro del Programa de Ayudas para la Elaboración de Recursos de Apoyo a la Enseñanza Universitaria. La placa de pruebas XS40-010XL v1.2 de XESS Corp. y la licencia del software Xilinx Foundation Series Express F2.1i han sido donadas por Xilinx.

Referencias

[1] D. Wilde. *Introduction to Computer Design.* Brigham Young University. http://www.ee.byu.edu/ee/class/ee325/ee325.html (accedido el 13/06/00).
[2] D. Van den Bout. *The Practical Xilinx Designer Lab Book.* Prentice-Hall (1998).
[3] N.J. Mathai. *A 16-bit Harvard architecture DSP with 5 pipeline stages.* http://www.xess.com/FPGA/projects/9903031280.pdf, y 9903031280.tar, (accedido el 07/10/99).
[4] Xilinx. *The Programmable Logic Data Book 1998.*
[5] Página web de XESS Corporation. http://www.xess.com (accedido el 13/06/00)
[6] A.V. Aho, R. Sethi y J.D. Ullman. *Compiladores. Principios, técnicas y herramientas.* Addison-Wesley Iberoamericana (1986).
[7] J.R. Levine, T. Mason, D. Brown. *UNIX Programming Tools: lex & yacc.* O'Reilly & Associates Inc. (1995).