jsYASP: Un simulador de un procesador educativo en Javascript

Lluís Ribas Xirgo, David Rodríguez Jurado
Departamento de Microelectrónica y Sistemas Electrónicos
Escuela Técnica Superior de Ingeniería (ETSE) – Universitat Autònoma de Barcelona (UAB)
08193 Bellaterra
e-mail: Lluis.Ribas@UAB.es

Resumen

Los procesadores sencillos se emplean en la docencia de los fundamentos de los computadores para mostrar su funcionamiento. En este trabajo se presenta un simulador de una de estas máquinas elementales. Se ha realizado en Javascript para poder ser ejecutado desde cualquier plataforma sin necesidad de instalación. Como resultado adicional, la interfaz gráfica resulta muy simple y amigable y, además, el simulador es de código abierto. Con todo, el uso del simulador facilita un modelo docente complementario al presencial que resulta conveniente para el marco que se establece en el nuevo espacio europeo de educación superior.

1. Introducción

En las asignaturas que tratan de los fundamentos de los computadores se enseña cómo funciona un ordenador a partir de su arquitectura básica. Así pues, un computador es una máquina que toma datos del exterior, los procesa de acuerdo a un programa y obtiene unos resultados. En este modelo, el procesador dispone de una memoria en el que se almacenan datos e instrucciones (la llamada arquitectura de Von Neumann) y una unidad de procesado que las ejecuta con los datos correspondientes.

El procesador es pues, el elemento clave de los ordenadores y el que merece una atención especial en los programas docentes de las asignaturas de los temas correspondientes.

Dado que el empleo de procesadores reales (como los i808x) causa una cierta dispersión del aprendizaje, es recomendable emplear otros de más simples que mantengan las características básicas de la arquitectura de los mismos. Más aun, si se tiene en cuenta que, en la asignatura de Fundamentos de Computadores de la titulación de Informática de la UAB, el tiempo para la parte de rudimentos de arquitectura es de 4,5 créditos.

Con todo, la utilización de un simulador de una máquina elemental didáctica permite consolidar los conocimientos logrados en las clases teóricas, referentes a la estructura y funcionamiento de un computador. Estos simuladores deben de tener un manejo simple para evitar que los estudiantes pasen más tiempo entendiendo su uso que aprendiendo el funcionamiento de los procesadores que simulan. Más aun, atendiendo al hecho de que la implantación del espacio europeo de educación superior supondrá la revisión de los planes de estudio y de la metodología de la enseñanza, resulta conveniente disponer de herramientas que permitan centrar la programación docente en el aprendizaje del alumno.

El objetivo de este trabajo consiste en desarrollar un simulador de una máquina elemental que pueda ejecutarse en cualquier plataforma y cuyo uso sea muy simple.

El procesador es el mismo que ya se utiliza en la asignatura de Fundamentos de Computadores desde el año 2000 [3] y la interfaz gráfica se inspira en el simulador existente.

La independencia de plataforma supone, en la práctica, que los estudiantes puedan desarrollar el trabajo práctico fuera del laboratorio. Esto supone una mayor flexibilidad tanto para ellos como para los profesores. La simplicidad del uso permite que el alumno se centre en el procesador. Dado que éste último debe enfatizar los aspectos clave de la arquitectura de los computadores, este mismo hecho implica un aumento de la eficiencia educativa de la herramienta.

Este artículo se organiza como sigue. En el apartado siguiente se describen brevemente algunos procesadores educativos y, en especial, el empleado en nuestra universidad. A continuación se presenta el simulador y, finalmente, se destacan los puntos más importantes conseguidos con este trabajo en el apartado de conclusiones.

2. Procesadores didácticos

Todos los procesadores de orientación didáctica se basan en la arquitectura de Von Neumann, pues es la empleada en la mayoría de los procesadores actuales.

En esta arquitectura, el procesador ejecuta un programa que se almacena en su memoria principal. El modo de ejecución de este programa es totalmente secuencial: la unidad central de procesamiento (UCP) lee una instrucción de la memoria, la ejecuta y calcula qué instrucción deberá ejecutar a continuación. Este ciclo de ejecución de las instrucciones del programa almacenado en la memoria se repite indefinidamente.

Los distintos procesadores didácticos que existen lo desarrollan de maneras ligeramente diferentes.

Una de las más simples es Blue [1], que tiene un repertorio de sólo 16 instrucciones y únicamente dispone de direccionamiento directo para acceder a los datos de las instrucciones. Este procesador tiene una UCP con un camino de datos con los registros imprescindibles para la comunicación con la memoria (MAR y MBR) y el seguimiento del ciclo de ejecución de instrucciones (PC e IR), así como tres registros para datos (A, Y y Z). La memoria es de 4K-palabras de 16 bits. La comunicación con el exterior sólo puede hacerse mediante mapeo en memoria de los dispositivos correspondientes.

En cambio, la máquina rudimentaria [4] es un procesador que dispone de una memoria de 256 palabras de 16 bits, con un camino de datos que cuenta con un banco de 8 registros. Las instrucciones pueden acceder a sus operandos mediante cuatro tipos de direccionamiento en memoria distintos: inmediato, directo, indexado y por registro. La comunicación con el exterior se supone que se produce mediante mapeo de los dispositivos periféricos en memoria.

Por otra parte, CODE-2 (o computador didáctico elemental [2]) dispone de una UCP más completa, con un camino de datos que incluye un banco de 16 registros y una unidad aritmético-lógica (ALU) con las operaciones básicas. Dado que toda la máquina es de 16 bits, la memoria tiene un tamaño de 64Kb. Dispone de dos instrucciones para acceder a los datos en memoria mediante direccionamiento directo o indexado. Así pues, se trata de un procesador que se inspira en los de arquitectura RISC. Por otra parte, dispone de instrucciones de llamada de funciones. La comunicación con el exterior se realiza mediante puertos de entrada/salida.

El procesador que se emplea en la asignatura de Fundamentos de Computadores de nuestra universidad es una versión algo más compleja que Blue, pero mucho menos que CODE-2. En el apartado siguiente se describe con un poco más de detalle.

2.1. El procesador didáctico YASP

YASP es el acrónimo de *yet another simple processor* y fue desarrollado en el Departamento de Informática de la UAB [3] para ilustrar el funcionamiento de un computador con arquitectura de Von Neumann. Su camino de datos incluye sólo los registros necesarios para efectuar la comunicación con la memoria, el

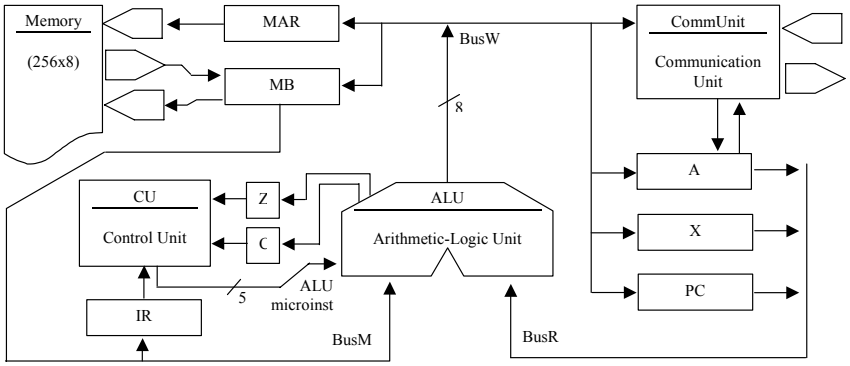

Figura 1. Esquema de la máquina elemental YASP

seguimiento del ciclo de ejecución de instrucciones y la realización de las operaciones indicadas con un registro de propósito general, el acumulador. (Hay otro registro auxiliar que puede emplearse para datos.) La memoria es de 256 bytes, puesto que todos los registros son de 8 bits.

Su repertorio de instrucciones incluye las más comunes y dispone de 4 modos de direccionamiento: inmediato, directo, indexado e indirecto. Este último se incluye sólo para ilustrar su funcionamiento, aunque lo más común en procesadores simples sería una forma indirecta mediante registro. (En YASP no tiene sentido hacerlo así, pues sólo dispone de dos registros para datos.)

La comunicación con los periféricos de entrada y/o salida se realiza mediante puertos y líneas de petición y aceptación. Con ello, YASP se puede ver perfectamente como un procesador completo que puede aceptar datos del exterior y también proporcionarle resultados.

Las líneas de petición y de aceptación permiten ilustrar el funcionamiento de los protocolos asíncronos de comunicación. YASP no acepta interrupciones porque implicaría el uso de llamadas a subrutinas, que tampoco implementa. De todas maneras, las interrupciones pueden emularse mediante las líneas de aceptación.

La no inclusión de instrucciones de llamada/retorno de función se debe a que resulta habitual que los alumnos aun no hayan adquirido el conocimiento de estructuras de datos como la pila, necesarias para su implementación. Además, aumentaría mucho la complejidad de YASP.

3. Simuladores de procesadores docentes

Tal como se ha comentado en la introducción, estos simuladores deben de tener una interfaz de aspecto agradable y fácil de usar, y han de poder ejecutarse sin mayor problema en distintas plataformas.

Todos los procesadores que se han descrito en el apartado anterior disponen de simuladores que se ejecutan en plataformas de PC con alguna de las versiones del sistema operativo Microsoft Windows. Aunque no hay que olvidar que existen diferencias entre versiones de Windows que pueden dificultar la ejecución de algún simulador preparado para otra distinta de la que se emplea para su ejecución. A pesar de ser éstas las plataformas de ejecución más comunes, también hay otras que se emplean cada vez más. En especial las derivadas de la iniciativa del software libre o de código abierto y, claro, del Linux como su sistema operativo.

Así pues, se plantea la necesidad de desarrollar un simulador para un procesador didáctico que pueda ejecutarse en varias plataformas. Con ello se simplifica enormemente la gestión de las versiones del propio simulador y su instalación. Además, se consigue una independencia de la plataforma existente en los laboratorios que permite tanto su actualización como que los alumnos realicen su trabajo en otros lugares y pudiendo organizar su tiempo de forma más adecuada a sus necesidades.

3.1. Simulador en Javascript de YASP

Con el requerimiento anterior, se pensó en emplear Javascript para desarrollar un simulador de la máquina elemental YASP que fuera multiplataforma y de código abierto.

De hecho, con disponer de un navegador de web compatible en cualquier plataforma, se podrá ejecutar el simulador. Así pues, dado que es el navegador el que ejecuta el simulador, éste se convierte en una aplicación multiplataforma de manera directa. Por otra parte, no es necesaria su instalación, si bien es cierto que también puede ejecutarse desde el mismo ordenador para no depender de la conexión a la red.

El código Javascript se incluye en la página que se descarga para ejecutar el simulador, por lo tanto, es de código abierto y puede ser modificado libremente por quién lo desee. Es posible añadir nuevas opciones al simulador, como también realizar cambios en el procesador.

Se descartaron otras tecnologías de desarrollo de programas con tecnología de la web como *applets* en Java o aplicaciones en Flash de Macromedia porque requieren de la instalación de la máquina virtual de Java o del *plug-in* de Flash, respectivamente.

Aun así, Javascript tampoco es totalmente independiente de plataforma, pues los distintos navegadores que existen en el mercado no lo implementan de igual forma. En algunos casos hay instrucciones que solo son válidas para un determinado navegador o métodos que funcionan de diferente forma. Para conseguir un correcto

funcionamiento de jsYASP en la mayoría de navegadores posibles, se optó por utilizar los objetos y métodos que tienen una implementación común. Esto hizo que en alguna ocasión hubiese que renunciar a la eficiencia en detrimento de la compatibilidad.

La interfaz del simulador de la máquina elemental YASP, jsYASP, (véase la figura 2) se ha realizado con HTML, utilizando básicamente tablas y elementos de formulario, para conseguir la mayor homogeneidad posible en la visualización de YASP en los diferentes navegadores. Es muy simple e intuitiva. Todas sus funciones se engloban en una única superficie de trabajo organizada en subventanas: una para el código en ensamblador y otras para el procesador.

jsYASP se ha probado con Internet Explorer 5.0, Mozilla 1.7.3, Mozilla Firebird 1.0, Netscape 6.0 y otras versiones superiores de los navegadores ya citados.

4. Conclusión

Se ha presentado jsYASP, un simulador de un procesador educativo realizado con Javascript que es multiplataforma, de código abierto y con una interfaz amigable: Para ejecutar jsYASP sólo es necesario un navegador compatible. (Está disponible en microelec.uab.es/ribas/edu/fc/yasp.)

Está previsto su uso en el curso 2004/2005 en sustitución del simulador compatible con Windows preexistente.

En un futuro se realizará una versión para que varios simuladores intercambien datos a través de los puertos de los procesadores.

Referencias

[1] Foster, C. *Computer Architecture*. NY: Van Nostrand Reinhold, 1976. [web.frm.utn.edu.ar/tecnicad2/tec_dig2/tools/te2blue.html]

[2] Prieto, A.; Pelayo, F.J.; Gómez-Mula, F.; Ortega, J.; Cañas, A.; Martínez, A.; Fernández, F.J. *Un computador didáctico elemental (CODE-2)*. Actas de las VIII Jornadas de Enseñanza Universitaria de la Informática (JENUI'2002), pp. 117-124, Cáceres, 10-12 Julio 2002. [atc.ugr.es/intro_info_mcgraw/alumno/code2.htm]

[3] Ribas, Ll. *Pràctiques de fonaments de computadors*. Bellaterra: Serv. Pub. UAB, 2000.

[4] –. *Estructura básica de un computador*. Barcelona: UOC, 2002.

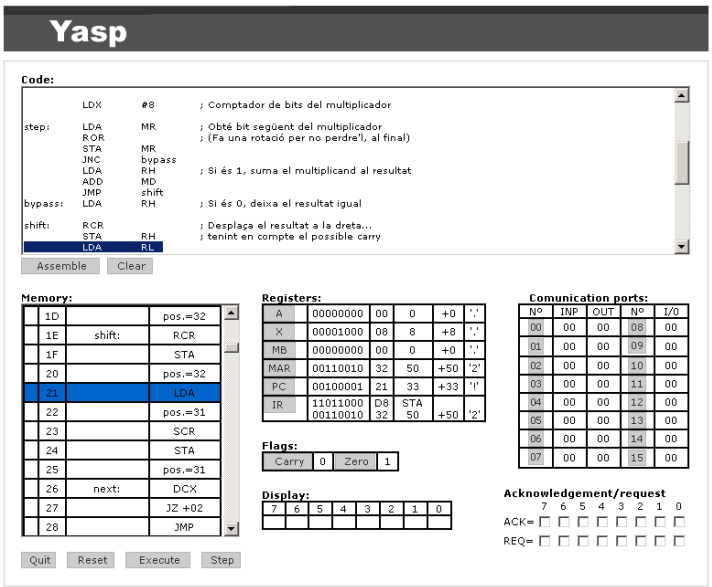

Figura 2. Ventana de ejecución