Facilitando el aprendizaje de la Arquitectura del Juego de Instrucciones

José M. Claver Iborra, María Isabel Castillo Catalán
Depto. de Ingeniería y Ciencia de los Computadores
Universitat Jaume I
12071 Castellón
e-mail: {claver | castillo}@icc.uji.es

Resumen

En este artículo se presenta la metodología y estrategias llevadas a cabo para la enseñanza de la Arquitectura del Juego de Instrucciones en la asignatura de Estructura de Ordenadores de 2º curso de Ingeniería Informática y en particular dentro del desarrollo de sus prácticas. Éstas están dedicadas a la introducción del lenguaje ensamblador de un procesador de propósito general. La metodología utilizada tiene en cuenta, entre otros aspectos, la elección del tipo de procesador y la herramienta de trabajo, la personalización del ritmo de trabajo y la responsabilidad del alumno en la consecución de los objetivos como ejes fundamentales del aprendizaje. Así, se pretende conseguir un acercamiento mucho menos traumático del alumno a una de las asignaturas que mayor importancia tiene en la formación de base de los futuros ingenieros informáticos. Los resultados obtenidos hasta el momento nos indican que la acogida por parte del alumno es positiva, quedando ésta reflejada en la facilidad e interés con que resuelven los problemas propuestos en cada práctica. Por ello se produce una mejora en el aprendizaje de la asignatura, aspecto que queda reflejado en la evaluación de su rendimiento.

1. Introducción

La docencia de Arquitectura de Computadores en las titulaciones de informática aparece generalmente en el primer curso y se prolonga a lo largo de toda la carrera, variando en intensidad y profundidad según se trate de Ingeniería Informática (I.I.), Ingeniería Técnica en Informática de Sistemas (I.T.I.S.) o Ingeniería Técnica en Informática de Gestión (I.T.I.G.). Los contenidos de esta materia son en la actualidad, y previsiblemente lo serán en los próximos 10 a 20 años, un aspecto fundamental en la formación de los futuros informáticos [2], ya que el desarrollo de aplicaciones y sus prestaciones tienen un importante impacto en la arquitectura, estructura y organización de los computadores [10, 11, 14]. Por otra parte, no se prevé en este periodo una modificación importante en las bases de la construcción y funcionamiento de los computadores. Actualmente estamos utilizando algunas ideas que aparecieron hace más de 30 años, aunque habrá que estar a la expectativa de iniciativas que proponen modelos de computación muy diferentes a los que conocemos hoy en día, como la computación cuántica o molecular [4].

Si bien hay un acuerdo generalizado en cuanto a los contenidos que deben impartirse en los primeros cursos de arquitectura de computadores [1,2], existen diversas alternativas referentes al tipo de procesador y herramientas utilizadas para la descripción y estudio de su funcionamiento [6]. También puede variar la profundidad del estudio de los diferentes elementos del computador [7], dada la alta sofisticación de los computadores actuales y la variedad de equipos con los que nos podemos encontrar. De especial interés es el estudio del procesador y la arquitectura de su juego de instrucciones, y de manera muy especial la programación en ensamblador, el uso de la memoria y la gestión de la entrada/salida. Por ello, los cursos introductorios de Arquitectura de Computadores hacen hincapié en estos temas.

El complemento de esta materia lo constituiría el estudio de los aspectos más tecnológicos del diseño de computadores, pero estos caen fuera de los contenidos tratados en el presente artículo.

Como veremos a lo largo de este trabajo, la elección del procesador o las herramientas utilizadas para la enseñanza de esta materia son importantes, pero también influyen de forma decisiva el ritmo y la responsabilidad en el aprendizaje de los conceptos y técnicas que queremos trasladar. Esto determinará que los alumnos puedan, en su gran mayoría, alcanzar las habilidades que se pretenden al finalizar el curso, o que muchos de ellos se descuelguen del fino hilo conductor que les liga a la asignatura, y que a partir de un momento dado les parezca kafkiana y la abandonen.

El resto de este trabajo se organiza como sigue: En el apartado 2 describimos algunos de los aspectos más destacados relacionados con la docencia en las asignaturas introductorias de Arquitectura de Computadores. En el apartado 3 hacemos un pequeño repaso a la docencia de estas asignaturas en otras universidades y en el apartado 4 desarrollamos los aspectos más destacados de nuestra propuesta educativa. En el último apartado, presentamos las conclusiones derivadas de la experiencia de nuestra propuesta y planteamos las mejoras que abordaremos en próximos cursos.

2. Docencia introductoria a la arquitectura de computadores

Los contenidos de las asignaturas introductorias de arquitectura de computadores incluyen, entre otros, el estudio de los siguientes ítems:

- Procesador: Estructura y organización, ruta de datos, juego de instrucciones y lenguaje máquina, lenguaje ensamblador y su relación con lenguajes de alto nivel.
- Memoria: Organización, almacenamiento de instrucciones y datos y cache.
- Entrada/Salida: Gestión mediante consulta de estado e interrupciones.
- Rendimiento: Análisis de prestaciones y comparativas entre computadores.

Estos contenidos deben transmitirse al alumno, tanto a través de la docencia en el aula como de su trabajo en el laboratorio, de tal forma que adquiera los conocimientos y habilidades establecidos en los objetivos del curso. En el aula deben fijarse los aspectos más conceptuales de la materia, mientras que en el laboratorio estos deben reforzarse y ampliarse mediante su uso y la adquisición de habilidades de análisis y síntesis. Así, suele ser habitual en estas asignaturas potenciar el estudio en el laboratorio del lenguaje ensamblador (exoarquitectura) mediante el uso de una máquina real o un simulador que esté estrechamente relacionado con el modelo presentado en el aula (más relacionado con la endoarquitectura y la microarquitectura [8]). En estas prácticas deben manejarse todos los elementos conceptuales vistos en teoría de forma aplicada, lo que supone seguir en lo posible el ritmo y orden de los conceptos introducidos en el aula.

Como esta materia no puede verse de forma abstracta, existe un momento en el diseño de su proyecto docente en el que hay que decidir el procesador ejemplo y la herramienta o entorno de trabajo a utilizar en cada asignatura de entre varias alternativas.

2.1. Procesador

En el dilema entre procesador real (comercial) o procesador ficticio (hipotético) parece claro que el interés de un procesador real está en la posibilidad de utilizarlo como parte integrante de un dispositivo real como es un computador comercial o un sistema de entrenamiento especialmente diseñado para las prácticas, algo imposible de ver con procesadores ficticios. Pero los procesadores reales habitualmente introducen particularidades, que pueden ser poco generalizables. Esta característica, en el caso de los primeros cursos, puede introducir complejidades añadidas en el proceso de aprendizaje, así como introducir una visión poco acorde con la generalidad de los procesadores, por ello la importancia de una adecuada elección.

Como contrapartida los procesadores ficticios pueden adaptarse a las necesidades educativas en cada momento de la carrera, en función de la preparación del alumno para asimilar, y aplicar los conceptos y técnicas que estos integran. Estos procesadores suelen diseñarse eligiendo abstracciones de las características más generalizadas en los procesadores reales.

2.2. Herramienta de trabajo

La elección de la herramienta a utilizar varía entre

el uso directo de un computador comercial o un sistema de desarrollo, y el uso de un simulador. Sólo es posible utilizar directamente un computador si hemos optado por un procesador real, y entonces el ensamblador y el software utilizado deben adaptarse a las características de dicho procesador, y a las posibilidades y restricciones de la máquina en el que se encuentre integrado. En estos casos muchos de los análisis del comportamiento de procesador tienen que ser indirectos y están limitados por la estructura particular de la máquina (memoria, cache, bus, etc.).

Si se elige un simulador, éste puede estar diseñado para un procesador real o para un procesador ficticio (de origen o como abstracción de un procesador real, con lo que se transforma en un procesador ficticio). El uso de simuladores de procesadores ficticios permite una adecuación mayor a las necesidades de un determinado curso e ir añadiendo, en cursos posteriores, ampliaciones o modificaciones de las abstracciones iniciales. De esta manera se van introduciendo conceptos cada vez más avanzados sin necesidad de que el alumno tenga que aprender una nueva herramienta ni un nuevo lenguaje ensamblador.

En cualquiera de los dos casos es posible utilizar recursos del computador (directa o indirectamente), aunque estos siempre suelen ser mucho más limitados en el caso de los simuladores.

3. Docencia en otras universidades

En los últimos años, la experiencia docente en las asignaturas de introducción a la arquitectura de computadores se ha caracterizado por:

- Elección de un primer procesador simple o sencillo ficticio en vez del procesador real de 8 bits (8085, Z80, 6800 ó 6500) de hace unos años, para conseguir reducir el escalón entre los conocimientos del alumno y los conceptos introductorios [3, 13] en los primeros cursos de introducción a los computadores.
- Continuación en el segundo curso con un procesador real, normalmente un CISC de los años 80, de 16 bits (Intel i8086 o Motorota MC68000) y en algunos caso un procesador RISC de 32 bits (como el MIPS R2000 simplificado, el ARM, Motorota MC88010 u otros), para obtener una mayor aproximación a la complejidad de los procesadores reales [6].
- Uso de simuladores o combinación de estos con sistemas de desarrollo o computadores comerciales, que permiten ver con comodidad la visualización del estado y comportamiento del procesador, y la posibilidad de realizar aplicaciones sobre hardware real, respectivamente.

En las universidades españolas suelen darse las características anteriores, aunque con algunas variaciones. Hemos elegido para mostrar este hecho las experiencias llevadas a cabo por cuatro de éstas en las asignaturas de los dos primeros cursos de Ingeniería Informática (Universidad Complutense de Madrid (UCM), Universidad Politécnica de Cataluña (UPC), Universidad Politécnica de Madrid (UPM) y Universidad Politécnica de Valencia (UPV)). La elección se ha basado en la antigüedad de sus estudios, la experiencia de su profesorado y el número de alumnos, sin menoscabo de aquellas aquí omitidas.

En el primer curso se aborda, en todas ellas, el estudio de un procesador simple: en la UPM se utiliza el picocomputador, y en la UPC y la UCM la máquina sencilla [3]. Estos procesadores son completamente ficticios, mientras que en la UPV se estudia el procesador R2000 en su abstracción simplificada no segmentada [14]. En todos los casos se hace uso de un simulador como herramienta de trabajo.

Ya en el segundo curso se utilizan procesadores reales de tipo CISC en la UCM (MC68000, sobre un sistema de desarrollo) y en la UPC (i8086, sobre un PC), combinando, en ambas, el uso del ensamblador y del lenguaje C. Por el contrario, se utilizan simuladores de procesadores RISC en la UPM (MC88010) y la UPV (abstracción del MIPS R2000).

4. Nuestra propuesta

La asignatura de Estructura de Ordenadores (EO) de 2º curso de la Ingeniería Informática es la segunda de las asignaturas dedicadas a la Arquitectura de Computadores, después de la

asignatura Introducción a la Informática, de primer curso, donde se introducen conceptos muy elementales acerca del funcionamiento de los computadores. Por tanto, la enseñanza del juego de instrucciones del procesador y, por ende, el aprendizaje del lenguaje ensamblador forma parte importante de los contenidos de EO.

Hasta el curso pasado el microprocesador elegido (desde hace 10 años) para las prácticas era el MC68000, uno de los exponentes más elegantes de la arquitectura CISC. Por muchas razones éste es uno de los procesadores más utilizados en la enseñanza de la Arquitectura de Computadores y sobre el que se han escrito más libros, aunque está muy alejado de los procesadores seminales de las actuales arquitecturas superescalares. Las prácticas se realizaban sobre un sistema de desarrollo que sólo podía ser utilizado en el laboratorio, con un entorno de trabajo bastante tedioso, y sin la posibilidad de que el alumno pudiera utilizarlo en su casa.

Cada sesión de prácticas se organizaba de la siguiente forma: comenzaba con una larga explicación por parte del profesor, donde se introducían muchos conceptos nuevos y se fijaban los objetivos de la práctica. Estos objetivos eran claros, pero muy ambiciosos, y requerían que el alumno, por un lado, prestara mucha atención durante la explicación realizada por el profesor, y por otro, antes de empezar con el desarrollo de la práctica, realizara un concienzudo estudio del enunciado de ésta, donde se incluía una breve descripción de los conceptos anteriormente explicados. Ambas tareas ocupaban una parte importante de la sesión, dando lugar a que el alumno dispusiese de muy poco tiempo para desarrollar los problemas propuestos en la práctica. Además, y aunque las prácticas estaban diseñadas en orden creciente de complejidad, desde la primera práctica el alumno tenía que enfrentarse al análisis y diseño de un programa en ensamblador más o menos complejo, con declaración de datos en memoria, instrucciones de distinto tipo y diversos modos de direccionamiento. Esta última circunstancia se da habitualmente en la enseñanza del lenguaje ensamblador en las universidades antes analizadas.

Todo lo anterior nos llevó a plantearnos una alternativa que hiciera más sencillo y cómodo para el alumno conseguir los objetivos que pretendíamos para estas prácticas. Esta debía basarse en un conjunto de condiciones iniciales que facilitaran la obtención de nuestros objetivos y que resumimos a continuación:

1. Sencillez de la arquitectura del procesador.
2. Abstracción de un procesador real más avanzado.
3. Sencillez de uso del simulador.
4. Posibilidad de seguir las prácticas en casa.
5. Posibilidad de autoaprendizaje.
6. No obligatoriedad de asistencia con horario fijo.
7. Ritmo de aprendizaje personalizado.

Para alcanzar la primera condición se ha elegido el procesador MIPS R2000, y en particular una abstracción no segmentada de éste que evita las dificultades asociadas a su estructura y los problemas de su programación real [14]. Este procesador es de tipo RISC, por lo tanto con un conjunto de instrucciones sencillo y modos de direccionamiento reducidos. Sin embargo, esta elección permitirá en cursos más avanzados el uso del mismo procesador, o similar, sin las actuales simplificaciones [9,11], con lo que alcanzamos la segunda condición.

Las condiciones 3 y 4 se consiguen mediante el uso del simulador SPIM, en particular su versión gráfica XSPIM, desarrollado por James R. Larus de la Universidad de Wisconsin, que funciona tanto en plataformas Linux como DOS/Windows [12]. Se trata de un simulador integrado, donde toda la información está visible en todo momento, y muy fácil de utilizar. En las últimas versiones (a partir de la 6.3) tiene la posibilidad de mostrar las dificultades de programación del MPIS R2000 segmentado con la introducción de cargas y saltos retardados (delayed load y delayed branch). Además, gracias a que se trata de un simulador de libre distribución y multiplataforma, el alumno puede utilizarlo en casa.

Las tres últimas condiciones se consiguen gracias a la planificación y desarrollo del material de prácticas. El contenido de las sesiones de laboratorio debe diseñarse para que introduzcan en cada una de ellas nuevos conceptos, cada vez más complejos a medida que avanza el curso, y en las que deben utilizarse los conceptos vistos en las sesiones anteriores. La Figura 1 muestra de dentro hacia fuera el contenido de las sesiones prácticas programadas en la asignatura.

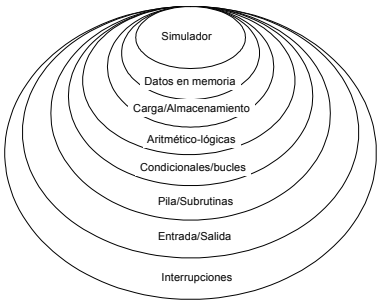

Figura 1. Contenidos de las sucesivas sesiones prácticas.

Dentro de cada sesión se debe tener especial cuidado con el ritmo del aprendizaje del alumno y con no violar la máxima de que primero hay que analizar antes de pasar a la síntesis. Por ello se comienza siempre con una breve introducción y unos programas ejemplo en los que el alumno debe analizar su comportamiento. Con posterioridad se le proponen diversos cambios sobre el programa ejemplo que vuelven a analizar y en los que se pretende aumentar su participación. Finalmente, y antes de seguir adelante, se le propone un pequeño problema de síntesis para comprobar su comprensión de la técnica o concepto introducido. Estos pasos se resumen el la Figura 2.

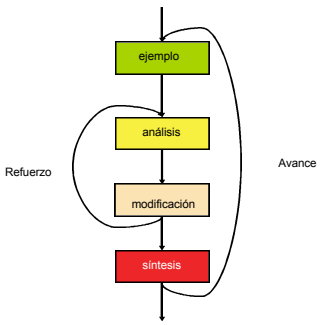

Figura 2. Flujo de aprendizaje de las prácticas.

Así, las prácticas se estructuran en forma de sucesivas cuestiones que demandan del alumno acciones de análisis de programas ejemplo, modificaciones de estos y diseños completos a partir de lo aprendido hasta ese momento (como puede verse en la Figura 3). Cada práctica concluye con problemas más complejos que adquieren la estructura de pequeños proyectos. Todas estas prácticas se encuentran recogidas en un libro de prácticas publicado a tal efecto que puede conseguirse también electrónicamente a través de la página Web de la asignatura [5].

Cuando el alumno tiene dudas acerca de algún concepto o técnica, sólo tiene que revisar las sesiones anteriores y consultar o repetir alguna parte de las prácticas ya realizadas. De esta forma, la necesidad de un profesor todo el tiempo al lado del alumno se reduce, el alumno va, paso a paso, resolviendo las cuestiones que se le plantean y aprendiendo de forma activa, por lo que la asistencia a las sesiones de prácticas no es obligatoria. Como es lógico, tampoco se exige al alumno un ritmo de aprendizaje determinado, y éste puede llevar su propio *tempo*. Lo que se le recuerda, en el laboratorio, en las tutorías y en la página Web de la asignatura, es el ritmo general. Así, el alumno al que le pueda costar más alguna práctica sabe que debe trabajar fuera del horario de las sesiones programadas, en casa o en el horario de acceso libre al laboratorio, para llegar al final del curso con todas las prácticas realizadas. De ésta forma, podrá afrontar con mayores garantías la superación de la evaluación de esta parte de la asignatura.

Ejemplo	Crea un fichero con el siguiente código: … Descripción: …
Análisis	**Cuestión 1.** ¿Qué hace? ¿Cuál es el valor de...? **Cuestión 2.** ¿Indica qué instrucciones hacen...? **Cuestión 3**. ¿Si el dato... tiene el valor 5 qué ocurre....?
Modificación	**Cuestión 4.** Modifica el código para que....
Síntesis	**Cuestión 5.** Implementa un programa que....

Figura 3. Estructura general de los apartados de las prácticas.

El manejo del simulador no es un objetivo finalista de las prácticas, y por ello el alumno no deberá demostrar la destreza en su uso en la evaluación. Sí que se le exige, por el contrario,

que sepa analizar, modificar y diseñar pequeños programas utilizando los conceptos y técnicas vistos en las clases prácticas.

En la Figura 4 se puede constatar el incremento del número de aprobados y las notas obtenidas en la evaluación del ensamblador en el curso 00/01, en el que hemos introducido los cambios propuestos, respecto de los cursos anteriores (98/99 y 99/00). Este hecho queda claramente reflejado en las líneas de tendencia de los dos últimos años.

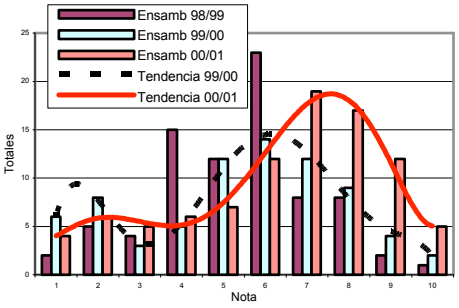

Figura 4. Resultados en la evaluación del ensamblador en los últimos 3 cursos.

También se ha incrementado el número de alumnos presentados a los exámenes en un 20%, por lo que se ha reducido el número de abandonos de la asignatura (y que en mayor número se producen en el primer año en que se cursa ésta). A pesar de ello aún existen notas muy bajas (entre 0 y 3) que suponen el 16 % del total de alumnos presentados, aunque los porcentajes son menores a los del curso 99/00 en el que éstos alcanzaban el 23 %.

5. Conclusiones y trabajos futuros

Las impresiones obtenidas en el laboratorio, a través de la actitud activa de los alumnos y de las apreciaciones personales de aquellos que han repetido la asignatura con el nuevo método, indican que las prácticas se siguen mucho mejor y por tanto el aprovechamiento por parte del alumno es mayor. Este extremo ha sido corroborado por los resultados de la evaluación del ensamblador en el último curso. Además, en las nuevas prácticas se trata la entrada/salida de datos y el manejo de las interrupciones de forma práctica, cosa que no se llegaba a ver en las prácticas de años anteriores.

La metodología descrita en este artículo, parte de la cual queda plasmada en el libro de prácticas elaborado [5], se ha adoptado como guía para el desarrollo de las prácticas de las asignaturas introductorias a la programación en ensamblador de todas las titulaciones de Informática en nuestra Universidad. En el caso de la I.T.I.G. se han seleccionado solo los contenidos que consideramos más importantes para la formación del alumno.

Por último, comentar que otras universidades nos han pedido permiso para adoptar este material como guía para sus prácticas en cursos similares. Para el próximo curso tenemos previsto ampliar este material con la inclusión de las operaciones aritméticas en coma flotante y el uso de las interrupciones para la gestión de tareas.

Referencias

[1] ACM/IEEE-CS Joint Curriculum Task Force. *Computing Curricula 1991.* http://computer.or/education/cc1991.

[2] ACM/IEEE-CS Joint Curriculum Task Force. *Computing Curricula 2001.* http://computer.or/education/cc2001.

[3] E. Ayguadé, J.J. Navarro, M. Valero García. *La màquina senzilla. Introducció a l'estructura bàsica d'un computador*. Col•lecció Aula, CPET, 1992.

[4] A. Barenco, A. Ekert, A. Sanpera, C. Machiavello. *Un saut d'échelle pour les calculateurs*. La Recherche, Nov 1996, http://www.qubit.org/intros/comp/comp.html.

[5] M. Castillo, J. Claver. P*rácticas guiadas para el ensamblador del R2000.* Ediciones Universitat Jaume I. 2001, http://yan.act.uji.es/E38.

[6] A. Clements. *Selecting a Processor for Teaching Computer Architecture*. Microprocessors and Microsystems, mayo 1999.

[7] A. Clements. *The Undergraduate Curriculum in Computer Architecture*. IEEE Micro, pp. 13-22, mayo - junio 2000.

[8] S. Dasgupta. *Computer Architecture - A Modern Synthesis*. John Wiley & Sons, 1989.

[9] E. Farquhar, P.J. Bunce. *The MIPS Programmer's Handbook.*. Morgan Kaufmann, 1993.

[10] J.P. Hayes. *Computer Architecture and Organization*. McGraw-Hill, 1998.
[11] J.L. Hennessy, D.A. Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, 2ª edición,1996.
[12] J.R. Larus. *MIPS. A R2000/3000 Simulator*. University of Wisconsin, http://www.cs.wisc.edu/~larus/spim.html.
[13] E. Pastor, F. Sanchez. *La máquina rudimentaria: Un procesador Pedagógico*. III Jornadas de Enseñanza Universitaria sobre Informática JEUNI'97, pp. 395-402, junio 1997.
[14] D.A. Patterson, J.L. Hennessy. *Computer Organization and Design. The Hardware/Software Interface*. Morgan Kaufmann, 2ª edición, 1997.