

Características deseables en un procesador pedagógico para la enseñanza básica de Arquitectura de Computadores

Fermín Sánchez

Dept. de Arquitectura de Computadores
Universitat Politècnica de Catalunya
Barcelona e-mail: fermin@ac.upc.es

Resumen

En este artículo¹ se analiza cómo debería ser la arquitectura a nivel de lenguaje máquina de un procesador pedagógico que se estudie en una asignatura de primeros cursos de una Ingeniería Informática, ya sea técnica o superior. Se estudian también las características básicas de un lenguaje ensamblador y de un simulador para dicho procesador. Finalmente, se presentan tres ejemplos de procesador pedagógico que se usan o se han usado en universidades españolas y se discuten sus ventajas e inconvenientes.

1. Introducción

¿Qué conocimientos debería tener un ingeniero en informática sobre estructura y arquitectura de computadores² (AC)? En realidad, la respuesta a esta pregunta debería ser diferente en función de la titulación del ingeniero. Parece coherente pensar que los conocimientos sobre AC que debe poseer un Ingeniero Técnico en Informática de Gestión (ITIG) deben ser inferiores a los que ha de tener un Ingeniero Técnico en Informática de Sistemas (ITIS), y a su vez los de éste deben ser inferiores a los de un Ingeniero (superior) en Informática (II). Las directrices generales propias de los planes de estudios del BOE 278 del 20 de Noviembre de 1990 corroboran esta idea, e indican que el número de créditos troncales en AC ha de ser de 9 para un ITIG, 15 para un ITIS y 24

para un II (15 en el primer ciclo y 9 en el segundo). No obstante, muchas Universidades españolas han optado por incluir un mayor número de créditos obligatorios en esta materia. Citaremos a continuación tres ejemplos de ello³.

- La Universidad Politécnica de Madrid imparte en la Escuela Universitaria de Informática 21 créditos obligatorios para la ITIG y la ITIS, y en la Facultad de Informática 31.5 créditos obligatorios para la II, 22.5 de ellos en el primer ciclo.
- La Universidad Politécnica de Valencia imparte en la Escuela Universitaria de Informática 21 créditos obligatorios en la ITIG y 28.5 en la ITIS, y en la Facultad de Informática 36 créditos obligatorios para la II, 21 de ellos en el primer ciclo.
- La Universitat Politècnica de Catalunya imparte en la Facultat d'Informàtica de Barcelona 15 créditos obligatorios para la ITIG, 19.5 para la ITIS y 28.5 para la II, de los cuales 19.5 corresponden al primer ciclo.

Como el objetivo final es que en cada titulación el alumno adquiera los conocimientos apropiados sobre AC, el modo más sencillo de conseguirlo es disponiendo de asignaturas independientes para cada una de las titulaciones. Sin embargo, el solapamiento de temarios entre ellas, especialmente en las asignaturas más básicas, sería notable. Las variaciones se encontrarían más en la profundidad con que deberían tratarse algunos temas que en las descripciones mismas de los temarios. Estas diferencias pueden describirse mediante los objetivos [1] de la asignatura, que ayudan al

¹ Este trabajo ha sido financiado por el Ministerio de Educación y Ciencia mediante el contrato TIC 98-0410

² Dentro de esta terminología no se consideran, en este artículo, los conocimientos relativos a tecnología de computadores

³ La información proporcionada en este artículo ha sido extraída de las *websites* de las universidades correspondientes

docente a clarificar el nivel y profundidad con que debe explicar cada tema y los conocimientos que debe adquirir el alumno. Sin embargo, por razones técnicas (y presupuestarias) algunas universidades han optado por impartir de forma común algunas asignaturas básicas de las dos ingenierías técnicas, e incluso en algún caso éstas son comunes con asignaturas básicas de la II.

En este artículo analizaremos las características básicas que debe tener un procesador pedagógico que se explique en la/s primera/s asignatura/s sobre AC de cualquiera de las titulaciones de informática: ITIG, ITIS e II.

En la sección 2 se describen los conocimientos básicos de AC que deberían impartirse en las asignaturas de primeros cursos. La sección 3 presenta las características básicas de un procesador pedagógico y en la sección 4 se resumen las características de tres procesadores "sencillos". La sección 5 expone los aspectos de lenguaje ensamblador que el alumno debe conocer en este nivel de estudio y en la sección 6 se presentan las características aconsejables en un simulador de una máquina como la aquí descrita. Las conclusiones se presentan en la sección 7.

2. Conocimientos básicos de AC en asignaturas de primeros cursos

El objetivo general de la primera asignatura sobre AC es que el alumno entienda la arquitectura von Neumann [2] y sepa cuál es la misión de cada uno de sus componentes: memoria, unidad de entrada/salida, procesador y red de interconexión.

La memoria debe estudiarse, en este nivel, como si se tratase de un banco de registros de gran tamaño. Los aspectos relativos al estudio de la jerarquía de memoria deben dejarse para cursos posteriores, aunque es aconsejable mencionarlos para que el alumno vaya concienciándose de que está estudiando una versión simplista de lo que es realmente un computador.

La unidad de entrada/salida tampoco debe ser objeto de estudio profundo en este curso, ya que:

- es necesario poseer conocimientos básicos (previos) de programación para programar los dispositivos de entrada/salida. Estos conocimientos suelen adquirirse de forma simultánea con el estudio de esta asignatura.

- entender su funcionamiento (a un nivel profundo) implica comprender el concepto de concurrencia

Dado que el curso debe estar centrado en que el alumno entienda los principios básicos de funcionamiento de un procesador, lo más eficaz es ejemplificar el tema a través de un procesador pedagógico diseñado específicamente para facilitar el aprendizaje de estos conceptos.

Este procesador debe tener una Unidad de Proceso (UP) y una Unidad de Control (UC) bien diferenciadas. Es conveniente que la UC sea sencilla y que cada uno de los estados pueda identificarse claramente con cada una de las fases de ejecución de una instrucción. La UP debe estar diseñada con la circuitería mínima necesaria que permita al alumno entender los conceptos de AC que se le pretenden inculcar. Para entender el funcionamiento del procesador es preciso que el alumno tenga algunos conocimientos previos de sistemas digitales, los mínimos que le permitan comprender cómo funciona el procesador. Estos conocimientos pueden adquirirse en la misma asignatura en muy pocas horas o, en los casos de la II y la ITIS, pueden impartirse de forma más profunda en una asignatura (obligatoria) distinta.

El alumno debe conocer también los distintos tipos de buses y la información que circula por cada uno de ellos. Los aspectos relativos a sincronización han de estudiarse en cursos posteriores por motivos similares a los descritos en la unidad de entrada/salida.

La literatura está llena de máquinas ideadas para ayudar al alumno a aprender estos conceptos; casi cada libro de sistemas digitales propone su propia "máquina pedagógica", por lo que una lista detallada sería muy larga y no es el objetivo de este artículo. No obstante, una vez que se describan en la siguiente sección las características arquitectónicas deseables en un procesador pedagógico, analizaremos cómo las cumplen algunos de ellos.

3. Arquitectura y Lenguaje máquina

La arquitectura a nivel de lenguaje máquina está definida por el repertorio de instrucciones visibles por el programador [3].

Una máquina pedagógica debe tener una arquitectura ortogonal y sencilla, con los elementos indispensables para que el alumno

pueda adquirir los conocimientos que se le pretenden enseñar. Cualquier capacidad “extra” de la máquina contribuye a la dispersión de la atención del alumno.

Dada la actual tendencia RISC en el diseño de procesadores, lo más recomendable es explicar una arquitectura de carga-almacenamiento con dos espacios distintos de direcciones: un banco de registros de propósito general (BR) y una memoria. En esta arquitectura coexisten tres tipos de instrucciones:

- Instrucciones aritmético-lógicas que realizan operaciones sobre datos contenidos en el BR y escriben su resultado en un registro del BR.
- Instrucciones de transferencia que permiten mover (copiar) datos entre la memoria y el BR
- Instrucciones de salto que pueden romper el secuenciamiento implícito del programa.

Para ejecutar estas instrucciones, es preciso que la UP disponga de un BR de tamaño adecuado: basta con 8 ó 16 registros de 16 ó 32 bits. También debe poseer un registro *Contador de Programa* (PC) de tamaño acorde con la memoria (no es necesaria una memoria grande, pero puede usarse si por razones de ortogonalidad el PC tiene 16 ó 32 bits). Es necesario también un registro de instrucciones (IR), cuyo tamaño dependerá del juego de instrucciones del procesador. En general, será suficiente también con 16 ó 32 bits. Es conveniente que el tamaño de instrucción coincida con el tamaño de los datos, ya que esto evitará problemas de alineamiento y facilitará el posterior diseño de la UC y la UP. Por la misma razón, parece conveniente que el direccionamiento de la memoria sea a nivel de palabra y no a nivel de byte.

La UP debe contener una unidad aritmético-lógica capaz de realizar las operaciones requeridas por las instrucciones, y un registro de estado con tantos bits de condición (*flags*) como requieran las instrucciones de salto. Puede ser suficiente con disponer únicamente del bit de cero (Z) o se pueden incluir además el bit de signo (N), el bit de desbordamiento (V) y el bit de acarreo (C). También puede optarse por diseñar una máquina sin *flags*, en la que las instrucciones de salto consulten el valor de un registro previamente actualizado por una instrucción de comparación.

Es conveniente que la máquina disponga de los siguientes modos de direccionamiento:

- el modo registro y

- el modo inmediato para las instrucciones aritmético-lógicas,
- el modo relativo (base+desplazamiento) para las instrucciones de transferencia y
- el modo absoluto o el modo relativo (al PC) para las instrucciones de salto

La elección entre el modo absoluto o el relativo para las instrucciones de salto no es fácil. Por un lado, con una memoria tan pequeña resulta difícil justificar el modo relativo en los saltos (aunque no en las instrucciones de transferencia, ya que es imprescindible para poder programar el acceso a vectores dentro de un bucle). Por otro lado, si el alumno ya ha comprendido el modo relativo en las instrucciones de transferencia, no parece difícil que lo entienda también en las instrucciones de salto; máxime, cuando la práctica totalidad de los procesadores comerciales lo usan.

La experiencia nos demuestra que utilizar el modo relativo en las instrucciones de salto confunde innecesariamente al alumno. Por otra parte, si el PC de la máquina es un registro especial (no contenido en el BR), el uso del modo relativo al PC en las instrucciones de salto complica innecesariamente el diseño de la UP.

4. Algunos ejemplos de máquinas pedagógicas

Citaremos a continuación, a modo de ejemplo, algunas de las máquinas que se usan o se han usado en universidades españolas para ilustrar los conceptos básicos sobre AC.

4.1 La Máquina Sencilla

La Máquina Sencilla (MS) [4] tiene una arquitectura memoria-memoria, muy distinta de la que se propone en este artículo. No obstante, su extrema simplicidad la hace útil para ilustrar algunos conceptos básicos de AC (por ejemplo, las fases de ejecución de una instrucción).

Todas las instrucciones realizan las operaciones directamente sobre datos almacenados en memoria. La memoria consta de 128 palabras de 16 bits y es direccionable a nivel de palabra. El tamaño de instrucción es fijo, de 16 bits, y los datos son números naturales codificados en binario en 16 bits. El juego de instrucciones es muy reducido: tres instrucciones aritmético lógicas de dos operandos (sumar, mover y

comparar) y una de salto condicional (saltar si igual) que usa el modo absoluto para calcular la dirección de salto. Tiene un bit de condición (Z), que se actualiza con la ejecución de cualquier instrucción aritmético-lógica.

El mayor inconveniente de la MS es que no dispone de un BR, por lo que algunos conceptos importantes no pueden ser explicados sobre ella (por ejemplo, los diferentes espacios de direcciones del computador y, por tanto, algunos modos de direccionamiento). Por el contrario, la simplicidad de su UP facilita enormemente la comprensión de la UC y, en general, el funcionamiento de un procesador.

4.2 El DLX

El DLX [3] es, probablemente, uno de los procesadores más utilizados en todas las universidades del mundo y en todos los niveles de enseñanza de AC. Tiene una arquitectura de carga-almacenamiento y un sencillo repertorio de instrucciones fácilmente decodificables.

El DLX tiene dos BRs, uno de 32 registros de propósito general de 32 bits y otro de 16 registros de 64 bits de números reales representados en coma flotante (se pueden usar como 32 registros de 32 bits). El PC es de 32 bits y la memoria es direccionable a nivel de byte con una dirección de 32 bits. Las instrucciones deben estar alineadas. Tiene un tamaño fijo de instrucción de 32 bits (con un IR de 32 bits). El registro de estado está formado por un solo bit que contiene información relativa a las operaciones sobre reales. Se pueden hacer accesos a operandos de 8, 16 y 32 bits, pero es preciso que estén alineados. Dispone de los modos de direccionamiento registro, inmediato y relativo y tiene 4 tipos de instrucciones:

- Cargas y almacenamientos
- Aritmético-lógicas (enteras)
- Aritméticas de coma flotante
- Saltos (condicionales) y bifurcaciones

Las instrucciones de transferencia pueden mover operandos de distinto tamaño entre la memoria y cualquiera de los dos BRs, y también dentro de un BR.

Las instrucciones aritmético-lógicas enteras son instrucciones registro-registro. Los inmediatos son valores de 16 bits que pueden cargarse indistintamente en la parte alta o baja de un registro, lo cual permite almacenar en realidad

constantes de 32 bits usando dos instrucciones consecutivas. Hay instrucciones de suma, resta, multiplicación, división, *AND* lógica, *OR* lógica, *OR* exclusiva lógica y desplazamientos, que pueden ser aritméticos a la derecha y aritméticos o lógicos a la izquierda.

Las instrucciones aritméticas de coma flotante (comparación, suma, resta, multiplicación y división) operan directamente sobre datos contenidos en el BR de números reales. Las comparaciones en coma flotante inicializan un bit del registro de estado que puede ser examinado por dos instrucciones especiales de salto.

Por el contrario, para saltar en función de comparaciones de enteros se evalúa el valor de un registro (en lugar de usar bits de condición). Este registro se ha de escribir previamente mediante una instrucción aritmético-lógica de comparación. Se pueden hacer todo tipo de comparaciones (mayor, menor, mayor o igual, etc.).

Hay dos tipos de instrucciones de bifurcación: simple (salto incondicional) y enlace (llamada a subrutina). Las instrucciones de bifurcación pueden especificar la dirección destino del salto de dos formas distintas: con direccionamiento relativo al PC (usan un desplazamiento de 26 bits) o con el modo registro indirecto. En las instrucciones de salto, la dirección destino se especifica con el modo relativo al PC mediante un desplazamiento con signo de 16 bits. Para decidir si se efectúa el salto se comprueba si el valor de un registro es o no 0. La condición de salto se especifica en la instrucción.

El DLX es, a todas luces, un procesador muy completo; excesivamente completo para una asignatura de primeros cursos. No obstante, es posible explicar únicamente un subconjunto de su arquitectura, de modo que sea fácilmente comprensible para el nivel del alumno. En concreto, se puede explicar un subconjunto apropiado de instrucciones, y puede obviarse toda la parte del procesador que manipula números reales y trabajar únicamente con operandos enteros de 32 bits, lo que elimina el problema de alineación de instrucciones y datos.

El resultado de esta simplificación convierte al DLX en un procesador prácticamente ideal. Su UP está estructurada de forma lineal para que sea fácilmente segmentable, ventaja que puede usarse en cursos posteriores para estudiar conceptos más avanzados de AC. Para un curso de iniciación, sin

embargo, es recomendable usar una UP con registros intermedios adicionales para que cada etapa de la ejecución de una instrucción pueda ser relacionada con un estado diferente de la UC; de otro modo, la UC sería excesivamente sencilla porque las instrucciones se ejecutarían en un sólo ciclo, y por tanto al alumno podría resultarle más complicado identificar las fases de ejecución de una instrucción.

4.3 La Máquina Rudimentaria

La Máquina Rudimentaria (MR) [5] es un computador cuyo procesador tiene una arquitectura de carga-almacenamiento. La memoria está compuesta por 256 palabras de 16 bits y es direccionable a nivel de palabra. Las instrucciones tienen tamaño fijo de 16 bits y los datos son números enteros codificados en complemento a 2 en 16 bits. La UP tiene un BR de 8 registros de 16 bits, un PC de 8 bits y un IR de 16 bits. El registro de estado tiene los bits de condición N y Z, aunque en su última versión se ha incorporado también el bit V. Tiene tres tipos de instrucciones:

- Cargas y almacenamientos
- Aritmético-lógicas
- Saltos

Las cargas y almacenamientos usan el modo relativo para acceder a la memoria con un desplazamiento de 8 bits (para calcular la dirección efectiva se usan únicamente los 8 bits más bajos del registro). Las instrucciones aritméticas (sumar, restar, *AND* lógica y desplazamiento aritmético a la derecha) usan el modo registro, aunque la suma y la resta pueden usar también un operando inmediato de 5 bits. Hay 7 instrucciones de salto que utilizan el modo absoluto: una es de salto incondicional (BR) y las otras utilizan los bits de condición según todas las condiciones posibles para números enteros (BG, BL, BGE, BLE, BEQ, BNE).

La UC es muy sencilla y en ella se reflejan perfectamente las fases de ejecución de una instrucción. Existe una versión optimizada que permite estudiar cómo algunas mejoras arquitectónicas disminuyen el tiempo de ejecución de los programas.

Puede parecer que la MR cumple perfectamente con las especificaciones que en este artículo se proponen como “idóneas” para un

procesador pedagógico. Sin embargo, la UP ha sido diseñada *ad hoc* para que la UC tenga un ciclo (al menos) para cada una de las fases de ejecución de una instrucción. Eso hace que, por ejemplo, el cálculo de la dirección de acceso a memoria de las instrucciones de salto y de las de carga-almacenamiento no se realice en la ALU, sino en un sumador específico, lo que puede dar a los alumnos una visión irreal sobre cómo funcionan la mayoría de los procesadores actuales. Este diseño específico hace, además, que no sea posible segmentar su UP, por lo que la MR no puede ser usada en cursos posteriores para ilustrar algunos conceptos avanzados.

5. Lenguaje ensamblador

Una máquina pedagógica debe estar orientada fundamentalmente al estudio de la arquitectura a nivel de lenguaje máquina (LM). No obstante, una vez el alumno ha entendido cómo se ejecutan las instrucciones de LM en el procesador, es el momento ideal para que aprenda algunos conceptos básicos sobre lenguaje ensamblador (LE) y la relación entre los lenguajes de alto nivel y los de bajo nivel.

El LE de una máquina pedagógica debe ser, ante todo, muy sencillo. Además de codificar las instrucciones de LM, debe permitir el uso de direcciones simbólicas (etiquetas) y contener unas cuantas directivas básicas que ayuden al alumno a entender cómo se realiza la traducción de alto nivel a bajo nivel y de LE a LM. El conjunto de estas directivas y las instrucciones de LE deben permitirle traducir a LE y LM programas sencillos escritos en algún lenguaje de alto nivel. Esta traducción incluye variables simples o estructuradas (vectores) y sentencias estructuradas en alto nivel (bucles y condicionales). El alumno debe aprender, además, cómo el ensamblador crea la tabla de símbolos y por qué se realiza el proceso de ensamblaje en dos pasos.

Para aprender estos conceptos básicos basta con un lenguaje ensamblador que disponga de unas pocas directivas que citamos a continuación:

- de reserva de memoria, útil para definir variables, al estilo de la *.Rx núm_elementos* del VAX o la *Dx núm_elementos DUP(?)* del Pentium⁴.

⁴ La x se refiere al tamaño del dato (B, W o L)

- de definición de constantes, como las asignaciones directas “=” del VAX o “EQU” del Pentium.
 - de principio y fin de programa, para indicar cuál es la primera instrucción ejecutable y cuándo el programa ha terminado su ejecución y debe devolver el control al sistema operativo
- Es conveniente, aunque no imprescindible, que el alumno disponga de un entorno de trabajo sencillo que le permita depurar fácilmente sus programas. Esto le ayudará a enfrentarse a ensambladores reales en cursos posteriores.

6. Características de un simulador

Es aconsejable que el alumno disponga también de un simulador del procesador pedagógico que estudia. Siguiendo las pautas marcadas por las secciones previas, intentaremos analizar aquí cuáles deben ser sus características básicas.

En primer lugar, el alumno ha de poder escribir sus programas tanto en LE como en LM (hexadecimal). Esto requiere un programa ensamblador para el primer caso y la posibilidad de alterar “manualmente” el valor de la memoria en el segundo. También es conveniente que pueda cambiar durante la ejecución el valor de cualquiera de los registros de la UP. Esto le permitirá trabajar con el simulador desde el primer momento para resolver ejercicios sencillos (por ejemplo, estudiar el efecto de ejecutar una instrucción determinada a partir de un estado predeterminado del procesador).

El contenido de la memoria debe ser visible, de forma que se pueda comprobar la traducción a LM de un programa escrito en LE. Es conveniente que el contenido de una posición de memoria pueda interpretarse como una instrucción, un número natural o un número entero codificado en complemento a dos.

Es interesante que el alumno pueda ejecutar las instrucciones ciclo a ciclo para estudiar como se realizan las diferentes fases de ejecución. También debe poder ejecutarlas de forma compacta para estudiar el efecto que su ejecución produce sobre el estado del computador. En cualquiera de los casos, es conveniente que los buses por los que circulan datos y direcciones en cada momento estén resaltados en la UP, que debe estar visible en todo momento junto con la UC. De cara a analizar la ejecución de programas, es

conveniente que puedan ponerse puntos de ruptura en cualquier parte del código.

Por otra parte, resulta muy útil para el alumno que el simulador le presente diagramas de tiempo de las señales de control y de la evolución de los datos en los buses y registros de la UP.

Finalmente, es interesante que el alumno pueda imprimir sus resultados para presentar, de forma sencilla, sus prácticas al profesor. Por ello, el simulador debe estar preparado para generar diferentes tipos de informes.

Los tres procesadores descritos en este artículo cuentan con simuladores que facilitan su aprendizaje [7,3,6]. No es el objetivo de este artículo discutir las ventajas e inconvenientes de cada uno de ellos, pero es sin duda un tema que merece la pena estudiar.

7. Conclusiones

En este artículo se han presentado las características básicas que debería tener un procesador pedagógico, un lenguaje ensamblador y un simulador para dicho procesador. Como ejemplo se han mostrado tres procesadores descritos en la literatura y usados en la enseñanza y se han discutido sus ventajas e inconvenientes.

Referencias

- [1] J.J.Navarro, M.Valero-García, F.Sánchez y J.Tubella. *Formulación de los objetivos de una asignatura en tres niveles jerárquicos*. VI Jornadas sobre la enseñanza Universitaria de la informática, JENU'2000, Sept. 2000, pag. 457-462.
- [2] A.W.Burks, H.H.Goldstine y J.von Neumann. *Preliminary discussion of the logical design of an electronic computing instrument*. A.H.Taub ed., collected works of John von Neumann, vol. 5, pag. 34-79, The Macmillan Company, 1963.
- [3] J.L.Hennessy y D.A.Patterson. *Arquitectura de Computadores. Un enfoque cuantitativo*. McGraw Hill, 1993
- [4] E. Ayguadé, J.J.Navarro y M.Valero-García. *La màquina senzilla. Introducció a l'estructura bàsica d'un computador*. Col·lecció Aula. CPET, 1992
- [5] R.Hermida, A.M.del Corral, E.Pastor y F. Sánchez. *Fundamentos de Computadores*. Ed. Síntesis, 1998
- [6] E.Pastor y F.Sánchez. *La Máquina Rudimentaria: Un procesador pedagógico*. III Jornadas de Enseñanza Universitaria sobre Informática JENU'97 , Junio 1997, pag. 395-402.
- [7] J.M.Angulo. *Estructura de Computadores*. Ed. Paraninfo, 1996