RAC_{FP}: una herramienta didáctica para el estudio de la representación, algoritmos y circuitos de coma flotante

Rafael Ubal, Salvador Petit[1], Juan Carlos Cano[2] y Julio Sahuquillo[2]
[1]Facultad de Informática, [2]Escuela Técnica Superior de Informática Aplicada
Departamento de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia
Camino de Vera, s/n, 46022 Valencia
raurte@inf.upv.es, {spetit, jucano,jsahuqui}@disca.upv.es

Resumen

El diseño de herramientas pedagógicas es un desafío interesante en la enseñanza de cualquier área. En este sentido, muchas herramientas han sido propuestas como apoyo para el aprendizaje de la aritmética digital, tanto entera como en coma flotante. Este último tipo de aritmética es mucho más compleja y requiere más esfuerzo para su aprendizaje. Sin embargo, debido a restricciones temporales, los planes de estudio en cursos universitarios de ingeniería de computadores no dedican tanto tiempo como sería deseable al estudio de la aritmética en coma flotante.

En este artículo, se propone RAC_{FP}, una herramienta pedagógica diseñada para tratar los siguientes aspectos relacionados con la aritmética en coma flotante: representación, algoritmos de operaciones aritméticas y circuitos hardware reales. Estos tres niveles de abstracción se encuentran implementados independientemente en RAC_{FP}, por lo que la herramienta es útil también en otras disciplinas diferentes de arquitectura de computadores, como matemática discreta, métodos numéricos, etc., donde las implicaciones de la representación en coma flotante resulta un aspecto de especial interés.

El diseño de RAC_{FP} persigue dos objetivos principales: reducir la dificultad del proceso de aprendizaje y animar a los estudiantes al estudio de la aritmética en coma flotante. RAC_{FP} consigue estos dos objetivos gracias a su diseño en varios niveles y a su enfoque en la implementación de circuitos reales.

1. Introducción

La aritmética digital es una disciplina fundamental para muchos campos de la informática (como seguridad en redes, criptografía, procesamiento de señales, etc.), donde sirve de base a los algoritmos software e implementaciones hardware utilizados en el cálculo numérico [6].

La representación numérica en computadores abarca principalmente a los números naturales, enteros y reales. Los primeros se representan directamente en binario. Los números enteros, por otra parte, utilizan la representación en complemento a dos, debido a que simplifica los circuitos de cálculo. Por último, los números reales disponen de varias posibles representaciones, la mayoría incompatibles entre sí. Hace veinte años, el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) propuso su estándar IEEE 754 para aritmética en coma flotante [5], el cual se convirtió en un estándar *de facto* para los diseñadores de hardware.

Al contrario que en el caso de los números enteros, que pueden representarse de forma exacta en un computador, la representación de los números reales es imprecisa por naturaleza. Entre dos números reales cualesquiera, existen infinitos números reales intercalados, mientras que los computadores utilizan un número finito de bits (32 ó 64). En consecuencia, se aplican métodos de redondeo para obtener representaciones cercanas disponibles.

El cálculo numérico está estrechamente relacionado con la representación en coma flotante. Su estudio abarca una gran variedad de aspectos, desde el estudio de la representación de los reales en coma flotante hasta el diseño de circuitos complejos como el del cálculo de la raíz cuadrada. Para facilitar la enseñanza de la materia, se han propuesto ya programas informáticos pedagógicos que tratan aspectos de la representación en coma flotante y el cálculo numérico, como la representación según el estándar IEEE 754 o la simulación de algoritmos aritméticos.

En este artículo, se presenta RAC_{FP} como una herramienta software cuyo objetivo es ayudar a los estudiantes a comprender la representación de los números reales en coma flotante, los algoritmos relacionados y la implementación de circuitos basados en estos algoritmos. La herramienta está destinada para su uso en cursos universitarios de Ingeniería de Computadores, y diseñada con un fin exclusivamente pedagógico.

Para ello, la herramienta clasifica la temática relacionada en tres niveles de abstracción. El primer nivel incluye la representación interna del número real. El nivel intermedio incluye algoritmos genéricos (por ejemplo suma y multiplicación). El tercer nivel introduce a los estudiantes al mundo real con ejemplos que ilustran el funcionamiento de circuitos disponibles en el mercado.

La novedad principal de esta propuesta reside en que RAC_{FP} muestra cómo se han plasmado los algoritmos genéricos de operaciones aritméticas comunes en las implementaciones reales de circuitos. Esta idea pretende inculcar al alumno un conocimiento sólido acerca del funcionamiento del hardware dedicado a la coma flotante.

El resto del artículo está organizado de la siguiente manera: la sección 2 resume el estándar IEEE 754, presenta las recomendaciones internacionales para la enseñanza de la representación en coma flotante y argumenta cómo RAC_{FP} se adapta a ellas. La sección 3 describe la herramienta propuesta. La sección 4 incluye un conjunto de ejercicios prácticos para cubrir los objetivos de aprendizaje y muestra cómo RAC_{FP} se puede utilizar para resolverlos. Finalmente, la sección 5 presenta las conclusiones de este trabajo.

2. La enseñanza de la representación en coma flotante

2.1. La representación de números reales según el estándar IEEE 754

El estándar IEEE 754 fue, al comienzo de los años 80, el primer esfuerzo exitoso de estandarizar la representación de los números reales en coma flotante dentro de los computadores. Está basada en la *notación científica* de los números reales. Según esta notación, un número real x se escribe como $x = m \times B^E$, siendo m un número real, B un número natural fijo y E un número entero. Las variables m, B y E reciben el nombre de *mantisa*, *base* y *exponente* respectivamente.

Un número real en notación científica se puede representar de varias formas variando el exponente. Así pues, el número decimal 362.25 se puede escribir como 362.25×10^0, 36225×10^{-2}, 3.6225×10^2 ó 0.36225×10^3. Las últimas dos notaciones, en las que el punto decimal se encuentra justo antes o justo después del primer dígito diferente de cero, se denominan notaciones normalizadas. Cuando se utiliza base binaria, como es el caso de los computadores, un número real x se escribe como $x = m \times 2^E$. El estándar IEEE 754 utiliza un dígito diferente de cero para la mantisa antes del punto decimal. Por tanto, el bit más significativo de la mantisa siempre es 1, y m se expande como $1.b_0 b_1 \mathrm{K}\ b_{n-1}$.

Al utilizar el estándar, un número real se codifica con una secuencia de bits separada en tres campos: signo, exponente y mantisa. El bit de signo es 0 ó 1, indicando que el número representado es positivo o negativo respectivamente. El exponente se codifica como un número entero en exceso Z. Por último, sólo la parte fraccionaria de la mantisa queda codificada en el formato (se utiliza de ahora en adelante el término *fracción* para referirse a ella). El estándar proporciona diferentes formatos para la representación de números reales, siendo los dos más utilizados los que se muestran en la Tabla 1.

Precisión	Signo	Exponente	Mantisa
Simple (32 bits)	1 bit	8 bits	23 bits
Doble (64 bits)	1 bit	11 bits	52 bits

Tabla 1. Formatos de coma flotante de simple y doble precisión

El estándar IEEE 754 reserva dos valores del exponente para representar eventos especiales: E=0 ó E=111...1 (255_{10} para simple precisión y 2047_{10} para doble precisión). En ambos casos, el número representado depende del valor de la fracción F. La Tabla 2 resume la codificación IEEE 754 de los números en coma flotante.

Exponente (*E*)	Fracción (*F*)	Número representado
0	cualquiera	Número denormalizado: $\pm 0.F \times 2^{-2^{N-1}-2}$
0 < *E* < 111...1	cualquiera	Número normalizado: $\pm 1.F \times 2^{E-2^{N-1}-1}$
111...1	igual a 0	$\pm\infty$
111...1	distinta de 0	NaN (Not a Number)

Tabla 2. Codificación IEEE 754 para números en coma flotante. Leyenda: *N* hace referencia al número de bits del exponente (8 para simple y 11 para doble precisión)

2.2. Recomendaciones internacionales para el estudio de la representación en coma flotante

Muchas instituciones académicas en todo el mundo consideran las recomendaciones de planes de estudios de sociedades profesionales internacionales para la elección del temario de los cursos que ofrecen.

Los planes de estudio más importantes se recogen en los *curricula* propuestos por asociaciones como *Joint IEEE Computer Society* y *Association for Computing Machinery* [2][3][7]. Cada una de ellas define los conocimientos que abarcan una determinada disciplina y propone una serie de cursos para cubrirlos.

Las recomendaciones pedagógicas publicadas en 2004 registran los planes de estudio para cursos universitarios de cuatro disciplinas relacionadas con la computación: Informática (entendida como ciencias de la computación), Ingeniería de Computadores, Ingeniería del Software y Sistemas de Información, siendo la coma flotante un objetivo común a todas ellas. La Ingeniería de Computadores es una de las disciplinas que más hace énfasis en este aspecto, ya que abarca el diseño, la construcción, implementación y mantenimiento de hardware relacionado, además de la aritmética. La temática de la Ingeniería de Computadores relacionada con la coma flotante se puede resumir en los siguientes puntos:

1. Representación de números reales.
2. Aritmética multi-precisión.
3. Algoritmos para llevar a cabo operaciones de coma flotante.
4. Significado de rango y precisión en la aritmética digital.
5. Implementación hardware y software de las unidades aritméticas.

Estos puntos persiguen los siguientes objetivos:

- Conocer cómo se representan los diferentes valores numéricos en los ordenadores.
- Comprender las limitaciones de la aritmética digital y los efectos de los errores en los cálculos.
- Conocer el efecto de la aritmética del procesador en su rendimiento global.

A continuación se describen los tres niveles mediante los cuales RAC_{FP} consigue plasmar las recomendaciones propuestas en los planes de estudio internacionales de 2004 de Ingeniería de Computadores:

1. *Representación numérica y conversión entre representaciones*. Este nivel permite a los estudiantes realizar ejercicios de representación y conversión. RAC_{FP} trata tanto el estándar IEEE 754 (de simple y doble precisión) como formatos personalizados, en los que se puede especificar el número de bits destinados a la mantisa y al exponente. Los números introducidos pueden tener formato decimal, binario o hexadecimal. Los formatos personalizados permiten analizar el rango o la precisión de diferentes representaciones, resultando de gran ayuda pedagógica. Este nivel se adapta a los puntos 1 y 2 de los planes de estudio internacionales de Ingeniería de Computadores.
2. *Algoritmos aritméticos*. Este nivel permite a los estudiantes practicar con algoritmos de operaciones aritméticas, mostrando su ejecución paso a paso y los resultados intermedios. Para observar la ejecución de un algoritmo, el usuario debe seleccionar la operación deseada (por ejemplo: suma o multiplicación), el método de redondeo (por ejemplo: al par más próximo) y los operandos. La aplicación dibuja entonces un diagrama de flujo sobre el que

señala las partes activas del algoritmo en cada momento. Este nivel se adapta a los puntos 2, 3 y 4 de los citados planes de estudio.

3. *Circuitos hardware*. Una vez el alumno comprende el funcionamiento de algoritmos genéricos, puede continuar en este nivel, que lo introduce en el mundo real. Para ello, se utilizan diagramas de bloques que simulan operadores reales proporcionados por sus fabricantes. Al igual que en el nivel anterior, la herramienta muestra paso a paso los resultados intermedios, de forma que se ilustra la adaptación del algoritmo genérico al circuito real. Se ha escogido la implementación del sumador y multiplicador de Hewlett-Packard, ya que sigue casi estrictamente los pasos del algoritmo genérico. Este nivel se basa principalmente en el punto 5 de los citados planes de estudio.

3. La herramienta RAC$_{FP}$

La implementación de RAC$_{FP}$ se ha realizado sobre el entorno de programación Borland Delphi Enterprise 7 [1], y la versión se ha probado sobre los sistemas operativos Windows y Linux. Esta sección describe la interfaz de usuario. La Figura 1 muestra la ventana principal, que destaca las funcionalidades principales de la herramienta y permite acceder a cada uno de los niveles presentados en la sección 2.2. A continuación se describen los pasos que debe realizar el usuario para trabajar en cada uno de ellos.

Figura 1. Ventana principal de la aplicación

3.1. Representación numérica y conversión entre representaciones

Para trabajar con la representación en coma flotante o la conversión, el usuario debe presionar el botón *Conversión*. El cuadro de diálogo correspondiente permite introducir el número deseado y el formato al que queremos convertirlo. El número introducido puede estar en formato decimal, binario o hexadecimal. En los dos últimos casos, se asume que el número introducido sigue el estándar IEEE 754 con formato igual al de la salida deseada. El formato de la salida se puede especificar, teniendo como elección el formato IEEE 754 de simple precisión, formato de doble precisión y formato personalizado. Para el último, se especifica el número de bits de la mantisa y el exponente.

3.2. Algoritmos aritméticos

Un clic en uno de los dos botones del nivel titulado *Algoritmos* nos lleva a la ejecución paso a paso del algoritmo deseado. En la nueva ventana, se muestra un diagrama de flujo que indica el paso actual de la ejecución del algoritmo, avanzando cada vez que se pulsa el botón *Siguiente*. La primera vez que se pulsa este botón, aparece un cuadro de diálogo que permite introducir los operandos, el método de redondeo a aplicar y la operación concreta. Los métodos de redondeo posibles son los especificados en el estándar IEEE 754: a $+\infty$, a $-\infty$, a 0 y al par más próximo. Por último, se debe observar que el sumador/restador de mantisas puede trabajar en complemento a dos o en signo y magnitud [6]. Al utilizar coma flotante, este hecho puede ser transparente. En el caso de que se preste atención a este detalle, se aprecia que la operación efectiva llevada a cabo por el sumador/restador de mantisas puede diferir de la operación seleccionada por el usuario (por ejemplo, una suma codificada se puede traducir en una resta de mantisas [4]). RAC$_{FP}$ proporciona información acerca de estos casos particulares a través del botón *Tablas* en la ventana principal.

3.3. Circuitos hardware

Los estudiantes pueden familiarizarse con la implementación hardware haciendo clic en uno de los botones *Circuitos HP* de la ventana principal. La Figura 2 muestra el diagrama del circuito sumador de coma flotante de Hewlett-Packard [9].

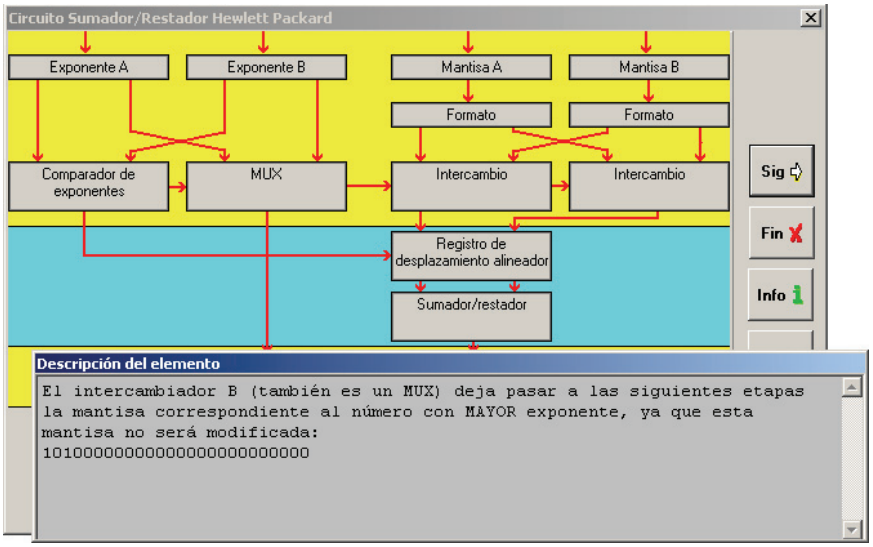

Figura 2. Diagrama de bloques del sumador de coma flotante de Hewlett-Packard, junto con ventanas informativas

Cuando los operandos se han introducido y el método de redondeo se ha seleccionado, RAC_{FP} simula el circuito paso a paso. Cuando el usuario mueve el ratón sobre cada componente del diagrama de bloques, una ventana emergente (con título *Descripción del elemento*) muestra información sobre la funcionalidad del componente y los resultados intermedios generados.

4. Ejercicios prácticos

Esta sección presenta un conjunto de ejercicios prácticos que remarcan las características pedagógicas de la herramienta. Los ejercicios se agrupan de nuevo siguiendo la estructura de los tres niveles de abstracción descritos en la sección 2.2.

4.1. Representación numérica y conversión

La Tabla 3 muestra los propósitos didácticos y el tipo de ejercicios prácticos propuestos para este nivel. Algunos ejercicios propuestos se describen a continuación.

Determinar el número mínimo de bits del exponente utilizando un formato personalizado para representar +16.0.

Objetivos didácticos y ejercicios prácticos
1. Practicar la representación numérica en el estándar IEEE 754
• Convertir ente representación decimal y representación IEEE 754 y viceversa
2. Practicar representación en formatos personalizados
• Convertir entre representación decimal y una representación personalizada y viceversa
3. Comprender el rango de los números normalizados
• Obtener el máximo y el mínimo número normalizado posible
4. Identificar excepciones
• Obtener la representación de +∞ y NaN
5. Comprender la representación denormalizada
• Obtener el máximo y el mínimo número denormalizado posible
6. Comprender los métodos de redondeo
• Representar los números de acuerdo a los cuatro métodos de redondeo
7. Analizar el error de representación
• Calcular los errores inducidos por la representación
8. Practicar la utilización de la simple y la doble precisión
• Comparar el rango de representación en ambas precisiones

Tabla 3. Resumen de ejercicios prácticos de representación.

La máxima mantisa representable es $1.\hat{1}_2$, independientemente del número de bits. Para poder representar +16.0, el exponente (*exp*) debe satisfa-

cer la ecuación $1.\hat{1}_2 \times 2^{exp} > 16.0$. El mínimo valor de exponente capaz de satisfacer esta ecuación es 4. Al ser representado el exponente en exceso Z (en concreto, $Z = 2^{n-1} - 1$, donde n es el número de bits utilizados en la representación), se debe cumplir que $(2^n - 2) - (2^{n-1} - 1) \geq 4$. Nótese que el máximo valor del exponente ($2^n - 1$) está reservado para casos especiales. Despejando, se deduce que se requieren al menos 4 bits. La Tabla 4 muestra la solución calculada con RAC$_{FP}$.

#bits del exponente	Número máximo (formato personalizado)	Número máximo (decimal)
2	0 10 11111	+3.9375
3	0 110 11111	+15.75
4	0 1110 11111	+252.0

Tabla 4. Máximo número representable para una mantisa de 5 bits, variando el número de bits del exponente

Representar el valor +1.2 asumiendo 10 bits para el exponente y variando el número de bits de la mantisa desde 5, 7, 9 hasta 11. Calcular la precisión para cada caso.

Este tipo de ejercicio es útil para comprender cómo afecta el número de bits de la mantisa a la precisión de la representación. La Tabla 5 muestra los resultados proporcionados por RAC$_{FP}$. Como se puede observar, la precisión decrece con el número de bits de la mantisa.

# bits de la mantisa	# bits del exponente	Número	Distancia
5	10	+1.1875	0.0125
7	8	+1.1953	0.0047
9	6	+1.1992	0.0008
11	4	+1.1997	0.0003

Tabla 5. Análisis de la precisión incrementando el número de bits de la mantisa.

Representar +8.25 y +∞ en el formato IEEE 754 de simple precisión. Presentar los resultados en notación hexadecimal.

Este modelo de ejercicio es básico en la enseñanza de la representación según el estándar IEEE 754. Después de representar el número a mano, RAC$_{FP}$ se puede utilizar para verificar la solución. La herramienta proporciona la representación IEEE del número real, así como detalles del proceso de conversión.

Indicar cuál es el mayor número real representable en el formato IEEE 754 de simple precisión. Obtener el valor decimal utilizando RAC$_{FP}$.

El mayor valor representable viene dado con el mayor exponente y la mayor mantisa. El mayor exponente es 11111110 (ya que 11...1 está reservado para eventos especiales) y la mayor mantisa es 11...1). El resultado proporcionado por RAC$_{FP}$ es 3.402×10^{38}.

Hay infinitos números cuya representación decimal tiene una longitud finita, pero cuya representación binaria es periódica (infinita), como por ejemplo en el caso de +1.7. Representar este número utilizando el estándar IEEE 754 de simple precisión y calcular el porcentaje de error. Asumir el método de redondeo predeterminado.

El error debido a la representación es:

$$\frac{1.70000004768372 - 1.7}{1.7} \times 100 = 2.804 \times 10^{-6}\%$$

Otros posibles ejercicios de representación (no resueltos) son los siguientes:

Los números +1, +8, +64, –1, –8 y –64 se pueden representar de forma exacta en el estándar IEEE. Para cada uno, obtener el número mayor más cercano representable y calcular la distancia entre ambos. ¿Por qué no son iguales todas las distancias?

¿Se puede representar el número 8.0×10^{42} en el formato IEEE 754 de doble precisión?

4.2. Aritmética

La Tabla 6 muestra los propósitos didácticos y el tipo de ejercicios prácticos propuestos para este nivel. Algunos ejercicios propuestos se describen a continuación.

Calcular paso a paso 40000000_{hex} + 40800000_{hex} y $0A000000_{hex}$ + $2F000000_{hex}$, donde los números vienen representados en el formato IEEE 754 de simple precisión. Utilizar RAC$_{FP}$ para comprobar los detalles de funcionamiento y los resultados intermedios de las diferentes etapas del algoritmo.

Este tipo de ejercicio se puede usar para reforzar la comprensión de un paso particular del algoritmo.

Objetivos didácticos y ejercicios prácticos
1. Calculo de operaciones básicas de coma flotante
• Llevar a cabo diferentes sumas, restas, multiplicaciones y divisiones en coma flotante
2. Familiarizarse con todas las etapas de un algoritmo aritmético
• Analizar paso a paso los resultados intermedios del algoritmo de suma de reales
3. Trabajar con diferentes métodos de redondeo
• Utilizando el algoritmo de suma, analizar los resultados según el método de redondeo seleccionado
4. Distinguir aquellas operaciones que producen excepciones
• Obtener pares de valores que producen desbordamientos inferiores o superiores
5. Trabajar con excepciones de operaciones sobre números no normalizados
• Analizar el resultado de ∞ + 0, ∞ / 0, etc.
6. Tratar la acumulación de error
• Calcular el error (en porcentaje) de diferentes operaciones

Tabla 6. Resumen de ejercicios prácticos de aritmética

Utilizando lápiz y papel, calcular la suma 2.0 + 4.0. Utilizar RAC_{FP} para comprobar el resultado.

Este tipo de ejercicio combina tanto el aspecto de representación como el de aritmética.

La Tabla 7 muestra algunos ejemplos de operaciones aritméticas que causan excepciones. Identificar las excepciones correspondientes y comprobar los resultados proporcionados por RAC_{FP}.

Suma	0 11111110 11111111111111111111110 + 0 11111000 00000000000000000000000 0 11111111 00000000000000000000000
Multiplicación	0 11111110 00000000000000000000000 × 0 10000000 00000000000000000000000 0 11111111 00000000000000000000000
División	0 00000000 00000000000000000000001 / 0 10000000 00000000000000000000000 0 00000000 00000000000000000000000

Tabla 7. Ejemplos de operaciones aritméticas que causan excepciones

Observando el gran tamaño de los exponentes de los operandos usados en la multiplicación y la suma, se puede deducir que el resultado va a ser demasiado grande para poder ser representado en el estándar. Por esta razón, RAC_{FP} detecta el desbordamiento y representa el resultado como *infinito*. En la operación de división, se produce el caso contrario, ya que el resultado es un número demasiado cercano a 0 para poder ser representado.

Otros posibles ejercicios para este nivel (no resueltos) son los siguientes:

Seguir el algoritmo de suma para calcular 0.12 + 1. Utilizar RAC_{FP} hasta que el algoritmo alcance la cuarta etapa (redondeo de la mantisa). En este paso, utilizando papel y lápiz, calcular los resultados para los cuatro métodos de redondeo expuestos. Comprobar los resultados con los proporcionados por la herramienta.

Utilizando lápiz y papel, llevar a cabo la suma $3FDFFFFF_{hex}$ + $3E800003_{hex}$. Comprobar qué método de redondeo produce desbordamiento. Utilizar RAC_{FP} para comprobar los resultados.

4.3. Circuitos

A pesar de la indudable importancia del hardware, la mayoría de cursos universitarios no incluyen ejercicios que abarquen esta categoría. Este hecho hace que los estudiantes no tengan una visión clara de cómo se aplicarían sus conocimientos en el mundo real. Uno de los principales objetivos de RAC_{FP} es motivar al alumno a trabajar con la coma flotante mostrándole su aplicación en circuitos reales. La Tabla 8 muestra los propósitos didácticos y el tipo de ejercicios prácticos propuestos para este nivel. Algunos ejercicios propuestos se describen a continuación.

Objetivos didácticos y ejercicios prácticos
1. Observar casos de estudio de circuitos reales
• Calcular paso a paso una suma identificando las etapas activas y los resultados intermedios
2. Comprender las relaciones entre las diferentes etapas de los circuitos
• Explicar la relación entre las etapas del decodificador de prioridad y el registro de desplazamiento a derechas
3. Comprender la relación entre las etapas del algoritmo genérico y las del circuito
• ¿Qué etapa del circuito HP sumador detecta un desbordamiento?

Tabla 8. Resumen de ejercicios prácticos de implementación de circuitos.

Un componente de la etapa de normalización en el circuito sumador/restador HP es el codificador de prioridad. Calcular su salida cuando la operación realizada es la resta 10.0 – 9.0. ¿Qué uso se hace de este valor en la siguiente etapa?

El valor calculado por el decodificador de prioridad es 000011. Esta parte del circuito identi-

fica la posición del primer bit significativo; su salida indica cuántas posiciones se debe desplazar la mantisa para estar normalizada y en cuántas unidades debe decrecer el exponente. Para ello, la salida del decodificador se utiliza en el registro de desplazamiento a izquierdas y en el componente decremento de la siguiente etapa.

¿Qué etapas realizan la multiplicación de mantisas en el circuito multiplicador HP?¿Por qué hay más de una etapa destinada a esta tarea? Las etapas cinco y seis llevan a cabo la multiplicación de mantisas. Son necesarias dos etapas debido a la complejidad de la operación. La quinta etapa se encarga de calcular y sumar los productos intermedios, mientras que la sexta etapa suma los resultados de la etapa anterior.

Otros posibles ejercicios para este nivel (no resueltos) son los siguientes:

Calcular paso a paso la suma 2.0 + 4.0 usando RAC_{FP} para comprobar los resultados de las diferentes etapas del circuito.

Utilizar RAC_{FP} con los operandos mostrados en la Tabla 7, comprobando las etapas donde ocurren los desbordamientos en los correspondientes operadores.

5. Conclusiones

En este artículo se ha presentado RAC_{FP} como una herramienta para la enseñanza de la representación y la aritmética en coma flotante en cursos universitarios. Para destacar los propósitos pedagógicos, los puntos a estudiar se han dividido en tres niveles de abstracción: representación numérica, algoritmos e implementación hardware. Estos niveles se adaptan a planes de estudio presentados por respetadas sociedades internacionales como IEEE y ACM.

RAC_{FP} permite la representación numérica utilizando el estándar IEEE 754 y otros formatos personalizados. Asimismo, se implementan algoritmos aritméticos, cuyos pasos pueden seguirse paso a paso observando resultados intermedios. El problema del distanciamiento del estudio de la coma flotante y su aplicación en el mundo real se solventa mediante la simulación de circuitos hardware.

La versión actual de RAC_{FP} posee una interfaz de usuario en dos idiomas (español e inglés), y ha sido probada en los sistemas operativos Windows y Linux (utilizando la librería *wine*). El código fuente está disponible en http://www.disca.upv.es/jucano/utilidades.htm. Cualquier usuario puede ejecutar directamente la herramienta o modificar el código fuente para añadir nuevas funcionalidades (por ejemplo un nuevo algoritmo aritmético). Para facilitar esta tarea, se ofrece un manual para el programador [8], en el que se especifican los pasos para crear nuevos componentes.

Agradecimientos

Este trabajo está subvencionado parcialmente por la Generalitat Valenciana (GV04B/487, GV06/326) y por la Comisión Interministerial de Ciencia y Tecnología (TIC200308154C0601).

Referencias

[1] Borland, *Borland Delphi 2005 Enterprise version*. http://www.borland.com/delphi/, 2005.

[2] *Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering*. Report of the ACM/IEEE-CS Joint Curriculum Task Force.

[3] *Computing Curricula 2001: Computer Science*. Report of the ACM/IEEE-CS Joint Curriculum Task Force.

[4] Hamacher, C., Vranesic, Z., and Zaky, S., *Computer Organization*. Fifth edition. Mcgraw Hill, 2002.

[5] I.S. 754-1985, *IEEE Standard for Binary Floating-Point Arithmetic*. IEEE Computer Society, 1985.

[6] Omondi, A.R., *Computer Arithmetic Systems. Algorithms, Architecture and Implementations*. Prentice hall, 1994.

[7] *Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*. Report of the ACM/IEEE-CS Joint Curriculum Task Force.

[8] Ubal, R., *RAC_{FP} Programmer's guide*. http://www.disca.upv.es/jucano/utilidades.htm, 2005.

[9] Ware, F.A., McAllister, W.H., Carlson, J.R., Sun, D.K., and Vlach, R.J., "64 bit monolithic floating point processors," *IEEE Journal of Solid-State Circuit*, vol. SC-17, no. 5, pp. 898-907, 1982.