Procedimientos aritméticos especializados. Computación racional exacta

Higinio Mora Mora

Tesis de Doctorado

Escuela Politécnica Superior

Directores: Dr. D. Juan Manuel García Chamizo
Dr. D. Jerónimo Mora Pascual

2003

Tesis Doctoral

PROCESADORES ARITMÉTICOS ESPECIALIZADOS

COMPUTACIÓN RACIONAL EXACTA

DEPARTAMENTO DE TECNOLOGÍA INFORMÁTICA Y COMPUTACIÓN

UNIVERSIDAD DE ALICANTE

Tesis Doctoral

PROCESADORES ARITMÉTICOS ESPECIALIZADOS

COMPUTACIÓN RACIONAL EXACTA

Presentada por:
Higinio Mora Mora

Dirigida por:
Dr. Juan Manuel García Chamizo
Dr. Jerónimo Manuel Mora Pascual

ARQUITECTURA Y TECNOLOGÍA DE COMPUTADORES

2003

A María José

Todo sistema formal que contenga a la aritmética elemental es incompleto, es decir, existen predicados de los que no se puede afirmar su veracidad o falsedad a partir de sus axiomas.

Kurt Gödel

Agradecimientos

La cita previa de Kurt Gödel pone de manifiesto la esencia misma de la vida, en la que se producen numerosas situaciones y pensamientos en los que nos asaltan las dudas. Ya podemos dedicar el tiempo que se quiera a la búsqueda de su verdad porque se ha demostrado que para la mayoría de ellas no es posible conocer su certeza. Sin embargo, se establecen algunos principios que, por definición, son aceptados como válidos. Quizá el más famoso sea el enunciado por Descartes: *pienso, luego existo*. A este conjunto añado también: *errar es humano*.

Al terminar este documento doy por cierto otro axioma más: el hecho de que para realizar una tesis doctoral se necesita algo más que una buena idea. Es necesario espíritu crítico, humildad, esfuerzo, mucha voluntad, constancia y, sobre todo, un entorno de personas a tu alrededor que te presten apoyo en los momentos difíciles, consuelo en la desilusión, ánimo en la preocupación, consejos en la duda y, como no, algún que otro disgusto.

Una verdad muy importante para mi es el sentimiento hacia mi novia María José. Te quiero. Gracias por estar a mi lado y mantenerme vivo.

Algún día sabré agradecer tu infinita paciencia, el hecho de sentirme afortunado por tenerte a mi lado y de coincidir contigo el día que te conocí.

Quiero expresar mi especial gratitud hacia mis padres, Higinio y Finita; hermanos María José, Cibeles, y Miguel Ángel; y amigos, quienes han sabido comprender mis altibajos de humor, carencias de atención y numerosas ausencias consecuencia de lo absorbente de esta empresa.

Mi más sincero agradecimiento a quienes me introdujeron en la investigación producto de esta tesis y en el seno de la ciencia. Gracias por dirigir mi trabajo por el camino correcto a base de método y juicio crítico. Gracias por vuestra dedicación y sabios consejos. Gracias Juanma, gracias Jerónimo.

A mi compañero de despacho, José Luis, así como a los chicos de la habitación de al lado Mora y José García, compañeros y amigos desde los comienzos de esta investigación. Vaya una mención especial a María Teresa y Goyo, compañeros de viaje por las arquitecturas aritméticas, quienes han contribuido a divulgar nuestro trabajo por medio mundo. Gracias a Andrés Fuster, Jorge, Soriano, Toni, Javi, Pujol, Flórez, Paco Maciá, Dani, Joan Carles, Ana, Juan Antonio, David, Vicente, José Luis, Antonio, Andrés Almela, Anabel y a todos los que no encuentren su nombre en estas líneas, por sus aportaciones y la valiosa ayuda prestada.

Termino enunciando un nuevo axioma de mi particular sistema: no dudéis en pedir mi ayuda en todo aquello que estiméis necesario.

Higinio Mora Mora

Alicante, 1 de octubre de 2003

Contenido

Índice de figuras

Índice de tablas

Presentación

El procesamiento de números es una parte imprescindible de la solución de muchos problemas computacionales. A modo ilustrativo se presentan algunos ejemplos de cálculo específico: los procesadores digitales de señal (DSP —*Digital Signal Processor*) [Kloos et al, 2002], [Wires, 2000], [Kneip et al 1994], [Weiss, 1991] y los procesadores multimedia (MMP —*MultiMedia Processor*) [Garrido et al, 2002], [Villalba et al, 2002], [Thakkar y Huff, 1999], [Oberman et al, 1998] favorecen la manipulación fluida de la información digital; los procesadores de comunicaciones [Wolf y Franklin, 2002], [Paulin et al, 2001], [Xiaoning et al, 1999] y los procesadores criptográficos [Neumann, 1999], [Smith y Weingart, 1999], [Yee y Tygar, 1995] gestionan el flujo de información de la red y contribuyen de manera decisiva al desarrollo del comercio electrónico.

La operatoria de los ejemplos anteriores contiene numerosos aspectos que difícilmente encajan en una solución global: cantidad y tipo de las operaciones, dominio de las funciones, naturaleza de los datos, rango numérico de los operandos y resultados, … Una de las cuestiones de mayor interés consiste en encontrar el equilibrio entre precisión y

complejidad de los cálculos. En este sentido, la disponibilidad de procesadores con capacidades de cálculo flexible es una herramienta útil para ajustar los resultados a los requerimientos.

La investigación que se presenta en esta memoria está orientada hacia la concepción de técnicas de representación exacta de números racionales en su notación fraccionaria y de operatoria con estos formatos de datos para producir resultados de precisión ajustable. Se han considerado los aspectos arquitecturales de rendimiento y complejidad al nivel de realización de las primitivas del procesador y se ha evaluado la solución basándose en pruebas comparativas de simulaciones. Todo ello se estructura en los capítulos que se describen a continuación:

El primer capítulo contiene la *introducción* de este trabajo. Se revisa el estado actual del conocimiento sobre la materia y se describen las propuestas más relevantes. A partir de ahí, se formula el problema de procesamiento exacto y se propone un modelo de solución basado en la concepción de una unidad aritmética flexible compuesta por funciones con propiedades de procesamiento ajustable.

En el segundo capítulo se describe el *formato de representación numérica de los números racionales* y se propone su *instrumentación en precisión variable*. La principal característica del esquema es su capacidad de expresión exacta.

En el tercer capítulo se expone la *metodología de operación* que se utiliza en el desarrollo del resto de los operadores aritméticos basada en esquemas iterativos y memorias con resultados precalculados.

En los capítulos cuarto y quinto se propone el *método de cálculo de las funciones suma y producto* así como su *instrumentación en precisión variable* que preserva las características de la representación en relación con la capacidad de expresión.

En el último capítulo, *conclusiones*, se enumeran las aportaciones más destacables de esta memoria y se plantean trabajos futuros para continuar con la investigación realizada que profundicen en algunos aspectos o desarrollen otras cuestiones relacionadas.

Esta tesis constituye un avance en la investigación sobre procesadores especializados a partir del cual abordar los aspectos instrumentales y de realización al nivel de investigación pre-competitiva e incluso de explotación rentable. El desarrollo de esta tesis forma parte de la investigación llevada a cabo en los proyectos de investigación: CICYT TAP98-0333-C03-03: *Sistemas de Visión para Navegación Autónoma*; y MCYT DPI2002-04434-C04-01: *Visión Mediante Periférico Robótico Inteligente para Sistemas Móviles Autónomos*.

Capítulo I

Introducción

1. Motivación
2. Objetivos
3. Conocimiento actual y problemas abiertos
4. Formulación del problema y propuesta de solución

I - 1

Motivación

Pensemos por un momento en el siguiente problema:

> *Sea la ecuación de primer orden*
>
> $10x = 6$
>
> *Su resultado es de sobra conocido y fácil de calcular, despejando x se obtiene un valor de* $x = 0{,}6_{10}$
>
> *Ese número, expresado en notación fraccionaria posicional, indica con precisión absoluta cuál es el valor de la variable x que cumple la relación anterior. Sin embargo, con la mayor parte de la algoritmia y de la tecnología disponible actual no se obtiene su resultado exacto, sino tan sólo una aproximación.*

Esta circunstancia se produce por la naturaleza de la representación binaria posicional del número decimal 0,6 que se compone de infinitas cifras fraccionarias. A este respecto surgen multitud de interrogantes acerca de las características de la representación numérica: ¿cuántas

cifras significativas posee un número? ¿en qué medida influye la base de representación? ¿qué conjuntos numéricos provocan esta situación? … Su análisis nos llevará a dilucidar si existen codificaciones numéricas alternativas para estos números o bien se tiene que renunciar a la expresión de su valor exacto mediante representación posicional. La medida que frecuentemente se adopta pasa por acometer aproximaciones al valor correcto utilizando una cierta cantidad de cifras [IEEE, 1985].

Determinadas actividades requieren elaborados cálculos matemáticos y, adicionalmente, cuentan con importantes restricciones de precisión. De inmediato se plantean situaciones en las que resulta conveniente, si no necesario, disponer de la expresión del valor exacto, al menos para dominios numéricos como el del problema anterior: cálculo de trayectorias en sistemas de guiado y posicionamiento, comunicaciones de alta frecuencia, alineamiento de antenas y, en general, en aquellos casos en los que la relación entre el tamaño de los operandos sea desproporcionada. En estos problemas, imprecisiones poco significativas en los cálculos pueden provocar desviaciones de consideración en los resultados que se obtienen. En relación con esta cuestión se deben tener en cuenta las incorrecciones que pueden surgir de las limitaciones de los actuales esquemas de representación y los errores que provocan en las operaciones [Schulte, 2000], [Michelucci y Moreau, 1997], [Goldberg, 1991], [Bohlender, 1990], [Ratz, 1990] y [Hoffmann, 1989]. Por otro lado, en algunas aplicaciones existen restricciones temporales que exigen disponer de la respuesta en un tiempo determinado y reducido. En este contexto, el empleo de métodos de alto nivel puede no ser adecuado para cumplir satisfactoriamente con estos requerimientos. Una posible línea de actuación consiste en traerse al nivel físico soluciones que tradicionalmente han sido del dominio superior: el desarrollo e implementación de operadores que junto con otros elementos y mecanismos del procesador regulan la precisión de los operandos y resultados y alcanzan su representación sin error y, al mismo tiempo, abren la posibilidad de proveer una respuesta ajustable en el tiempo desde el nivel más bajo de la arquitectura [García et al, 2003a], [García et al, 2003c], [Mora, 2001].

I - 2

Objetivos

El objetivo general de este trabajo consiste en avanzar en el conocimiento y en el desarrollo de modelos para procesadores de alto rendimiento y prestaciones. En particular, proporcionar el soporte formal para la concepción de métodos que permitan la representación numérica fraccionaria posicional adecuada para la operatoria con control de la precisión. La casuística es amplia, por lo que difícilmente se puede establecer un modelo general que aporte soluciones de interés aplicado para todos los casos. Se busca, en cambio, una solución que resuelva satisfactoriamente una familia de problemas desde el nivel bajo de la arquitectura. En concreto la investigación se centra en el conjunto de los números racionales y en algunas operaciones básicas.

Debido a la estrecha relación que existe entre la precisión de los resultados y el tiempo de procesamiento, se establece como consideración complementaria la consistencia del modelo con las restricciones temporales. Es decir, se debe proporcionar la flexibilidad necesaria para gestionar el par precisión y tiempo de procesamiento para cada problema.

El objetivo general se concreta en otros parciales relacionados con la representación de los datos y el modelo aritmético. Con respecto al método de representación de la información numérica se establecen los siguientes fines:

- Proporcionar suficiente capacidad expresiva para representar de manera exacta cualquier número racional en notación fraccionaria posicional.
- Extender el esquema de coma flotante tradicional para proporcionar mayor amplitud de expresión numérica.
- Facilitar la implantación hardware y el diseño de técnicas de cálculo flexibles que no penalicen el rendimiento.

En cuanto al modelo de cálculo aritmético se contemplan los siguientes cometidos a alcanzar:

- Constituir un conjunto de operadores sobre la base del esquema de representación propuesto para los números.
- Concebir las operaciones matemáticas con capacidad para realizar cálculos exactos al operar con números racionales y dotarlas de la flexibilidad suficiente para adaptarse a los requerimientos del hardware.

La investigación se orienta a la concepción de un coprocesador específico: Procesador Racional Flexible (FRP— *Flexible Rational Processor*) capaz de procesar números racionales expresados en el formato de representación posicional exacta. Dicho procesador incorporará las operaciones básicas suma y producto como base para la construcción de otros operadores por combinación. Se propone la arquitectura para la realización de dichos métodos de cálculo, teniendo en cuenta la propia implementación de los operadores elementales y el estudio de los elementos de memoria necesarios.

De forma esquemática, la funcionalidad que deberá proporcionar este coprocesador se resume a continuación:

- Almacenar números racionales sin error.
- Procesar números racionales de manera exacta para las operaciones suma y producto.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011110100

- Gestionar la precisión de los resultados en función de las restricciones existentes.

La operatoria se orienta hacia técnicas iterativas de procesamiento paralelo. Este trabajo engloba, tanto aspectos del ámbito de la investigación básica (concepción y especificación de modelos aritméticos de representación y procesamiento parametrizables), como aplicada (realización de las propuestas de procesador).

La posibilidad de realizar operaciones sucesivas sin acumular error apunta hacia una algoritmia donde la planificación de operaciones se realiza según parámetros de tiempo y precisión. Su gestión conjunta permite adaptar las capacidades del procesador a situaciones que interactúan fuertemente con el mundo real: sistemas de guiado por visión, sistemas de seguridad y control o en sistemas multimedia. En todos ellos se plantea el problema del procesamiento intensivo de datos proporcionados por periféricos. En estos casos se produce, necesariamente, una coordinación estrecha entre sensores y actuadores del propio dispositivo y de éste con el resto de los niveles del sistema, debiéndose profundizar en el determinismo de los tiempos de respuesta de los métodos empleados así como de la precisión de los cálculos. Ello, favorece su integración en la planificación junto con el resto de las tareas del sistema.

A continuación se trata el estado actual de la investigación referente a los métodos de representación numérica y los modelos de computación variable más relevantes para disponer de una base rigurosa sobre la que terminar de centrar los objetivos de la investigación. Posteriormente, se formaliza el problema y se propone la solución que será desarrollada a lo largo de esta memoria.

I - 3

Conocimiento actual y problemas abiertos

En este apartado se desarrolla el estado del arte acerca de la representación numérica en un computador y de su operatoria asociada. La relevancia de este problema se constata en la abundancia de propuestas y trabajos que tratan de resolver determinados aspectos del mismo. El análisis se realiza a varios niveles y abarca las soluciones de alto nivel, los formatos de codificación numérica, los modelos aritméticos para el cálculo y las unidades procesadoras que operan sobre estos esquemas. La exposición del estado actual de la investigación parte desde las herramientas software por su amplia utilización y flexibilidad, seguidamente se analiza la algoritmia para computación numérica de alta precisión y los métodos convencionales de representación para continuar con aquellas propuestas a bajo nivel que tratan de implementar capacidades específicas que no proveen los sistemas convencionales.

La práctica más habitual consiste en recurrir a herramientas software de precisión variable por razones de portabilidad. Esta solución forma una

11,0010010000111111010101000100010000101101000110000100011010011000100111000110001100110001010001011100000001101110000011100110100010010100100000010010011100000100010001001100111110011000111010100

capa de abstracción por encima del hardware del computador que dota al sistema de estructuras y operaciones de longitud ajustable a las necesidades de cada problema. Entre estas herramientas se encuentran librerías de funciones y tipos de datos, paquetes aritméticos y otras extensiones de los lenguajes de programación comunes (C, Pascal, Fortran, MatLab) [Press et al, 1994], [Aberth y Schaefer, 1992], [Klatte et al, 1991], [Smith, 1991]. Su principal contribución se encuentra en ampliar el espacio de memoria de la representación para almacenar una mayor cantidad de cifras significativas y alcanzar una mayor precisión. Sin embargo, la complejidad en el cálculo de las operaciones limita su uso para el caso general. Por otra parte, la representación última se apoya en formatos y esquemas de representación numérica soportados por el hardware de la máquina que propaga en cierta medida sus restricciones.

La algoritmia para el cálculo de alta precisión está dirigida generalmente a operar con datos de considerable longitud y a compensar las limitaciones de los formatos de representación estándar. En este último aspecto cobran especial interés las propuestas de procesamiento numérico decimal que evitan los errores de conversión a binario en la introducción de datos [Cowlishaw, 2003], [EU, 1999], [Bohlender, 1991]. Para el cálculo de operandos de un número elevado de cifras destacan los métodos basados en procedimientos que reducen su tamaño [Karatsuba y Ofman, 1963], [Toom, 1963] y los métodos recursivos de obtención progresiva de cifras significativas del resultado. En este conjunto se encuentran los conocidos algoritmos de *Newton-Raphson*, *Goldschmidt* y *Taylor* [Ercegovac et al, 2000b], [Ito et al, 1997], [Schulte, 1994] y la *aritmética on-line* [Schneider et al, 2000], [McIlhenny y Ercegovac, 1999], [Muller, 1991], [Ercegovac y Trivedi, 1987], [Ercegovac y Trivedi, 1977]. Ésta última se caracteriza por realizar cada operación desde las posiciones más significativas a las menos significativas de los números. Algunos de estos algoritmos se incluyen en las herramientas software mencionadas, por lo que adquieren sus inconvenientes, si bien se han desarrollado propuestas hardware de su implementación que serán mencionadas más adelante.

Los formatos de codificación numérica vienen condicionados fundamentalmente por las características de los conjuntos numéricos a

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110010100010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

representar, existiendo distintos convenios y formatos según se representen números naturales, enteros o reales [Patterson y Hennessy, 2002], [Kornerup, 1994], [Omondi, 1994]. Para la representación de estos últimos, la notación fraccionaria posicional ofrece una expresión directa del número donde cada cantidad consiste en una parte entera y en una parte fraccionaria que indican fielmente su valor.

El esquema de representación fraccionaria más sencillo es el de coma fija. Este formato se caracteriza por destinar una cantidad determinada de memoria a la parte entera y a la parte fraccionaria de modo que se representan siempre las cifras del número en torno a su coma fraccionaria. Esta característica proporciona ventajas en el diseño de unidades aritméticas al utilizar en su procesamiento aritmética de enteros que posee una menor complejidad en los diseños [Patterson y Hennessy, 2002], aunque como inconveniente se destaca que sólo es capaz de representar de manera exacta números que coinciden con los elementos discretos codificables y realiza aproximaciones para el resto. Este formato es ampliamente utilizado en arquitecturas de procesadores digitales de señal DSP [Kim et al, 1998], [Inacio y Ombres, 1996].

Con el fin de abarcar un mayor intervalo de representación toman especial relevancia los sistemas de representación en coma flotante, en los que la parte entera se reduce a su mínima expresión, desarrollando ampliamente la parte fraccionaria. Esta representación se asemeja a la expresión científica de los valores numéricos reales y mejora la capacidad expresiva del formato de coma fija. [Patterson y Hennessy, 2002], [Sun Microsystems, 2000]. La codificación del número se estructura en tres campos: signo, exponente y mantisa. La mantisa constituye la parte significativa del número mientras que el exponente representa su orden de magnitud. Al igual que en la representación en coma fija, estos formatos establecen un esquema discreto de representación de los números reales en un computador. Como modelo característico de los esquemas de coma flotante cabe destacar la *norma de representación de datos en coma flotante IEEE-754* [IEEE, 1985], ampliamente adoptada por la gran mayoría de los sistemas informáticos. Este modelo describe varios formatos de representación, modos de redondeo y manejadores de excepciones.

11,0010010000111111010101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001000101001111100110001110100

En esos esquemas, llamados de precisión finita por utilizar una cantidad finita de bits en la representación, resulta patente la imposibilidad de representar cualquier valor fraccionario que supere la cantidad de cifras significativas del formato [Sun Microsystems, 2000]. La consecuencia es que se provoca un error en el cálculo [Goldberg, 1991] que dificulta resolver satisfactoriamente problemas que requieren una alta precisión en los resultados, como por ejemplo [Clark, 1998], [Hoffmann, 1989]: comunicaciones de alta frecuencia, simulaciones científicas, navegación y posicionamiento, etc.

En aquellos problemas que requieren de capacidades que desbordan los sistemas actuales, las investigaciones se orientan hacia diseños dedicados que tratan de cubrir a bajo nivel las carencias concretas existentes. Estas soluciones potencian determinadas características que no han sido suficientemente desarrolladas o bien añaden nuevas funcionalidades que las hacen apropiadas para resolver cierta clase de problemas. Ejemplos de propuestas de este tipo se encuentran en [Wolf y Franklin, 2002], [Villalba et al, 2002], [Mora, 2001], [Paulin et al, 2001], [Wires, 2000], [Nielsen y Kornerup, 1999] y [Nielsen, 1997]. La representación numérica no es ajena a esa situación y del mismo modo abundan las soluciones a medida válidas para determinados entornos. Las investigaciones realizadas se concretan en la búsqueda de métodos alternativos de expresión numérica y de cálculo aritmético implementados en diseños específicos de procesadores especializados. En lo referente a estas aportaciones, se presentan los avances realizados así como las carencias más relevantes. Se ha realizado una ordenación de las propuestas según su capacidad de expresión, entendiendo como tal, la precisión a la hora de expresar tanto los operandos como el resultado que produce su procesamiento.

En primer lugar, como método de mayor expresividad, se encuentra el modelo de computación simbólica, caracterizado por realizar una representación exacta de los datos con los que opera. Esta técnica, que constituye el esquema de *aritmética exacta* [Mencer, 2000], consiste en expresar las operaciones matemáticas mediante expresiones algebraicas que corresponden con valores numéricos. Basándose en estos principios se dispone del diseño de una unidad aritmética [Kornerup y Matula, 1983a], [Kornerup y Matula, 1983b] en la que se realiza la

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

representación de los números de forma simbólica mediante fracciones. Se contemplan las operaciones básicas de suma, resta, multiplicación y división, operando sobre los numeradores y denominadores de las fracciones con aritmética de enteros. En esta propuesta se representan y operan los números racionales sin error, presentando igualmente una expresión analítica del resultado. No obstante, el procesamiento con estas expresiones es costoso computacionalmente en especial cuando no se encuentran formas de simplificación [Buchberger, 1991]. Además, para expresar el resultado mediante un único valor numérico es necesario realizar una operación de división que puede provocar errores de aproximación.

Las fracciones continuas también ofrecen un método de representación exacta del conjunto de los números racionales. En este caso se realiza la codificación de los números mediante sucesivas fracciones de enteros [Brezinski, 1980], [Moore, 1964]. A pesar de ello, los diseños hardware realizados [Mencer, 2000], [Mencer et al, 1999], [Seidensticker, 1983], [Robertson y Trivedi, 1977], revelan una alta complejidad de las operaciones aritméticas [Vuillemin, 1990] y, al igual que en la computación simbólica, en caso de que se quiera un resultado numérico se deben realizar operaciones adicionales cometiéndose imprecisiones al transformar las expresiones.

Una interesante propuesta para la codificación sin error de números racionales consiste en la representación explícita del desarrollo periódico de los números fraccionarios [Hehner y Horspool, 1979]. Sin embargo este trabajo se limitó sólo a su formulación y no se presentaron procedimientos adecuados para su procesamiento ni arquitecturas que lo implementaran. En consecuencia, no captó el suficiente interés por la comunidad científica y fue abandonado al ser una propuesta carente de realismo.

Otro conjunto de propuestas se fundamenta en un modelo de aritmética de intervalos. El objetivo que persiguen consiste en acotar la imprecisión cometida al proveer un resultado numérico. En este caso, un número viene expresado por los extremos del intervalo en el que se encuentra, los cuales son codificados mediante notación en coma flotante [Arnold et al, 2003], [Schulte, 2000], [Hormigo el al, 2000], [Schulte y

Swartzlander, 2000], [Hormigo et al, 1999], [Sáez et al, 1998], [Knuppel, 1994], [Neumaier, 1990], [Alefeld y Herzberger, 1983], [Moore, 1979]. Las operaciones aritméticas se realizan sobre los extremos del intervalo conservando los resultados incluidos entre sus límites. Los diseños hardware basados en esta técnica mejoran notablemente el coste temporal frente a paquetes software de simulación con la misma funcionalidad. Como ejemplos representativos se destacan los siguientes:

- La *arquitectura AIX* [Kolla et al, 1999] empaqueta los extremos de los intervalos en la misma palabra del computador para que sea tratada en paralelo por la unidad aritmética. Se sugiere el uso de datos de doble precisión para almacenar cada intervalo de manera que el cálculo sea realizado en una unidad compatible con la norma de IEEE. Su principal limitación se encuentra en que la cantidad de cifras destinadas a representar cada intervalo es fija y acotada por un formato de representación de precisión finita.
- El *procesador VP de aritmética de intervalos* [Schulte, 2000], [Schulte, 1996] consiste en una estructura de representación en coma flotante en la que el campo que almacena la mantisa admite una longitud variable aunque limitada por el propio formato. Ofrece una precisión considerablemente mayor que el modelo anterior utilizando intervalos más estrechos. Sin embargo, aunque se aumente la precisión de los extremos, el grado de aproximación está acotado por la cantidad de cifras significativas a representar.

El mayor inconveniente de esta técnica radica en la renuncia a la obtención de un único valor exacto, provocando falta de precisión al producir intervalos demasiado amplios como consecuencia de determinadas operaciones y cálculos. No obstante, se produce un avance con respecto a las técnicas anteriores al mantener el error del resultado numérico acotado por los límites del intervalo. Aprovechando esta circunstancia se han realizado propuestas combinadas entre la aritmética de intervalos y la notación simbólica: la técnica de aritmética relajada, comúnmente conocida como *lazy arithmetic* [Michelucci y Moreau, 1997], [Benouamer et al, 1993], describe los resultados mediante una expresión matemática simbólica y al mismo tiempo los expresa numéricamente a través de un intervalo. Con este

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

procedimiento, en caso de requerir más precisión se realizan de nuevo los cálculos obteniendo una mejor aproximación.

Otras investigaciones tratan de incrementar la precisión de los valores numéricos con los que trabajan a costa de utilizar una mayor cantidad de cifras significativas para representar los operandos y los resultados. De este modo se pretende alcanzar una aproximación *adecuada* a las necesidades del problema sobre el que se aplica. No obstante, aunque estos sistemas proporcionan una única expresión numérica de los datos, son incapaces de codificar números con una cantidad infinita de cifras fraccionarias, perdiendo la representación exacta de valores racionales y cometiendo errores en su procesamiento. En este conjunto de propuestas se encuentran aquellas basadas en la *aritmética escalonada* y la *aritmética on-line*.

La *aritmética escalonada* [Priest, 1991], [Dekker, 1971] representa cada número mediante una lista variable de valores en coma flotante no solapados. La suma de estos elementos proporciona el valor del número que codifican. Las operaciones aritméticas se realizan sobre esa lista de números, obteniendo como resultado otro valor representado con el mismo formato. Esta propuesta ha sido implementada en el *procesador para aritmética de intervalos escalonada* [Schulte y Swartzlander, 1995]. Su principal ventaja es la de poder usar las unidades de coma flotante convencionales para datos en formato estándar de IEEE, mientras que como inconvenientes se tiene que la precisión de los valores escalonados es limitada y el proceso de cálculo y conversión a este formato es complicado y costoso. Por ejemplo, la simple comparación entre dos números se convierte en un algoritmo complejo por el hecho de que para un mismo número pueden existir múltiples representaciones distintas.

En la *aritmética on-line* los operandos se introducen en serie y el procesamiento se realiza dígito a dígito a partir del conocimiento parcial de los datos de entrada. De esta forma pueden diseñarse métodos de cálculo segmentados que permitan un encadenamiento de las operaciones y, además, debido a sus propiedades iterativas, es posible establecer estructuras regulares que pueden ser usadas para el cálculo de las operaciones elementales con un número variable de cifras. Entre

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011110100

las propuestas que se basan en este tipo de aritmética se encuentran varios diseños:

- El *coprocesador JANUS* [Guyot et al, 1989] considera una precisión máxima de 600 dígitos, implementando tan sólo la operación de multiplicación.
- El *coprocesador VLP* [Ferreira, 1998] consiste en un coprocesador aritmético desarrollado bajo plataforma FPGA con capacidades de reconfiguración dependiendo de las operaciones a realizar.

Finalmente, con el mismo objetivo de aumentar la cantidad de cifras significativas de los números, existen variantes hardware capaces de trabajar con datos de una cantidad variable de dígitos:

- El *procesador CACAD* [Cohen et al, 1983] realiza la codificación de los datos mediante palabras de longitud variable. Cada palabra contiene los campos de signo, exponente, longitud de la mantisa, mantisa y un indicador. La mantisa se representa en BCD [Hull et al, 1991]. En su diseño se evitan los errores de representación en las interacciones de entrada y salida consecuencia de las transformaciones decimal a binario. Presenta como inconvenientes la complejidad adicional de la unidad aritmética al operar con datos expresados en BCD así como la limitación en la precisión representable.
- El *coprocesador VP para FPGA* [Hsu, 1996] supone un avance respecto al diseño anterior al no limitar la cantidad de cifras de la mantisa significativa del número. En este caso, se utiliza una estructura basada en una cantidad variable de palabras de 64 bits, destinando una cantidad fija de cifras a la representación de la mantisa y al exponente en cada bloque. Se da la posibilidad de concatenar varias palabras hasta completar la representación del número. Asimismo, se describen los algoritmos necesarios para operar con este tipo de datos, detallando los accesos a la memoria en la que se almacenan.

Las representaciones aproximadas con una cantidad determinada de cifras, entre las que se incluyen los formatos de representación convencionales, suelen disponer de métodos de refinamiento o redondeo de la codificación que producen con la misión de ajustarla al espacio disponible y cometer así el mínimo error. El efecto que

11,0010010000111111011010100010001000010110100011000010001101001100010011000110010100010111000000011011100000111001101000100101001000000100100111000001000100100111001111100110001110100

provocan es la modificación de las cifras menos significativas de la representación respecto a las cifras exactas del número. Entre los criterios de redondeo más usuales se encuentran el redondeo por exceso, por defecto, al más cercano y el redondeo al par [Bruguera y Lang, 2000], [Park et al, 1999], [Quach et al, 1991] y [IEEE, 1985]. El procedimiento que se utiliza en su instrumentación está estrechamente relacionado tanto con el método de representación que se utilice como con la operación que provea el resultado a redondear [Even y Seidel, 2000], [Oberman, 1996].

La investigación en procesamiento numérico y precisión variable, frecuentemente, forma parte de algún proyecto de diseño y desarrollo de un dispositivo de procesamiento específico. En el entorno geográfico más cercano se mencionan los siguientes grupos de investigación con trabajos en la materia:

El *departamento de arquitectura de computadores* de la *Universidad de Málaga* realiza investigaciones para el desarrollo de procesadores especializados en cálculo numérico. Con este objetivo se destacan los trabajos encaminados a concebir una arquitectura de precisión variable basada en aritmética de intervalos [Hormigo el al, 2000], [Hormigo et al, 1999], [Sáez et al, 1998].

El *grupo de arquitectura de computadores* de la *Universidad de Santiago de Compostela* se centra en varios aspectos de la disciplina destacando sus trabajos sobre la aritmética del computador. En esta materia desarrollan algoritmia de cálculo de altas prestaciones para procesadores tanto de propósito general como de aplicación específica [Piñeiro y Bruguera, 2002], [Piso et al, 2002], [Bruguera y Lang, 2001].

En el ámbito internacional se referencian a continuación los grupos de investigación más relevantes en esta línea:

El *laboratorio de arquitectura reconfigurable y aritmética digital* de la *Universidad de California en Los Ángeles (Estados Unidos)* destaca por su profunda investigación en nuevos métodos de cálculo de funciones matemáticas [Benowitz et al, 2002], [Ercegovac et al, 2000a], [Ercegovac et al, 2000b]. Sobresalen sus aportaciones en la aritmética on-line y en el procesamiento en precisión variable [Schneider et al, 2000], [McIlhenny y Ercegovac, 1999], [Tenca y Ercegovac, 1998], [Ferreira, 1998].

El *laboratorio de arquitectura de computadores e investigación aritmética* de la *Universidad de Lehigh en Pennsylvania (Estados Unidos)* trabaja en la realización de un proyecto para dotar de soporte hardware a la operatoria de precisión variable basándose en aritmética de intervalos [Arnold et al, 2003], [Schulte et al, 2000], [Schulte, 2000], [Schulte y Swartzlander, 2000].

El *grupo de arquitectura y aritmética de computadores* de la *Universidad de Stanford (Estados Unidos)* se orienta al desarrollo de un procesador *Stanford Nanosecond Arithmetic Processor* (proyecto SNAP) en el que se presta especial atención al tiempo de respuesta de las operaciones aritméticas y su implementación [Fanhm y Flynn, 2003], [Liddicoat y Flynn, 2001]. Se realizan otras investigaciones relacionadas con la concepción y el diseño de unidades aritméticas sobre números racionales con operatoria basada en fracciones continuas [Mencer, 2000], [Mencer et al, 1999].

El *laboratorio de informática* de la *Escuela Superior de Lyon (Francia)*, trabaja en el proyecto *Arenaire* para la creación de conocimiento en el área de aritmética de computadores. En este contexto se desarrolla investigación en métodos numéricos y en cálculo en precisión variable así como en la construcción de librerías y funciones software para su procesamiento [Boldo y Daumas, 2003], [Lefevre y Muller, 2003], [Muller, 2003].

El *departamento de matemáticas e informática* de la *Universidad de Southern Denmark (Dinamarca)* realiza investigación en el ámbito de sistemas de representación numérica y unidades aritméticas de procesamiento matemático intensivo. En esta línea destaca su aportación al procesamiento racional basado en fracciones continuas y representación simbólica de fracción [Kornerup, 2003], [Nielsen y Kornerup, 1999], [Nielsen, 1997].

El *laboratorio de sistemas informáticos avanzados* de la *Universidad de California en Davis (Estados Unidos)* se centra en estudiar la relación entre los algoritmos aritméticos y la tecnología en la que se implementan con el propósito de mejorarlos en función de la misma [Oklobdzija et al, 2003], [Yu et al, 2003], [Nedovic et al, 2002].

11,0010010000111111011010101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010100

Otros grupos destacan por su importante investigación en arquitecturas aritméticas y en el desarrollo de algoritmia para el procesamiento de operadores matemáticos. En este conjunto se menciona el *laboratorio de procesadores numéricos de la Universidad de California en Irvine (Estados Unidos)* [Antelo et al, 2002], [Lang y Antelo, 2001]; el *departamento de informática e ingeniería* de la *Universidad Southern Methodist de Dallas (Estados Unidos)* [Paul y Seidel, 2003], [Even et al, 2003], [Seidel et al, 2001]; el *departamento de sistemas de información* de la *Universidad de Nagoya (Japón)* [Kaihara y Takagi, 2003], [Takagi y Horiyama, 1999], [Takagi, 1998] y el *grupo de arquitecturas y redes de computadores* de la *Universidad de Torino (Italia)* [Montuschi y Lang, 2001], [Montuschi y Lang, 1999], [Sanna et al, 1998].

Finalmente se señala el *laboratorio de arquitectura y sistemas de tiempo real* de la *Universidad de Massachusetts (Estados Unidos)* por su investigación en nuevas arquitecturas y algoritmos de alto rendimiento con características de tiempo real [Koren et al, 2003], [Bertoni et al, 2003], [Lakamraju et al, 2002].

Debido al carácter específico de esta materia cobran especial importancia los foros internacionales relacionados a los que acuden expertos en la disciplina así como destacados miembros de los grupos de investigación anteriores para compartir ideas, intercambiar opiniones y críticas acerca de sus investigaciones. Por su relación directa con este tema destaca el *Symposium on Computer Arithmetic* (Arith) de periodicidad bianual y la *Conference on Real Numbers and Computers* (RNC). Asimismo, se celebran otros eventos con sesiones dedicadas a arquitectura y aritmética de computadores entre los que se mencionan los siguientes: *Symposium on Digital Signal Design* (DSD), *International Conference on Electronics Circuits and Systems* (ICECS), *International Conference on Computer Design* (ICCD), *Conference on Application-specific Systems, Architectures and Processors* (ASAP), *Conference on Very Large Scale Integration* (VLSI SoC), *Symposium on High Performance Computer Architecture* (HPCA) y, *Conference on Design of Circuits and Integrated Systems* (DCIS).

Con respecto a las revistas de divulgación científica relacionadas con la materia se destacan por su alta relevancia e impacto para la comunidad

11,0010010000111111011010100010001000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010100100000010010011100000100010001001100111110011000111010 0

investigadora el *Journal of the ACM* [JACM, ISSN: 0004-5411] y el *IEEE Transactions on Computers* [IEEETC, ISSN: 0018-9340]. Existen otras publicaciones de contenido más específico entre las que se señalan las siguientes: IEEE *Transactions on Very Large Scale Integration Systems* [IEEEVLSI, ISSN: 1063-8210], *Journal of Systems Architecture* [JSC, ISSN: 1383-7621], *Journal of Computer and System Sciences* [JCSS, ISSN: 0022-0000] y, *Journal of Circuit Systems and Computers* [JCSC, ISSN: 0218-1266].

11,0010010000111111011010101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010100

Conclusiones

La revisión de las diferentes metodologías así como de las estrategias para hacer frente a las nuevas necesidades computacionales pone de manifiesto que las propuestas *a medida* se abren camino como método válido y ampliamente aceptado de solución. Las técnicas convencionales de representación numérica y formatos estándar reflejan la dificultad de representar y, por tanto de operar, números de infinitas cifras fraccionarias. De las investigaciones realizadas se concluye que mejoras en la capacidad de expresión de nuevos métodos y formatos chocan frecuentemente con aumentos considerables en la complejidad de las operaciones derivadas de su utilización, al margen de que para muchos problemas y aplicaciones, resulta absolutamente necesario disponer de una cantidad numérica que represente el valor del resultado.

Las propuestas software de representación y operación, a pesar de que constituyen soluciones versátiles compatibles con la mayoría de sistemas para la construcción de aplicaciones especializadas, no dan la talla en cuanto a las exigencias de rendimiento necesarias en ciertas aplicaciones. Con respecto a las soluciones a bajo nivel, las más recientes propuestas basadas en la aritmética de intervalos y los métodos que manejan una cantidad variable de cifras significativas de los números ofrecen alternativas de interés que mejoran la precisión de los resultados. Sin embargo, no son capaces de capaces de proveer un valor exacto sino, tan sólo, una aproximación para la que, en muchas ocasiones, se carece incluso de una medida de su calidad.

En definitiva, la exigencia para muchos problemas de disponer de una gran precisión en sus cálculos a un coste computacional aceptable pone de manifiesto la necesidad de realizar una expresión apropiada de los operandos y desarrollar la algoritmia a bajo nivel correspondiente. Por todo lo anterior, la concepción de un método de codificación exacta de los operandos basado en su representación numérica constituye un avance respecto a los métodos actuales ya que, desde el primer momento, se conoce el valor de los datos y los resultados parciales que se van generando, permitiendo concebir políticas de ajuste o aproximación eficaces en caso de que los requerimientos del problema lo permitan. El camino hacia estos objetivos pasa por establecer un

11,0010010000111111011010100010000100001011010001100001000110101100010011000110010100010111000000011011100000111001101000100101001000000100100111000001000100010100110011111001100011101001

marco formal adecuado en el que se encuadre la definición del problema y la especificación de la solución, cuestiones éstas que se abordarán en el siguiente apartado.

I - 4

Formulación del problema y propuesta de solución

La definición del problema en términos formales elimina ambigüedad y establece el soporte expresivo necesario para presentar las posibles soluciones. El estudio se centra en los aspectos concretos del procesador aritmético. En este sentido, tal y como se ha señalado en los objetivos, se trata de avanzar en el diseño de unidades aritméticas con propiedades específicas de representación y operatoria exacta con control sobre la precisión de números racionales.

Definición del problema

Como paso previo a la formalización de la unidad aritmética, se definen los componentes necesarios que describen el modelo de cálculo así como los elementos que influyen en el grado de precisión de los resultados obtenidos ya sea fruto de la propia codificación realizada o por el funcionamiento de los operadores empleados.

11,0010010000111111010101000100001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

Sea f una función matemática genérica. Se denomina *función instrumentación* Γ *de* f a cualquier función computable cuyo resultado se aproxima a f según una implementación particular:

$$\text{codominio}(\Gamma_f(\overline{X})) \subseteq \text{codominio}(f(\overline{X})) \quad [1.1]$$

tal que,

$$\forall\ \overline{X} \in \text{dominio}(\Gamma_f),\ |\Gamma_f(\overline{X}) - f(\overline{X})| \leq \varepsilon \quad [1.2]$$

donde,

$\overline{X}$:Denota al conjunto de parámetros sobre los que opera la función.

$\varepsilon : \varepsilon \in \mathbb{R}^+ \cup \{0\}$. Indica el grado de aproximación de f por Γ.

El cometido general de una unidad aritmética es el de procesar funciones matemáticas. Una arquitectura A se caracteriza tanto por el conjunto de funciones que proporciona como por la forma en la que las instrumenta. La conjunción de estos dos aspectos constituye un procesador.

Dado el siguiente conjunto de funciones:

$$\Phi = \{f_1, f_2, \ldots, f_n\} \quad [1.3]$$

Una arquitectura A_Φ que proporcione dichas funcionalidades estará formada por:

$$A_\Phi = \{\Gamma_{f1}, \Gamma_{f2}, \ldots, \Gamma_{fn}\} \quad [1.4]$$

Es decir, A_Φ contendrá las instrumentaciones concretas de las funciones de Φ, donde cada Γ_{fi} produce una aproximación a su f_i correspondiente. Conforme lo mencionado anteriormente, cada f_i supone el objetivo a alcanzar por el procesador, mientras que cada Γ_{fi} corresponde con la función que finalmente se proporciona. La instrumentación de una función f no tiene por qué ser única, pueden existir varias instrumentaciones de una misma función que representen distintas aproximaciones a f, por ejemplo con un valor diferente de **ε.**

11,0010010000111111010101000100001011010001100001000110101100010011000110010100010111000000011011100000111001101000100101001000000100100111000001000100010100110011111001100011101 00

A lo largo de esta memoria se maneja el término instrumentación en modo general, haciendo entender que una función proporciona un resultado dependiente de su propia implementación. De este modo, se da el caso de la existencia de múltiples arquitecturas donde cada una de ellas contiene una serie de instrumentaciones concretas para un conjunto de funciones Φ. Asimismo, es posible que una misma arquitectura contenga varias instrumentaciones para una misma función f, prestando varios grados de aproximación a la función, por ejemplo las distintas implementaciones de la suma aritmética para formatos diferentes.

A continuación se define el significado de *precisión variable* y su efecto en los conceptos definidos hasta ahora.

Se denomina *función instrumentación* Γ *en precisión variable (*VP —*Variable Precision) de* f a cualquier función computable cuyo resultado tiende a f según el valor de ciertos parámetros $\bar{d}$ y de acuerdo con una instrumentación particular:

$$\Gamma_f^{VP}(\bar{x}, \bar{d}) \qquad [1.5]$$

tal que,

$$\forall\ \bar{x} \in \text{dominio}(\Gamma_f),\ |\Gamma_f^{VP}(\bar{x}, \bar{d}) - f(\bar{x})| \leq \varepsilon \qquad [1.6]$$

donde $\bar{x}$ y ε tienen el mismo significado que en la expresión [1.2].

La función Γ debe ser lo suficientemente general como para admitir diversas aproximaciones de f. No obstante, para que sea considerada función VP, el grado de similitud entre los resultados proporcionados por ambas funciones debe estar ligado al valor de $\bar{d}$.

Es preciso resaltar que el grado de aproximación de f por Γ_f^{VP} está relacionado con la función f, la propia instrumentación Γ y el valor $\bar{d}$. Para determinadas instrumentaciones Γ_f^{VP} sólo unos valores de $\bar{d}$ realizan una aproximación, obteniendo el valor exacto de f para el resto. Otras funciones sin embargo, no producen nunca el resultado sin error para ninguna instancia de $\bar{d}$.

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100101000101110000000110111000001110011010001001010010000001001001110000010001000101001100111111001100011101100

Se dice que una función Γ_f^{VP} *tiene una evaluación exacta de* f si existe algún valor de $\bar{d}$, tal que $\varepsilon = 0$. Según la ecuación [1.6] se deduce que:

$$\forall\ \bar{x} \in \text{dominio}(\Gamma_f),\ \exists\ \bar{d} /\ \Gamma_f^{VP}(\bar{x},\ \bar{d}) = f(\bar{x}) \qquad [1.7]$$

Por el contrario, se dice que una función Γ_f^{VP} *no tiene una evaluación exacta de* f si para ningún valor de $\bar{d}$ se alcanza el valor de f.

$$\forall\ \bar{x} \in \text{dominio}(\Gamma_f),\ \neg\exists\ \bar{d} /\ \Gamma_f^{VP}(\bar{x},\ \bar{d}) = f(\bar{x}) \qquad [1.8]$$

Se dice que una *arquitectura es de precisión variable* A^{VP}, si al menos una de las funciones que instrumenta es de precisión variable:

$$\exists\ \Gamma_{fi} \in A_\Phi.\ \Gamma_{f_i}^{VP} \Rightarrow A_\Phi^{VP} \qquad [1.9]$$

Para esta arquitectura, existirá al menos un conjunto de parámetros $\bar{d}$ tal que influya en el resultado de ciertas operaciones y varíe el grado de aproximación a f. De esta forma, es posible en esta arquitectura actuar sobre el valor de $\bar{d}$ para obtener diversa precisión en los resultados.

Asimismo, si la arquitectura de precisión variable A^{VP}, contiene alguna función con evaluación exacta, se dice que dicha arquitectura instrumenta de manera efectiva esa función. Es decir, cumple el objetivo de proporcionar su resultado sin error para cualquier instancia de sus operandos.

Por razones de escalabilidad de los sistemas, el diseño de una arquitectura puede organizarse en niveles. A partir de un núcleo elemental P, el de las primitivas del procesador, se construyen las demás funciones, es decir, las derivadas.

Dado el conjunto de funciones [1.3]:

$$\Phi = \{f_1, f_2, \ldots, f_n\}$$

se dice que la arquitectura

$$A = \{\Gamma_{f1}, \Gamma_{f2}, \ldots, \Gamma_{fs}\}_P \qquad [1.10]$$

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001110011010001001010010000001001001110000010001000101001100111111001100011110100

constituye un *conjunto de funciones primitivas*, si y sólo si la instrumentación de cada una de dichas funciones se realiza de forma independiente, es decir:

$$A = \{\Gamma_{f1}, \Gamma_{f2}, \ldots, \Gamma_{fs}\}_P \Leftrightarrow \neg\exists\, \Gamma_{fi} \in A.\ \Gamma_{fi} = \bigotimes_i (A) \qquad [1.11]$$

donde la expresión $\bigotimes_i (A)$ denota cualquier composición de las funciones de A, sin contar la función i.

Si alguna de las funciones primitivas es de precisión variable, entonces la arquitectura formada por esas primitivas será también de precisión variable.

$$\exists\, \Gamma_{fi} \in A.\ \Gamma_{f_i}^{VP} \Rightarrow A^{VP} \qquad [1.12]$$

A partir de estas funciones elementales, el conjunto Φ de funciones podrá ser instrumentado por una arquitectura que contenga, además, una serie de funciones derivadas a partir de las primitivas.

$$A_\Phi = \{\Gamma_{f1}, \Gamma_{f2}, \ldots, \Gamma_{fs}\}_P \cup \{\Gamma_{fs+1}, \ldots, \Gamma_{fn}\} \qquad [1.13]$$

donde,

$$\forall\, \Gamma_{fi} \in \{\Gamma_{fs+1}, \ldots, \Gamma_{fn}\}.\ \Gamma_{fi} = \bigotimes_i (A_\Phi) \qquad [1.14]$$

Si el conjunto de funciones primitivas es de precisión variable, la arquitectura completa también lo será:

$$A^{VP} \Rightarrow A_\Phi{}^{VP} \qquad [1.15]$$

Por otra parte, si la propia instrumentación de una función derivada es de precisión variable, la arquitectura también será de precisión variable, independientemente de que el conjunto de primitivas lo sea.

Desde el punto de vista de la arquitectura de los computadores se considera que el procesador de una arquitectura dada está dotado de una estructura interna consistente en sus módulos estructurales y en la lógica de relación entre ellos.

11,00100100001111110110101010001000100010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101100

$$\Pi = \Pi\,(M, \Lambda) \qquad [1.16]$$

donde,

M: Conjunto de módulos estructurales.

Λ: Lógica de relación entre los módulos estructurales.

Los criterios de diseño de M y Λ deberán ser coherentes para preservar sus características y mantener sus propiedades en su funcionamiento conjunto, de modo que la característica VP de la *arquitectura aritmética* sea mantenida y apoyada por todos los componentes para que sea efectiva. De esta forma, para que un procesador sea VP, Π^{VP}, deberá contener un núcleo funcional cuya implementación sea VP y mantener esta propiedad en toda su arquitectura. Teniendo en cuenta todo lo anterior, el enunciado del problema es el siguiente:

Definir un procesador de precisión variable, Π^{VP}.

En particular, la definición de un procesador aritmético para operandos racionales en precisión variable, Π^{VP}, con operaciones que admitan una evaluación sin error.

Las funcionalidades que proporcionará dicho procesador serán:

$$\Phi_Q = \{\text{identidad}, \text{suma}, \text{producto}\} \qquad [1.17]$$

Las operaciones en precisión variable que instrumenta dicho procesador son las siguientes:

$$P^{VP} = \{\Gamma^{VP}_{\text{identidad}}\ ,\ \Gamma^{VP}_{\text{suma}}\ ,\ \Gamma^{VP}_{\text{producto}}\}_P \qquad [1.18]$$

De tal forma que:

$$\Pi^{VP}_{\Phi_Q} = P^{VP} \qquad [1.19]$$

El problema se concreta en concebir métodos VP para evaluar cada una de las operaciones aritméticas y en incorporar los elementos necesarios para una correcta instrumentación que permitan su evaluación exacta.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

El procesador debe plasmar una arquitectura que instrumente de manera totalmente efectiva el conjunto de funciones sobre números racionales. Se propone implementar la función *identidad* para expresar la capacidad misma del procesador de representar y operar con números racionales. En la práctica, esta función corresponde con el formato de codificación numérico empleado, que representa sin error cualquier número racional.

Propuesta de solución

En términos generales, la solución pasa por desarrollar arquitecturas específicas que doten al computador de características que aporten propiedades de computación numérica exacta sobre los números racionales sin perjuicio de tolerar distintos niveles de aproximación al resultado. Cabe incorporar un conjunto de instrucciones con atributos de VP apoyado por otros elementos como la codificación y almacenamiento conveniente de los datos que favorezcan el funcionamiento correcto.

El punto de partida es la especificación de un formato de representación capaz de expresar cualquier elemento del conjunto de los números racionales. De esta forma se consigue su formulación exacta y única, así como su posterior manipulación. La idea parte de la combinación del método presentado por Hehner y Horspool [Hehner y Horspool, 1979] y los formatos de coma flotante. Su principal característica se encuentra en codificar cualquier valor perteneciente al conjunto de los números racionales mediante una notación fraccionaria normalizada que represente de forma separada las cifras fijas de las periódicas. Este esquema aporta una representación uniforme de los números racionales con una cantidad variable de cifras en cualquiera de sus campos pero que en todo caso será finita. Este hecho permite una representación compacta de los valores racionales periódicos sin recurrir a la notación simbólica de fracción. Adicionalmente, el esquema permite conocer directamente el valor del número y ofrece una cantidad de cifras fraccionaria ilimitada. El formato alcanza a la representación de valores extremos mediante un campo para el exponente que contiene el orden de magnitud de los números. Por otra parte, la codificación exacta de valores racionales abre otras vías de operación, dando lugar a enfoques que resuelvan sin error un conjunto de problemas del cálculo.

El almacenamiento de los datos expresados en este esquema condiciona su procesamiento. Para aprovechar todo su potencial resulta conveniente disponer de una organización flexible de memoria que permita manejar estructuras variables. Su instrumentación a bajo nivel en un contexto de registros de longitud fija sugiere la búsqueda de estrategias que aporten la flexibilidad necesaria para mantener las

11,0010010000111111011010100010001000010110100011000010001101001100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

propiedades VP de las instrucciones desde el nivel más bajo de la arquitectura. Se propone la utilización de punteros que marquen la separación entre los campos del número, la incorporación de varios juegos de registros con distintas longitudes o la capacidad de reconfiguración de algunas de las características del espacio de almacenamiento.

El diseño de las operaciones se fundamenta en el uso de esquemas iterativos y de resultados precalculados. La composición de estas dos técnicas va a permitir la instrumentación VP de funciones que, junto con los elementos anteriores, constituirán el procesador racional flexible. En este sentido, la búsqueda de estrategias para mejorar el rendimiento de las operaciones se concreta en el aumento de la granularidad de los operandos elementales utilizando en su construcción memorias que contienen los resultados precalculados de su ejecución. Aunque estas técnicas operan al nivel de bloque, el almacenamiento de los datos precalculados impone severas restricciones a su tamaño, lo cual sugiere a su vez la división de las partes y operar con los bloques individualmente, componiendo el resultado por concatenación.

Mediante la integración de todos esos aspectos, la propuesta de solución constituye un modelo que abarca tanto las representaciones de los datos como los métodos de operación en sí, dando lugar a un esquema de cálculo exacto adaptable a las necesidades de cada problema. Algunas de las propiedades que posee este modelo son: capacidad de representación exacta de los números racionales, lo cual facilita su manipulación y procesamiento sin error; predecibilidad en el tiempo de cálculo, debido al carácter finito de la codificación y; sistematicidad del esquema algorítmico debido al carácter repetitivo de los algoritmos, lo cual propicia un alto grado de paralelismo.

Finalmente, la investigación se materializa en la propuesta de un procesador aritmético flexible que utiliza métodos que sistematizan la implementación de funciones de bajo nivel y emplean un formato flexible de representación de valores numéricos. Se construye un repertorio de instrucciones aritméticas cuyo procesamiento evoluciona iterativamente hacia su valor exacto. Se diseñan las primitivas suma y producto orientadas hacia el uso de memorias con resultados

11,00100100001111110110101001000100001011010001100001000110100110001001100011001101000101110000000110111000001110011010001001010010000001001001110000010001000101001100111111001100011101000

precalculados y la primitiva identidad que consiste en la representación de un número en el formato de codificación propuesto. Se deja para trabajos futuros la instrumentación de otras primitivas.

En la siguiente figura se muestra un esquema que resume el diseño de la unidad. Existe un conjunto de parámetros $\bar{d}$ que condiciona el grado de precisión de los resultados y gradúa la aproximación realizada en función de los requerimientos del problema.

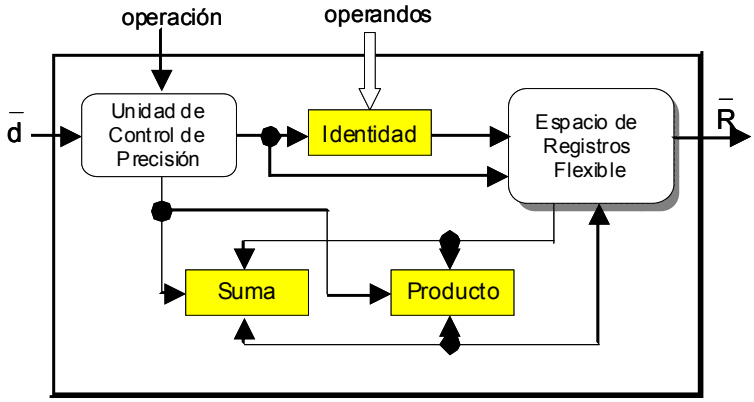


Figura 1-1: Esquema general del Procesador Racional Flexible

El sistema dispone de una unidad de control de precisión que, atendiendo a la operación solicitada y al grado de aproximación requerido, establece las características del conjunto de registros que intervienen en el procesamiento y ajusta los operadores a su precisión.

Capítulo II

Identidad en Precisión Variable

1. Representación de los números racionales
2. Instrumentación de la función identidad

II - 1

Representación de los números racionales

En este apartado se introduce un sistema de representación numérica para un computador capaz de representar valores racionales sin error. Dejando a un lado la notación numérica simbólica, nos referiremos en lo sucesivo a la representación fraccionaria posicional de los números como una forma de expresión directa de su valor.

Conceptos sobre representación numérica

Como paso previo a la definición del modelo de representación que se propone se establecen los principios de la expresión fraccionaria de los números racionales. A continuación se formalizan las relaciones existentes entre los números que intervienen en el problema y los datos que se manejan para su tratamiento, entendiendo como tales, los valores numéricos representados y manipulados por la máquina. En este aspecto se analizan los formatos de representación posicional de números racionales en coma fija y en coma flotante como punto de

11,0010010000111111010101000100010000101101000110000100011010011000100111000110011001010001011100000001110011010001001010010000001001001110000010001000101001100111111001100011110100

partida en la formalización del esquema de representación numérica propuesto.

Representación fraccionaria de los números racionales

Los números reales forman parte integrante de muchos problemas ya sea por los tipos de datos que manejan o por la naturaleza de las operaciones que intervienen. La manipulación y el almacenamiento de estos valores son tareas imprescindibles para el procesamiento de dichos problemas mediante un computador.

La representación fraccionaria posicional permite conocer directamente el valor del número, realizar comparaciones y operar de forma directa. Estas características son deseables en un formato de codificación numérica, sin embargo, no todos los números reales se pueden expresar de esta forma. Los números irracionales disponen de una cantidad infinita de cifras fraccionarias significativas y, por tanto, no es posible su expresión en notación posicional. Además, sólo un subconjunto numerable de números irracionales es computable, es decir, existe para ellos un procedimiento que es capaz de generar cada cifra del valor real [Gianantonio, 1993], [Turing, 1937]. Para estos números, la forma de expresar su valor exacto es procedimiental, donde se describe el método de obtención del valor en lugar del valor en sí, por ejemplo: π, $\sqrt{3}$, sen 2, $\lg_3 4$, e^5, ... Cabe recalcar que, si bien es posible obtener cualquier cantidad de sus cifras significativas en un tiempo finito, siempre será una aproximación respecto a su valor exacto. Por otra parte, es necesario considerar también las imprecisiones derivadas de las limitaciones físicas de la máquina para contener dichas cifras.

Los números racionales sí permiten su representación posicional exacta. Cualquier número racional expresado en notación fraccionaria puede representarse mediante una *cantidad finita de cifras significativas*.

La representación posicional de un número racional es la siguiente:

$$x \in \mathbb{Q} \Leftrightarrow \exists\, L(m_f), L(m_p)>0 \in \mathbb{N} \,/$$
$$/\; x = \alpha_{L(mf)}\alpha_{L(mf)-1}\ldots\, \alpha_i\,,\, \alpha_{i-1}\,\ldots\,\alpha_1\,\alpha_0\gamma_{L(mp)-1}\,\ldots\,\gamma_1\gamma_0\,\gamma_{L(mp)-1}\,\ldots\,\gamma_1\gamma_0\,\ldots \qquad [2.1]$$

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

donde para una base de representación B: $\alpha_i \in \{0..B-1\} \wedge \gamma_i \in \{0..B-1\}$

Esta representación está formada por la concatenación de una secuencia de cifras alrededor de la coma fraccionaria, llamada parte fija, con una serie periódica de cifras que forman un ciclo. A la secuencia de dígitos de menor longitud que se repite se denomina el periodo del número. En la notación que se emplea estas cifras se representan tan sólo una vez y se señalan trazando sobre ellas un arco de circunferencia.

$$x = \alpha_{L(mf)}\alpha_{L(mf)-1}\ldots\alpha_i\ ,\ \alpha_{i-1}\ldots\alpha_1\alpha_0\ \overbrace{\gamma_{L(mp)-1}\ldots\gamma_1\gamma_0} \qquad [2.2]$$

La expresión anterior muestra el valor exacto de un número racional con una cantidad de cifras finita. Existe una equivalencia entre la representación fraccionaria posicional y la notación simbólica de fracción para expresar los números racionales.

$$\forall\, x \in \mathbb{Q},\ \exists\, a, b \in \mathbb{Z} / x = \frac{a}{b} \wedge b \neq 0 \qquad [2.3]$$

Aunque las expresiones [2.2] y [2.3] representan al mismo número de forma exacta, la notación simbólica de fracción no es única ya que existen infinitas fracciones para un mismo número.

Como casos particulares de la configuración del periodo en la representación fraccionaria, cabe mencionar los siguientes:

- *Todas las cifras son cero*: con la información correspondiente a las cifras no periódicas es suficiente para conocer con exactitud el valor del número.

$$\alpha_{L(mf)}\alpha_{L(mf)-1}\ldots\alpha_i\ ,\ \alpha_{i-1}\ldots\alpha_1\alpha_0\overbrace{\gamma_{L(mp)-1}\ldots\gamma_1\gamma_0} =$$
$$= \alpha_{L(mf)}\alpha_{L(mf)-1}\ldots\alpha_i\ ,\ \alpha_{i-1}\ldots\alpha_1\alpha_0 \Leftrightarrow \forall\, i \in \{0..L(m_p)-1\}\ \gamma_i = 0 \qquad [2.4]$$

- *Todas las cifras son iguales a la base de la numeración menos uno* (en base 10 iguales a 9, en base 2 iguales a 1): en este caso el número se expresa sumando uno a la cifra fija menos significativa y eliminando las cifras periódicas.

$$\alpha_{L(mf)}\alpha_{L(mf)-1}\ldots\alpha_i\ ,\ \alpha_{i-1}\ldots\alpha_1\alpha_0\overbrace{\gamma_{L(mp)-1}\ldots\gamma_1\gamma_0} =$$
$$= \alpha_{L(mf)}\alpha_{L(mf)-1}\ldots\alpha_i\ ,\ \alpha_{i-1}\ldots\alpha_1(\alpha_0+1) \Leftrightarrow \forall\, i \in \{0..L(m_p)-1\}\ \gamma_i = (B-1) \qquad [2.5]$$

11,00100100001111110110101001000100001011010001100001000110101001100011001100010100010110000000111011000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

La cantidad de cifras, $L(m_p)$, que forman el periodo de un número racional está en estrecha relación con el denominador de la fracción irreducible que lo representa según la siguiente expresión.

$$\forall\, x = \alpha_{L(mf)}\alpha_{L(mf)-1}\ldots\alpha_i\ ,\ \alpha_{i-1}\ldots\ \alpha_1\alpha_0\,\overbrace{\gamma_{L(mp)-1}\ldots\gamma_1\gamma_0} \in \mathbb{Q}\ ,\ \exists\, a,b \in \mathbb{Z}\ /$$

$$/\ x = \frac{a}{b} \wedge b \neq 0 \wedge \text{mcd}(a, b) = 1 \Rightarrow L(m_p) = \Im\ (b) < b \qquad [2.6]$$

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100101000100101001000000100100111000001000100010100110011111001100011110100

La cantidad de cifras fraccionarias periódicas es siempre menor que b. En concreto, la definición de la función $\Im$ que determina $L(m_p)$ consiste en las siguientes relaciones de congruencia [Belski y Kaluzhnin, 1980], [Guelfond, 1979]:

Si $mcd(B, b) = 1 \Rightarrow B^{L(mp)} \equiv 1 \bmod (b)$

Si no

$$\exists\, b', m \in \mathbb{Z} \,/\, b = m \cdot b' \wedge mcd(B, b') = 1 \wedge m = \prod_{i=1}^{p} f_i^{\,j}$$

siendo f_1, ..., f_p la lista de factores primos de la base de representación B y j un natural, de tal forma que:

$$\frac{a}{b} = \frac{1}{m} \cdot \frac{a}{b'} \wedge mcd(B, b') = 1 \wedge B^{L(mp)} \equiv 1 \bmod (b')$$

Por las reglas de la aritmética modular, $\frac{1}{m}$, al estar compuesta por factores primos de la base, genera un valor fraccionario de periodo cero.

donde,

$\equiv$: Relación de congruencia.

mod : Operador módulo.

Existen infinitos valores de que cumplen la ecuación anterior, ya que todo múltiplo del valor $L(m_p)$ satisface la ecuación:

$$B^{L(mp)} \equiv 1 \bmod (b) \Rightarrow \forall\, j \in \mathbb{N},\, B^{j \cdot L(mp)} \equiv 1 \bmod (b) \qquad [2.7]$$

Este hecho justifica que cualquier grupo de cifras que incluya al periodo una cantidad entera de de veces y cuya longitud sea múltiplo de $L(m_p)$ constituye en sí mismo un periodo. Asimismo, todos los números racionales son periódicos, si bien éste no se representa cuando todas sus cifras son cero.

La cantidad de números periódicos con periodo distinto de cero está relacionada con la base de la numeración. Existen números racionales

11,0010010000111111011010100010000100001011010001100001000110100110001001100011001100101000100101001000000100100111000001000100010100110011111100110001110100

decimales de periodo cero que producen un periodo binario distinto de cero, mientras que al ser 2 un factor de 10, todo número binario de periodo cero produce un número decimal también de periodo cero. Esta observación es de gran relevancia para la representación numérica y la aritmética de computadores, ya que en un dominio binario existen números fraccionarios de periodo cero en decimal que producirán errores en la representación binaria ante la imposibilidad de codificar infinitas cifras fraccionarias. De ahí la importancia de definir un modelo que permita representar y operar con estos números de manera exacta.

Sin pérdida de generalidad se considera a lo largo de este trabajo números racionales no negativos menores que uno. Es decir,

$$x \in [0, 1) \subset \mathbb{Q} \Leftrightarrow 0,\alpha_{L(mf)-1}\alpha_{L(mf)-2}\ldots\alpha_i\alpha_{i-1}\ldots\alpha_1\alpha_0\overbrace{\gamma_{L(mp)-1}\ldots\gamma_1\gamma_0} \qquad [2.8]$$

y donde $\alpha_{L(mf)} = 0$.

A la secuencia de cifras $\alpha_{L(mf)-1}\ldots\alpha_1\alpha_0$ de longitud de $L(m_f)$ es a lo que se denomina *mantisa fija*:

$$m_f = \alpha_{L(mf)-1}\ldots\alpha_1\alpha_0 \qquad [2.9]$$

A la secuencia de cifras $\gamma_{L(mp)-1}\ldots\gamma_1\gamma_0$ de longitud $L(m_p)$ se denomina *mantisa periódica*:

$$m_p = \gamma_{L(mp)-1}\ldots\gamma_1\gamma_0 \qquad [2.10]$$

La mantisa periódica sólo tiene sentido para posiciones a la derecha de la coma fraccionaria. La cadena de dígitos que la forman es suficiente para conocer toda la secuencia infinita de cifras fraccionarias que contiene la representación posicional de los números racionales a partir de la mantisa fija.

$$\widehat{m}_p = \gamma_{L(mp)-1}\ldots\gamma_1\gamma_0\,\gamma_{L(mp)-1}\ldots\gamma_1\gamma_0 \qquad [2.11]$$

El número racional queda formado por la concatenación de las cifras de la mantisa fija con las de la mantisa periódica a la derecha de la coma fraccionaria. Para una mayor claridad en la argumentación, se utilizará en lo sucesivo la siguiente expresión para referirse a los números racionales:

11,0010010000111111011010100010001000010110100011000010001101001100011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

$$x \in [0, 1) \subset \mathbb{Q},\ x = 0{,}m_f\widehat{m}_p \qquad [2.12]$$

En función de la naturaleza de su mantisa se realiza una clasificación de éstos números racionales basándose en las siguientes definiciones.

11,0010010000111111010101000100001000010110100011000010001101001100011000110011000101000101110000000111001101000100101001000000100100111000001000100010010011001111110011000111010100

Definición 1

Se denomina *número periódico* a todo número racional que contenga mantisa periódica distinta de cero:

$$A \in [0, 1) \subset \mathbb{Q},\ A = 0,m_{fA}\widehat{m}_{pA} \wedge m_{fA} = \alpha_{L(mfA)-1}\ldots\alpha_0 \wedge m_{pA} = \gamma_{L(mpA)-1}\ldots\gamma_0\ /$$

$$/\ L(m_{pA}) > 0 \wedge \exists\ i \in \{0..L(m_{pA}) - 1\},\ \gamma_i \neq 0 \qquad [2.13]$$

Definición 2

Se denomina *número no periódico* a todo número racional que contenga mantisa periódica igual a cero:

$$A \in [0, 1) \subset \mathbb{Q},\ A = 0,m_{fA}\widehat{m}_{pA} \wedge m_{fA} = \alpha_{L(mfA)-1}\ldots\alpha_0 \wedge m_{pA} = \gamma_{L(mpA)-1}\ldots\gamma_0\ /$$

$$/\ L(m_{pA}) > 0 \wedge \forall\ i \in \{0..L(m_{pA}) - 1\},\ \gamma_i = 0 \qquad [2.14]$$

Definición 3

Se denomina *número periódico puro* a todo número periódico que contenga sólo mantisa periódica distinta de cero:

$$A \in [0,1) \subset \mathbb{Q},\ A = 0,m_{fA}\widehat{m}_{pA} \wedge m_{fA} = \alpha_{L(mfA)-1}\ldots\alpha_0 \wedge m_{pA} = \gamma_{L(mpA)-1}\ldots\gamma_0\ /$$

$$/\ L(m_{pA}) > 0 \wedge L(m_{fA}) = 0 \wedge \exists\ i \in \{0..L(m_{pA}) - 1\},\ \gamma_i \neq 0 \qquad [2.15]$$

Definición 4

Se denomina *número periódico mixto* a todo número racional que contenga mantisa periódica no nula y mantisa fija:

$$A \in [0,1) \subset \mathbb{Q},\ A = 0,m_{fA}\widehat{m}_{pA} \wedge m_{fA} = \alpha_{L(mfA)-1}\ldots\alpha_0 \wedge m_{pA} = \gamma_{L(mpA)-1}\ldots\gamma_0\ /$$

$$/\ L(m_{pA}) > 0 \wedge L(m_{fA}) > 0 \wedge \exists\ i \in \{0..L(m_{pA}) - 1\},\ \gamma_i \neq 0 \qquad [2.16]$$

Como aportación en este trabajo se presenta la siguiente definición que complementa a las anteriores. Los números de este tipo serán utilizados

en el desarrollo de la algoritmia de los operadores matemáticos que se describen en capítulos posteriores.

Definición 5

Se denomina *número periódico unidad de grado s* a todo número periódico puro cuya mantisa periódica esté formada por una secuencia de $s-1$ ceros seguidos de un uno en la posición menos significativa.

$$A \in [0,1) \subset \mathbb{Q},\ A = 0,m_{fA}\widehat{m}_{pA} \wedge m_{fA} = \alpha_{L(mfA)-1}\ldots\alpha_0 \wedge m_{pA} = \gamma_{s-1}\ldots\gamma_0\ /$$

$$/\ L(m_{pA}) > 0 \wedge L(m_{fA}) = 0 \wedge \forall\ i \in \{1..s-1\}\ \gamma_i = 0 \wedge \gamma_0 = 1 \qquad [2.17]$$

Los métodos para transformar un número racional desde una notación simbólica de fracción, $\frac{a}{b}$, a una notación fraccionaria posicional y viceversa se indican a continuación:

- La conversión de un número racional formulado en notación simbólica de fracción a expresión fraccionaria se realiza mediante la división indicada expresamente en la fracción. En el cociente se obtiene la expresión fraccionaria posicional y los restos de esta división entre b son valores β acotados: $0 \le \beta < b$.

- La conversión de número racional expresado en notación posicional a notación simbólica de fracción se realiza mediante la siguiente expresión,

$$0,m_f\widehat{m}_p = \frac{m_f\, m_p - m_f}{\underbrace{(B-1)\cdots(B-1)}_{L(m_p)}\ \underbrace{0\ \cdots\ 0}_{L(m_f)}} \qquad [2.18]$$

  donde el numerador de la fracción se constituye por la resta entre el entero formado por la concatenación de las cifras de la mantisa fija y periódica menos las cifras de la mantisa fija y, el denominador consiste en el entero formado por la concatenación de $L(m_p)$ cifras iguales a la base de la representación menos uno con $L(m_f)$ ceros en la parte menos significativa.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010010000001001001110000010001000101001100111111001100011101100

A modo de ejemplo se presentan las siguientes correspondencias entre esquemas de representación de los números racionales:

Base 10:

$$0,1208\widehat{367} = \frac{1208367 - 1208}{9990000} = \frac{1207159}{9990000}$$

donde, $L(m_f) = 4$ y $L(m_p) = 3$

Base 2:

$$0,10\widehat{110} = \frac{10110 - 101}{11000} = \frac{10001}{11000}$$

donde, $L(m_f) = 3$ y $L(m_p) = 2$

Representación en coma fija

La representación de un número en coma fija consiste en una secuencia de cifras de una longitud de palabra establecida, $L(n)$, correspondientes a su representación aritmética posicional desde un orden de magnitud inicial (m_i) hasta un orden de magnitud final (m_f), tal que $m_i \geq m_f$. En general, se añade una posición más para expresar el signo del número mediante un convenio discreto. De este modo: $L(n) = m_i - m_f + 2$.

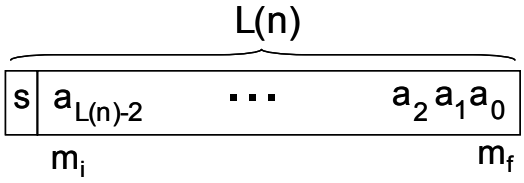


Figura 2-1: Esquema general de la representación en coma fija

El valor del número codificado en este formato se obtiene con el mismo procedimiento que en un sistema numérico posicional teniendo en cuenta únicamente los coeficientes asignados a tales potencias de la base.

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111101 00

$$A_{cfija} = (-1)^s \cdot \sum_{i=0}^{L(n)-2} a_i \cdot B^{mf+i} \qquad [2.19]$$

La denominación de coma fija proviene del hecho de que, del conjunto de cifras de la representación del número, se destina una de ellas a la codificación del signo, una parte fija de cifras, $L(m_e)$, a la representación de su parte entera y el resto, $L(m_f)$, a la representación de su parte fraccionaria. El orden de magnitud correspondiente a las unidades ocupa una posición intermedia, $m_i \geq 0 \geq m_f$, como describe la figura 2-2 [Cilio y Corporal, 1999], [Kum et al, 1997].

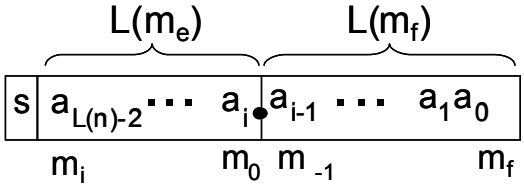


Figura 2-2: Estructura de la representación en coma fija

En lo que respecta a las características de representación numérica se destacan las siguientes:

1) El intervalo de los valores representables (R) viene determinado por la cantidad de cifras destinadas a la representación,

   $$R = [-(B^{L(me)} - B^{-L(mf)}), B^{L(me)} - B^{-L(mf)}] \qquad [2.20]$$

   donde B es la base de representación.

2) El intervalo de representación es discreto con una cuantización relacionada con la cantidad de cifras fraccionarias:

   $$q = B^{-L(mf)+1} \qquad [2.21]$$

Este esquema sólo es capaz de representar de manera exacta números racionales que contienen una cantidad de cifras enteras menor o igual a $L(m_e)$ y una cantidad de cifras fraccionarias menor o igual a $L(m_f)$. Esta limitación inherente a la coma fija supone un gran inconveniente para el cálculo matemático ya que existen números que no son representables sin error al no caber en alguna de las partes de que se

11,00100100001111110110101010001000100001011010001100001000110101001100010011000110011001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

compone el formato y, aunque se dedique una cantidad de cifras finita arbitraria a la representación numérica, tampoco es posible la codificación sin error de valores racionales con una parte fraccionaria periódica que se repita indefinidamente. Estas circunstancias se soslayan en la práctica realizando una aproximación a la representación exacta más cercana, lo que provoca imprecisiones y errores en los cálculos resultantes.

Por sus características, la representación en coma fija constituye una correspondencia entre el conjunto $\mathbb{Q}$ y los números racionales representados sin error por el formato. El redondeo al valor representable exacto provoca que una vez codificado un número resulte imposible conocer su valor original.

En la siguiente tabla se muestran algunos ejemplos de la representación en coma fija de valores racionales sin signo para una cantidad de cifras determinada: $L(n) = 10$; $L(m_e) = 5$; $L(m_f) = 5$. El conjunto de valores pretende ser lo más heterogéneo posible y debe interpretarse sólo como muestra de las características del formato.

$x \in \mathbb{Q}$	$A_{cfija}(x)$	|Error| $\lvert A_{cfija}(x) - x \rvert$
$123 \cdot 10^{1234567890}$	No representable	--
$123 \cdot 10^{123585}$	No representable	--
$123 \cdot 10^{3585}$	No representable	--
$123 \cdot 10^{7}$	No representable	--
123456	No representable	--
123,456	00123,45600	0
12345,67890123	12345,67890	0,00000123
0,123456	00000,12346	0,000004
0,00000000000012	00000,00000	0,000000000000123
$123 \cdot 10^{-7}$	00000,00001	$23 \cdot 10^{-7}$
$123 \cdot 10^{-3585}$	00000,00000	$123 \cdot 10^{-3585}$
123,45454545…	00123,45455	0,0000045454545…

11,0010010000111111011010100010001000010110100011000010001101001100011001100110010100010111000000011011100000111001101000100101001000000100100111000000100010100110011111100110001110100

0,017171717…	00000,017172	0,0000028282828…

Tabla 2-1: Ejemplos de representación en coma fija

Como se observa en la tabla anterior, los números con más de $L(m_e)$ cifras enteras se salen del rango de representación y no se pueden codificar por el formato. Para otros valores con más cifras que las disponibles en el esquema se realiza un redondeo que provoca un error de representación de una importancia proporcional al tamaño del número.

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000011011100000111001101000100101001000000100100111000001000100010100110011111100110001110100

Representación en coma flotante

En este formato se expresan separadamente las cifras del número y su orden de magnitud. La longitud de palabra de la representación, L(n), se distribuye entre el signo, el exponente L(e) y la mantisa L(m).

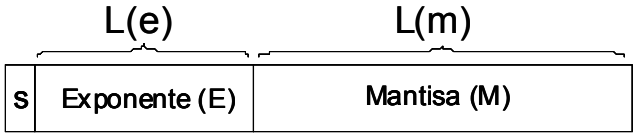


Figura 2-3: Esquema general de la representación en coma flotante

El valor del número se obtiene mediante la siguiente expresión,

$$A_{cflotante} = (-1)^s \cdot M \cdot B^E \qquad [2.22]$$

siendo B la base de la representación y, M y E el valor numérico de la mantisa y del exponente respectivamente.

La expresión separada del exponente permite la expresión de números de muy distinto valor. Esta característica proporciona una mayor flexibilidad y potencia expresiva a la vez que propicia su utilización en sistemas de propósito general [Patterson y Hennessy, 2002], [Sun Microsystems, 2000]. Su estandarización [IEEE, 1987], [IEEE, 1985] ha jugado un papel crucial en el desarrollo y extensión de este formato [Nielsen, 1997].

Se destacan las siguientes características:

1) La representación en coma flotante de un número no es única. Para evitar la múltiple codificación, la mantisa se expresa normalizada.

2) El intervalo de valores representables (R) depende de la cantidad de cifras que se destinan tanto a la mantisa como al exponente, alcanzando como se ha indicado, valores límite mucho mayores que en la anterior representación. Por ejemplo, el intervalo según la norma IEEE-754 tiene la siguiente expresión:

$$R = [-(2-2^{-L(m)})\cdot 2^{2^{L(e)-1}-1}, (2-2^{-L(m)})\cdot 2^{2^{L(e)-1}-1}] \qquad [2.23]$$

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

3) El intervalo de representación es discreto con una mayor separación entre los números representados conforme aumenta su orden de magnitud.

Al igual que en la representación en coma fija, tan solo se pueden expresar de manera exacta un conjunto finito de elementos. Todos los números expresados sin error son racionales con una cantidad finita de cifras y quedan fuera, además de los números irracionales, todo aquel número racional que no quepa en la longitud de palabra establecida. Para el resto de los valores que se encuentran en el intervalo de representación se realizan aproximaciones por algún número representable discreto. Por estos motivos, el formato establece una correspondencia entre el conjunto $\mathbb{Q}$ y los números racionales que representa de modo exacto y, debido al redondeo, no es posible recuperar el valor original desde la codificación realizada.

Se observa que en relación con el esquema anterior, se consigue cubrir un mayor espacio de la recta real aunque la cantidad de números representables de manera exacta es la misma que en coma fija [Sun Microsystems, 2000], [Kornerup y Matula, 1991], [Sterbenz, 1974], [Wilkinson, 1964].

En la tabla 2-2 se muestra un ejemplo de la representación con los mismos números y condiciones que en el formato de coma fija. En este caso se toma $L(m) = 5$, $L(e) = 5$. La mantisa se encuentra normalizada con el primer uno significativo a la izquierda de la coma fraccionaria.

$x \in \mathbb{Q}$	$A_{cflotante}(x)$		$\lvert Error \rvert$ $\lvert A_{cflotante}(x) - x \rvert$
	Exponente	Mantisa	
$123 \cdot 10^{1234567890}$	No representable		--
$123 \cdot 10^{123585}$	No representable		--
$123 \cdot 10^{3585}$	03587	1,2300	0
$123 \cdot 10^{7}$	00009	1,2300	0
123456	00005	1,2346	4
123,456	00002	1,2346	0,004

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001100101000100101001000000100100111000001000100010010011001111110011000111010 0

12345,67890123	00004	1,2346	0,32109877
0,123456	-00001	1,2346	0,000004
0,00000000000012	-00013	1,2300	0
$123 \cdot 10^{-7}$	-00005	1,2300	0
$123 \cdot 10^{-3585}$	-03583	1,2300	0
123,45454545…	00002	1,2345	0,0045454545…
0,017171717…	-00002	1,7172	0,00000028282828

Tabla 2-2: Ejemplos de representación en coma flotante

Por su amplia utilización cabe realizar una mención especial a los errores que produce el procesamiento en coma flotante. Entre los innumerables ejemplos de falta de exactitud se describen los siguientes como representativos a pesar de lo sumamente sencillos.

Ejemplo1:

Sea la siguiente operación,

$$r = a - b - c$$

para los números: $a = 0,6$; $b = 0,35$; $c = 0,25$.

El cálculo con el formato en coma flotante de simple precisión de IEEE obtiene $r = 0,00000002980232238769$. El resultado exacto es $r = 0,0$.

Ejemplo 2:

Sean las siguientes matrices:

$$A = [-10^{18}, 2246, 10^{27}, 10^{25}, 22, 10^{5}]$$

$$B^T = [10^{38}, 33, 10^{29}, -10^{22}, 1044, 10^{42}]$$

El cálculo de su producto con el formato de representación de doble precisión de IEEE es $A \times B^T = 0$. El resultado exacto es $A \times B^T = 97,086$.

Se observa, por tanto, que mediante el sistema de representación de datos en coma flotante IEEE-754 se obtiene un resultado incorrecto en

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011001010001011100000001101110000011100110101000100101001000000100100111000001000100010011001111110011000111010

mayor o menor grado. El tamaño de este error puede no hacer viable su aplicación en determinados problemas.

11,0010010000111111010101000100010000101101000110000100011010011000100011100011001101000101000100000100100111000001000100101001100111111001100011110100

Representación de números racionales en formato de doble mantisa

En los esquemas de representación de coma fija y coma flotante se ha comprobado que, a pesar de que su espacio de valores representable está contenido en $\mathbb{Q}$, existen infinitos números racionales no expresables aunque se permitan longitudes arbitrarias de $L(n)$. Es decir, por mucho que se aumente la cantidad de cifras representables y las longitudes de palabra es imposible la codificación exacta de tales valores.

Se plantea avanzar en la expresión numérica proponiendo un formato de representación que suministre una mayor potencia expresiva y sea capaz de cubrir un conjunto numérico más amplio. Esta nueva función de correspondencia abarca también la representación de los valores racionales que sistemáticamente quedan fuera de los anteriores esquemas, en concreto los valores racionales periódicos. El esquema de representación que se propone supone una extensión de los anteriores formatos al adoptar algunas de sus características. Se toma como base el método de Hehner y Horspool [Hehner y Horspool, 1979] y el esquema de coma flotante. De este último toma su flexibilidad y añade una segunda mantisa que representa explícitamente el periodo de los valores racionales.

El formato distribuye la cantidad de cifras significativas de la representación del número, $L(n)$, en cuatro partes: signo, exponente, $L(e)$, mantisa fija, $L(m_f)$, y mantisa periódica, $L(m_p)$. La mantisa fija constituye la parte fraccionaria del número racional no periódica, mientras que la mantisa periódica representa las cifras que forman la parte repetitiva. El exponente, al igual que en el formato de coma flotante, expresa el orden de magnitud del número.

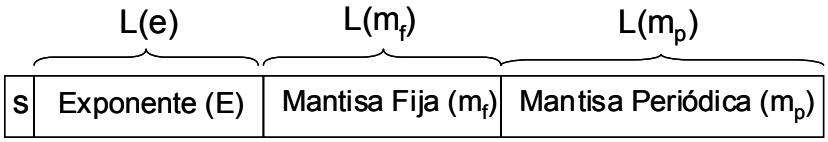


Figura 2-4: Esquema general del formato de representación de doble mantisa

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001100101000010110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111101 00

La mantisa del número se confecciona por la concatenación de la mantisa fija y la mantisa periódica una cantidad indefinida de veces, tal como se muestra en la figura 2-5. De este modo se representan números con infinitas cifras fraccionarias a partir de una codificación con una cantidad de cifras finita.

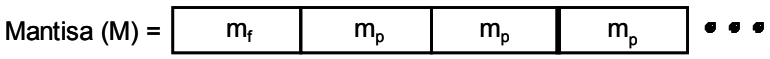


Figura 2-5: Esquema de la formación de la mantisa

Para evitar la múltiple codificación de un número y conseguir una estructura compacta, el modelo contempla una serie de características adicionales que condicionan la representación:

1) La mantisa del número se normaliza situando su primer dígito con valor distinto de cero a la derecha de la coma fraccionaria. Si no existe mantisa fija se normaliza la mantisa periódica rotando sus cifras componentes hacia la izquierda.

   $$M = 0,m_f\widehat{m}_p \,/\, M \in [B^{-1}, 1) \qquad [2.24]$$

   donde B es la base de la representación.

2) La mantisa fija no debe contener secuencias de dígitos iguales a los de la mantisa periódica en su parte menos significativa.

   Sea $m_f = \alpha_{L(mf)-1}\alpha_{L(mf)-2}\ldots\alpha_1\alpha_0 \wedge m_p = \gamma_{L(mp)-1}\gamma_{L(mp)-2}\ldots\gamma_1\gamma_0$

   Entonces:

   $$\forall\, i \in [0..L(m_p)\text{-}1],\ \alpha_{i-1}\alpha_{i-2} \ldots \alpha_1\alpha_0 \neq \gamma_{i-1}\gamma_{i-2} \ldots \gamma_1\gamma_0 \qquad [2.25]$$

3) La mantisa periódica tiene que estar formada por la menor cantidad de dígitos periódicos.

   Sea $m_p = \gamma_{L(mp)-1}\gamma_{L(mp)-2}\ldots\gamma_1\gamma_0$

   Entonces: $\forall\, i \in [1..L(m_p)\text{-}1] \,/\, L(m_p) \bmod i = 0 \Rightarrow$

   $\Rightarrow \gamma_{i-1}\ldots\gamma_1\gamma_0 \neq \gamma_{2i-1}\ldots\gamma_1\gamma_i \wedge$

   $\wedge\ \gamma_{i-1}\ldots\gamma_1\gamma_0 \neq \gamma_{3i-1}\ldots\gamma_{2i+1}\gamma_{2i} \wedge$

   …

11,0010010000111111011010100010000100010110100011000010001101001100011000110011000101000101110000000111001101000100101001000000100100111000001000100010100111001111100110001110100

$$\wedge\ \gamma_{i-1}\ldots\gamma_1\gamma_0 \neq \gamma_{L(mp)-1}\ldots\gamma_{L(mp)-i+2}\gamma_{L(mp)-i+1} \qquad [2.26]$$

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001100101000010111000000011011100000111001101000100101001000000100100111000001000100010010011001111110011000111010 0

El valor del número se obtiene mediante una expresión similar a la del formato de coma flotante:

$$A_{\mathbb{Q}} = (-1)^{s} \cdot M \cdot B^{E} \qquad [2.27]$$

Siendo B la base de la representación, s el signo expresado en un convenio discreto y, M y E el valor numérico de la mantisa completa y del exponente respectivamente.

El valor de M se calcula con la expresión matemática descrita en [2.18]:

$$M = \frac{m_f \, m_p - m_f}{\underbrace{(B-1)\cdots(B-1)}_{L(m_p)} \; \underbrace{0 \cdots 0}_{L(m_f)}} \qquad [2.28]$$

En caso de que no exista mantisa fija o la mantisa periódica sea igual a cero, su campo correspondiente en la codificación no contendrá ningún dígito, evitándose la representación de cifras innecesarias y no significativas. En estos casos el valor de la mantisa se calcula mediante las siguientes expresiones:

$$L(m_f) = 0 \Rightarrow M = \frac{m_p}{\underbrace{(B-1)\cdots(B-1)}_{L(m_p)}} \qquad [2.29]$$

$$m_p = 0 \Rightarrow M = \frac{m_f}{\underbrace{100\cdots00}_{L(m_f)}} \qquad [2.30]$$

Mediante la utilización de una cantidad variable de cifras para cada campo en función de sus necesidades de codificación se aporta una mayor flexibilidad y potencia expresiva al modelo. Esta característica da la posibilidad de mantener campos vacíos si es necesario y de ajustar la mantisa periódica a su longitud.

Si no se consideran las limitaciones físicas del espacio material de representación de la máquina y se asumen longitudes arbitrarias de los

campos del formato, el conjunto de números representables de manera exacta con este método abarca a todo el conjunto $\mathbb{Q}$. En esta situación, la función de representación propuesta corresponde con una aplicación biyectiva entre el conjunto de los números racionales.

Sea F la función de correspondencia que constituye el formato de representación:

$$F: \mathbb{Q} \rightarrow \mathbb{Q} \qquad [2.31]$$

Se resaltan sus características de interés siguientes:

- *Inyectiva*: cada valor racional tiene una codificación distinta. La cantidad de cifras fraccionarias de los números racionales es finita o bien es infinita periódica con periodo finito.

  $$\forall\, x_1, x_2 \in \mathbb{Q},\ F(x_1) \neq F(x_2) \qquad [2.32]$$

- *Sobreyectiva*: cualquier valor racional se puede representar con el formato propuesto.

  $$\forall\, x \in \mathbb{Q},\ \exists\, s, e, m_p, m_f \,/\, F(x) = x \qquad [2.33]$$

- *Existencia de inversa*: debido a la conjunción de las dos propiedades anteriores, la función de correspondencia dispone de inversa sobre el conjunto $\mathbb{Q}$, es decir, es posible construir una función que obtenga el valor racional inicial de cada codificación.

  $$\exists\, F^{-1}: \mathbb{Q} \rightarrow \mathbb{Q} \,/\, \forall\, x \in \mathbb{Q},\ x = F^{-1}(F(x)) \qquad [2.34]$$

Como consecuencia de estas propiedades todo número racional normalizado según el formato tendrá una expresión característica, compuesta por un signo, un exponente, una mantisa fija y una mantisa periódica.

Esta aplicación representa sin error los elementos de un conjunto numérico más amplio que el de las representaciones en coma fija o en coma flotante anteriores, las cuales son un caso particular del mismo. El nuevo formato constituye una función computable que mejora notablemente la capacidad de expresión de otros métodos que también muestran el valor numérico directo y evita los errores en la introducción de datos por el usuario a un computador que provoca el cambio de base

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001 0001010011001111110011000111010 0

de codificación desde decimal a binario. Por otra parte, admite procedimientos de cálculo que procesen números representados en este nuevo formato y que mantengan su capacidad de proveer un resultado exacto.

La tabla 2-3 muestra la representación con este formato de los ejemplos numéricos utilizados en los formatos anteriores. Se utilizan 10 dígitos repartidos entre el exponente, la mantisa fija y periódica.

$x \in \mathbb{Q}$	Formato propuesto, F(x)			|Error| |F(x) − x|
	Exponente	Mantisa fija	Mantisa periódica	
$123 \cdot 10^{1234567890}$	No representable			--
$123 \cdot 10^{123585}$	123588	0,123	--	0
$123 \cdot 10^{3585}$	03588	0,12300	--	0
$123 \cdot 10^{7}$	10	0,12300	--	0
123456	6	0,123456	--	0
123,456	3	0,123456	--	0
12345,67890123	5	0,12345678	--	0,0000012
0,123456	0	0,123456	--	0
0,0000000000001	-12	0,123	--	0
$123 \cdot 10^{-7}$	-4	0,123	--	0
$123 \cdot 10^{-3585}$	-03582	0,123	--	0
123,45454545…	3	0,123	45	0
0,017171717…	-1	--	17	0

Tabla 2-3: Ejemplos de representación en el formato propuesto

Como muestra la tabla anterior, siempre que la cantidad de cifras significativas del número sea menor o igual que los dígitos disponibles se permite su representación sin error. Sin embargo se observa que la instrumentación que se realice del formato va a estar sometida a las restricciones que imponga el espacio material de representación.

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010100100000010010011100000100010100100111111001100011110100

En el siguiente apartado se describe la instrumentación propuesta que relaciona los valores que intervienen en un problema con su codificación correspondiente en el marco de una arquitectura de procesador.

II - 2

Instrumentación de la función identidad

De acuerdo con la formulación del problema, la concepción de un procesador necesita de un método que proporcione una codificación de los valores numéricos con los que opera. Con este propósito se concibe el procedimiento de representación de los operandos en un formato adecuado para su procesamiento. La operatoria contempla operaciones de transferencia y almacenamiento flexible de los números.

Desde un enfoque matemático, se observa que un formato de representación corresponde con la función identidad,

$$f \equiv \text{identidad}: \mathbb{Q} \rightarrow \mathbb{Q} \qquad [2.35]$$

tal que,

$$\forall\, x \in \mathbb{Q},\ \text{identidad}(x) = x \qquad [2.36]$$

Formalmente, el procedimiento de representación numérica se considera como una instrumentación, $\Gamma_{\text{identidad}}$, de la función identidad

11,00100100001111110110101001000100001011010001100001000110100110001001100011001101000101100000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

cuyo objetivo es el de codificar en un computador elementos pertenecientes al conjunto de los números racionales.

La instrumentación según un diseño particular proporciona una codificación del número original, la cual, de acuerdo con la ecuación [1.2], cumple que:

$$\forall\, x \in \mathbb{Q},\ |\, \Gamma_{identidad}\,(x) - x| \leq \varepsilon \qquad [2.37]$$

Cualquier función de correspondencia entre valores racionales y su codificación en un computador (coma fija, coma flotante y el formato propuesto en el apartado anterior) encaja como instrumentación $\Gamma_{identidad}$, ya que realiza implementaciones concretas de métodos que representan valores racionales con un cierto grado de aproximación al valor exacto. Estos formatos constituyen instrumentaciones distintas de la función identidad.

La posibilidad de representación exacta de un formato está directamente relacionada con la existencia de inversa de la instrumentación en el conjunto numérico. Esta propiedad, reflejada en la expresión siguiente para el conjunto de los números racionales, establece un procedimiento de caracterización de formatos de representación sin error.

$$\forall\, x \in \mathbb{Q},\ \Gamma_{identidad}\,(x) = x \ \Leftrightarrow \forall\, x \in \mathbb{Q}, \exists\ \Gamma^{-1}_{identidad} \qquad [2.38]$$

Las características del esquema de representación propuesto en el apartado anterior garantizan la representación exacta de cualquier valor racional con la única restricción de las limitaciones físicas de la máquina sobre la que se implemente y no por restricciones impuestas por el propio formato numérico. En consecuencia se escoge esta función de correspondencia para construir la *instrumentación* Γ *de la función identidad*.

El carácter de precisión variable de la función se adquiere mediante algún parámetro $\bar{d}$ adicional que regule el grado de aproximación de la función. Debido a las características del formato que se propone se asegura que para cualquier número racional existe una instancia de $\bar{d}$ con los que se alcanza su valor exacto.

11,0010010000111111011010100010001000010110100011000010001101001100011001100110100101000110111000000111001101000100101001000000100100111000001000100010010011001111110011000111010 0

$$\forall x \in \mathbb{Q}, \exists \bar{d} \ / \ \Gamma^{VP}_{identidad}(x, \bar{d}) = x \qquad [2.39]$$

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100010100010111000000011100110100010010100100000010010011100000100010100110011111100110001110100

El parámetro $\bar{d}$ admite distintas configuraciones. Una posible consiste en establecer $\bar{d}$ como un vector de números naturales que indican la cantidad de cifras que, como máximo, contienen los campos de la representación, es decir,

$$\bar{d} \equiv < d_1, d_2, d_3>, \text{ con } d_i \in \mathbb{N} \qquad [2.40]$$

siendo, d_1, d_2 y d_3 la cantidad de cifras límite para la codificación del exponente, la mantisa fija y la mantisa periódica respectivamente.

Las condiciones sobre las longitudes de los campos vienen dadas por la siguiente expresión:

$$(L(e) \leq d_1) \wedge (L(m_f) \leq d_2) \wedge (L(m_p) \leq d_3) \qquad [2.41]$$

Otra configuración establece $\bar{d}$ como la cantidad máxima de cifras que se destinan a la representación del número en su conjunto sin especificar qué cantidad de dígitos se asigna a cada campo concreto. Esta opción aporta flexibilidad a la representación.

$$\bar{d} \equiv d \in \mathbb{N} \qquad [2.42]$$

La relación que ilustra esta condición se muestra en la ecuación siguiente:

$$1 + L(e) + L(m_f) + L(m_p) \leq d \qquad [2.43]$$

Configuraciones intermedias pueden establecer el tamaño de algunos campos del número y limitar el crecimiento conjunto de otros.

La interpretación de la restricción $\bar{d}$ en la instrumentación de la función corresponde con las limitaciones del espacio material de representación y constituye el tamaño máximo de la palabra o de la memoria disponible para la codificación del número o alguna de sus partes.

Como muestra la figura 1-1, el parámetro $\bar{d}$ es gestionado por la unidad de control de precisión para establecer las características de la arquitectura acordes con su valor. La instrumentación de esta gestión está estrechamente relacionada con la cantidad y capacidad de los registros de la unidad y presenta varias posibilidades de diseño. La primera alternativa consiste en implantar un juego de registros de

11,0010010000111111011010100010001000010110100011000010001101001100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

tamaño fijo para mantener los datos. En este caso, las restricciones impuestas suponen un límite a la cantidad de espacio utilizado de los registros, sin poder sobrepasar su longitud. Otra posibilidad consiste en la creación de diversos juegos de registros de distintas longitudes y escoger el más favorable al valor de $\bar{d}$ para realizar los cálculos y contener los datos, ignorando el resto. También puede considerarse la construcción de registros mediante lógica reconfigurable con capacidad para adecuar dinámicamente su longitud y número en función de los requerimientos del problema. La técnica de gestión de la precisión que se utilice debe ser transparente al resto de módulos la unidad, de modo que en lo sucesivo, el desarrollo de los algoritmos y la instrumentación de las funciones considera en todo momento registros de una longitud fija w establecida.

A partir de los conceptos presentados hasta el momento se adopta la instrumentación $\Gamma_{identidad}^{VP}$ como formato de representación flexible de los números racionales. Esta relación corresponde con la *instrumentación* Γ *en precisión variable de la función identidad*. Sus características más importantes son las siguientes:

- Representación de los números mediante una notación fraccionaria en coma flotante en la que se codifica el periodo de los valores racionales con un conjunto de cifras específico que constituye la mantisa periódica.
- Adopción de una representación flexible de acuerdo con los requerimientos definidos por un parámetro $\bar{d}$ que interviene en la codificación.
- Concepción de operadores aritméticos que preserven las propiedades de exactitud y flexibilidad.

Una arquitectura que instrumente la función *identidad* para los números racionales mediante $\Gamma_{identidad}^{VP}$ representa una arquitectura en precisión variable.

$$A_{\Phi}^{VP} = \{\Gamma_{identidad}^{VP}\} \qquad [2.44]$$

Construcción de la función identidad

En este apartado se consideran los aspectos relativos a la arquitectura del procesador que implementan el formato de representación propuesto. Se destaca el interés por el diseño de $\Gamma^{VP}_{identidad}$ según las características de su instrumentación así como la inclusión del operador identidad como primitiva de la arquitectura junto con otros operadores aritméticos. Se adopta la base de representación binaria para la codificación de las cifras por razones obvias.

El planteamiento es esencialmente de modelización y habrá de ser la realización aplicada sobre la base de requerimientos reales la que incorpore y plantee los aspectos concretos relativos a la implementación: tecnología a utilizar, camino crítico, etc. En realidad, se espera que investigaciones posteriores profundicen en los aspectos de realismo de implementación hardware. No obstante, por cuestión de completitud, se ha decidido abordar en este apartado un supuesto de realización aportando líneas a seguir como punto de partida para su posterior desarrollo y perfeccionamiento.

Teniendo en cuenta lo anterior, el diseño propuesto para su implementación hardware se sustenta en la disposición de los campos que conforman el número en una palabra de longitud finita y en aportar flexibilidad en la distribución de las cifras fraccionarias entre la parte fija y periódica. La longitud w de los registros habrá sido establecida previamente por el módulo de gestión de la precisión según se ha mencionado. El formato consiste en cuatro campos diferenciados (figura 2-4): *signo, exponente, mantisa fija y mantisa periódica.* El significado y la codificación de cada campo es el siguiente:

- *Signo (s):* indica el signo del número mediante un solo bit con el convenio: 0 positivo, 1 negativo.
- *Exponente (e):* almacena el exponente del formato en coma flotante expresado en representación sesgada.
- *Mantisa fija (m_f):* secuencia de cifras que forma la parte fraccionaria del número racional no periódica. La mantisa fija se encuentra normalizada con el primer bit significativo a la derecha de la coma

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

fraccionaria. En caso de que no exista mantisa fija este campo no contendrá ninguna cifra.

- *Mantisa periódica (m_p):* secuencia de cifras que almacena la parte fraccionaria periódica del número racional.

11,0010010000111111011010100010000101101000110000100011010011000100111000101000101110000000111001101000100101001000000100100111000001000100101001100111111001100011110100

El valor del número se obtiene según la expresión siguiente:

$$N = (-1)^s \cdot 0,m_f\hat{m}_p \cdot 2^E \qquad [2.45]$$

donde,

E: Es el valor del exponente teniendo en cuenta su representación sesgada: $e = E_S$.

$0,m_f\hat{m}_p$: Es el valor obtenido al establecer a la derecha de la coma las cifras fraccionarias de m_f concatenadas con la parte periódica m_p indefinidamente.

Este valor es equivalente al de la expresión [2.22] y tiene en cuenta la representación concreta de cada campo.

Las partes del número se colocan consecutivamente ocupando un registro de longitud fija. El signo, el exponente y la mantisa disponen de una cantidad de posiciones determinada para su representación, donde ésta última está formada por las cifras fijas y periódicas del número. Con esta instrumentación, el valor del parámetro $\bar{d}$ que limita la precisión viene dado por la cantidad de cifras que se destinan al exponente y la mantisa en su conjunto. En lo que sigue se tomará esta configuración como significado propio de $\bar{d}$ y se usará un par de números naturales para su expresión, como se muestra a continuación,

$$\bar{d} \equiv <d_{L(e)}, d_{L(m)}>, \text{ con } d_{L(e)} \in \mathbb{N} \wedge d_{L(m)} \in \mathbb{N} \qquad [2.46]$$

tal que, las longitudes de los campos cumplen la siguiente relación:

$$(L(e) \leq d_{L(e)}) \wedge (L(m_f) + L(m_p) \leq d_{L(m)}) \qquad [2.47]$$

Gráficamente, el registro que contiene el dato mantiene la siguiente distribución de las longitudes de sus campos:

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

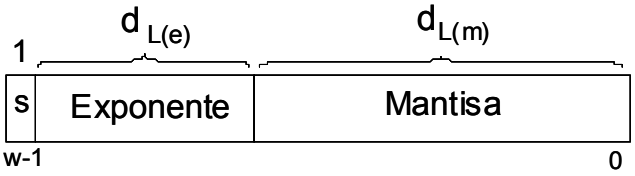


Figura 2-6: Distribución de las cifras de los campos del número

El reparto flexible de las cifras de la mantisa entre la parte fija y periódica de la misma requiere de un puntero que marque la separación entre ambas y permita su procesamiento separado. Para completar todas las cifras asignadas al campo de mantisa se concatenan las cifras del periodo formando un ciclo y se almacena su longitud y la de la mantisa fija junto con el puntero anterior. Estos tres datos se asocian al registro que contiene el número y se colocan adyacentes a él, como se observa en la estructura que ilustra la siguiente figura:

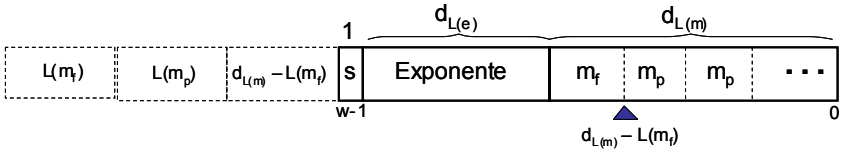


Figura 2-7: Estructura de la implementación del formato de doble mantisa

La mantisa de los números que carecen de parte fija se construye únicamente mediante la concatenación de las cifras de la mantisa periódica. En este caso, el puntero que marca la separación entre las mantisas toma el valor $d_{L(m)}$.

Cuando la longitud de alguna parte del número sobrepasa el límite impuesto por $\bar{d}$ no es posible su expresión exacta y será necesario ajustar la codificación al valor representable más cercano. El criterio que se propone es el de representar completamente el exponente y el signo y aplicar el recorte en la mantisa. Cuando ésta queda afectada por la limitación de cifras $d_{L(m)}$ se considera su expresión aproximada y se descartan las últimas cifras siguiendo el orden de magnitud de los dígitos.

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000111011000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

Se presentan varias situaciones en relación con el valor de la condición $\bar{d}$ y la cantidad de cifras de cada campo de la representación:

- Si la cantidad de cifras $<d_{L(e)}, d_{L(m)}>$ es suficiente para la codificación completa del exponente y la mantisa en su conjunto se realiza su representación exacta.

$$(d_{L(e)} > L(e)) \wedge (d_{L(m)} \geq L(m_f) + L(m_p)) \Rightarrow \textit{Representación exacta} \qquad [2.48]$$

11,0010010000111111011010100010001000010110100011000010001101001100010011001100110001010001011100000001101110000011100110100010010010000001001001110000010001 0001010011001111110011000111010 0

- Si la cantidad de cifras $d_{L(m)}$ no es suficiente para la representación completa de la mantisa del número se obtiene una representación aproximada.

  $$(d_{L(e)} > L(e)) \wedge (d_{L(m)} < L(m_f)+L(m_p)) \Rightarrow \textit{Representación aproximada} \quad [2.49]$$

  Se plantean dos escenarios en la codificación según el campo del número al que afecte el recorte de dígitos:

  - Si no es posible la codificación completa de la mantisa periódica carece de sentido su expresión incompleta y se elimina. Se asigna un valor cero al indicador de la longitud del periodo y utilizan sus dígitos para construir una mantisa fija extendida que aproveche la longitud total disponible.

    $$d_{L(m)} < L(m_f) + L(m_p) \quad [2.50]$$

    Tras esa reasignación de cifras se redondea el valor aproximado de la mantisa con el propósito de obtener un resultado con el menor error posible. Como ilustra la figura siguiente, el redondeo se realizará con las cifras del periodo que quedan fuera del límite. Se aplica cualquier sistema de redondeo referido en la bibliografía utilizando para ello la cantidad de dígitos necesaria [Even y Seidel, 2000], [Bruguera y Lang, 2000], [Park et al, 1999], [Quach et al, 1991] y [IEEE, 1985].

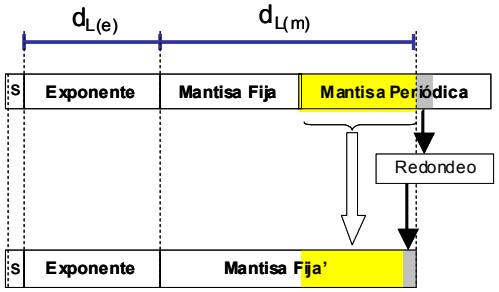


Figura 2-8: Transferencia de dígitos entre mantisas periódica y fija y redondeo

  - Si tampoco es posible la codificación completa de la mantisa fija se desecha directamente la mantisa periódica asignando al indicador de su longitud un valor cero y se procede a ajustar la mantisa fija mediante un redondeo sobre los dígitos sobrantes.

11,00100100001111110110101001000100001011010001100001000110101100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

$$d_{L(m)} < L(m_f) \qquad [2.51]$$

En ambos casos, la posición de comienzo de la mantisa periódica que marca el puntero asociado es irrelevante.

- Finalmente, si la cantidad de dígitos disponibles $d_{L(e)}$ no abarca la representación del orden de magnitud, el número no es representable con las condiciones impuestas. En este caso el resultado mostrará una expresión de error.

$$d_{L(e)} \leq L(m_e) \Rightarrow \textit{Número no representable} \qquad [2.52]$$

El error que se produce en la codificación aproximada será cuantificado por el orden de magnitud del número. Por este motivo hay que seleccionar con especial cuidado el valor de la restricción $\bar{d}$ y considerar no sólo aspectos relativos al valor de los datos a codificar sino también al tipo y número de operaciones que intervienen. Las características del formato de representación mencionadas en el apartado anterior aseguran que para cualquier conjunto de valores racionales existe un tamaño de registro adecuado para su representación exacta. Si bien, será conveniente buscar una longitud de compromiso entre complejidad de la representación y capacidad de expresión.

La consideración de los campos de la mantisa en su conjunto, tanto en su representación exacta como aproximada, permite equiparar el formato de representación propuesto de doble mantisa con el clásico de coma flotante y presenta la capacidad adicional de codificación sin error de un conjunto de números racionales periódicos. En concreto, si se establece la estructura de los registros según el tamaño de los campos que indica la norma IEEE-754 se obtiene una representación compatible con la misma. Este hecho faculta el procesamiento de los números mediante la operatoria existente para coma flotante. Sin embargo, se obtienen ventajas de concebir métodos de operación adecuados para el nuevo formato que aprovechen su capacidad de expresión exacta y produzcan igualmente resultados sin error.

Las tablas siguientes muestran algunos ejemplos característicos de la codificación que realiza el formato de representación en comparación con el esquema estándar de representación en coma flotante. Se emplea

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001 00010100110011111100110001110100

una longitud de registro de 32 bits distribuidos entre 1 para el signo, 8 para el exponente y 23 para la mantisa ($d_{L(e)} = 8 \wedge d_{L(m)} = 23$) y se establece un sesgo para el exponente de $2^{L(e)-1}-1$. El conjunto de muestras pretende ser lo más heterogéneo posible y abarca números racionales periódicos y no periódicos tanto en decimal como en binario.

11,0010010000111111011010100010000100010110100011000010001101001100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010100

$N_i \in \mathbb{Q}$	Valor decimal	Valor binario
N_1	0,125	0,001
N_2	0,2	0,0011001100110011...
N_3	0,6	0,1001100110011001...
N_4	2,18	10,001011100001010001111.
N_5	$7,\widehat{5}$	111,1000111000111...
N_6	25,04	11001,000010100001111010.
N_7	$37,\widehat{21}$	100101,0011011001...
N_8	$64,\widehat{01369863}$	1000000,000000111...

(a)

N_i	Representación según el formato IEEE de Simple Precisión		
	Exp.	Mantisa	Error
N_1	01111100	00000000000000000000	0
N_2	01111100	10011001100110011001 10	$1,19 \cdot 10^{-8}$
N_3	01111110	00110011001100110011 01	$7,15 \cdot 10^{-8}$
N_4	10000000	00010111000010100011 11	$1,72 \cdot 10^{-7}$
N_5	10000001	11100011100011100011 10	$2,11 \cdot 10^{-7}$
N_6	10000011	10010000101000111101 10	$2,75 \cdot 10^{-6}$
N_7	10000100	00101001101100100110 11	$1,16 \cdot 10^{-6}$
N_8	10000101	00000000000011100000 10	$3,86 \cdot 10^{-6}$

(b)

N_i	Representación según el formato propuesto			
	Exp.	m_f	m_p	Error
		Mantisa conjunta		
N_1	01111101	1	0 00000000000000000000	0
N_2	01111101	--	1100 1100110011001100110	0

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001010001011100000001101110000011100110100010010010000001001001110000010001
000101001100111111001100011101 00

N_3	01111111	-- 1001 10011001100110011001100	0
N_4	10000001	100 01011100001010001111 1000101110000 101000111	$2{,}91 \cdot 10^{-7}$
N_5	10000010	1 111000 11110001110001110001110	0
N_6	10000100	11001 00001010001111010111 11001000010100011110110	$1{,}38 \cdot 10^{-7}$
N_7	10000101	1001 0100110110 10010100110110010011011	0
N_8	10000110	1000000 000000111 10000000000001110000001	0

(c)

Tabla 2-4: Ejemplos de representación de números racionales
(a) Valores racionales en decimal y binario
(b) Representación en el formato IEEE Simple Precisión
(c) Representación en el formato propuesto

Los ejemplos de las tablas anteriores ponen de manifiesto que el formato propuesto realiza la representación exacta siempre que la cantidad de cifras significativas de los números sea menor o igual que las posiciones disponibles y es capaz de codificar números con una cantidad de cifras infinita gracias a la existencia de una mantisa periódica.

El modelo de representación deja abierta la posibilidad de plantear codificaciones especiales para expresar aquellas situaciones anómalas o de excepción que se pueden producir: representación del cero, infinito, desbordamiento y error [Schulte, 2000], [IEEE, 1985].

Evaluación empírica

Una vez conocidas las características del método de representación propuesto así como su capacidad de expresión se realiza una serie de pruebas con ánimo de analizar su comportamiento. Los experimentos se centran en estudiar la codificación de la parte significativa del número considerando que tanto el exponente como el signo se representan correctamente en todos los casos. Se han realizado dos conjuntos de

11,0010010000111111011010100010001000100001011010001100001000110100110001001100011001100010100010111000000011101100000111001101000100101001000000100100111000001000100010100110011111001100011101100

pruebas mediante una simulación del formato en un entorno de programación en C[1].

Experimentos I

En este primer bloque de experimentos se analiza la relación de números con mantisa fija y periódica así como la cantidad de cifras necesarias en la codificación de los valores racionales.

La siguiente tabla muestra los 80 primeros términos de la serie armónica en su representación binaria fraccionaria posicional antes de su codificación por el formato.

1/n	m_f	m_p
1/1	0	0
1/3	--	01
1/5	--	0011
1/7	--	001
1/9	--	000111
1/11	--	0001011101
1/13	--	000100111011
1/15	--	0001
1/17	--	00001111
1/19	--	000011010111100101
1/21	--	000011
1/23	--	00001011001
1/25	--	00001010001111010111
1/27	--	000010010111101101
1/29	--	00001000110100111101110011011
1/31	--	00001
1/33	--	0000011111
1/35	--	000001110101
1/37	--	00000110111010110011111001000

1/n	m_f	m_p
1/2	1	0
1/4	01	0
1/6	0	01
1/8	001	0
1/10	0	0011
1/12	00	01
1/14	0	001
1/16	0001	0
1/18	0	000111
1/20	00	0011
1/22	0	0001011101
1/24	000	01
1/26	0	000100111011
1/28	00	001
1/30	0	0001
1/32	00001	0
1/34	0	00001111
1/36	00	000111
1/38	0	000011010111100101

[1] *Entorno de desarrollo*: C++ Builder 5.0 Professional. Borland software corporation.
Entorno hardware: Procesador Intel Pentium III 500 MHz 512 MB RAM.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001000101001100111111001100011110100

		1010011
1/39	--	000001101001
1/41	--	000001100011111100111
1/43	--	00000101111101
1/45	--	000001011011
1/47	--	000001010111001001100001
1/49	--	00000101001110010111 1
1/51	--	00000101
1/53	--	000001001101010010000111001111101100101010110111100011
1/55	--	000001001010011111001
1/57	--	000001000111110111
1/59	--	000001000101011011000111100101111110111010100100111100001101

1/40	000	0011
1/42	0	000011
1/44	00	0001011101
1/46	0	00001011001
1/48	0000	01
1/50	0	00001010001111010111
1/52	00	000100111011
1/54	0	000010010111101101
1/56	000	001
1/58	0	000010001101001111 0111001011
1/60	00	0001

1/61	--	00000100001100100101110001010 01111101111001101101000111010 11	1/62	0	00001
1/63	--	000001	1/64	000001	0
1/65	--	000000111111	1/66	0	0000011111
1/67	--	00000011110100100010011000110 10101111110000101101110110011 10010101	1/68	00	00001111
1/69	--	0000001110110101110011	1/70	0	000001110101
1/71	--	00000011100110110000101011010 001001	1/72	000	000111
1/73	--	000000111	1/74	0	000001101110101100 111110010001010011
1/75	--	00000011011010011101	1/76	00	000011010111100101
1/77	--	00000011010100110001110111101 1	1/78	0	000001101001
1/79	--	00000011001111011001000111010 0101010001	1/80	0000	0011

Tabla 2-5: Representación fraccionaria posicional binaria de los 80 primeros términos de la serie armónica

En las representaciones de la tabla 2-5 se observan diferencias en la codificación de los números según el denominador de la fracción que los genera. La mantisa fija está formada por una cantidad de ceros igual a la mayor potencia de dos divisor del denominador de la fracción. Si éste no fuera divisible entre dos entonces carece de mantisa fija. En cuanto a la mantisa periódica, se comprueba que todos los números contienen un periodo y que éste sólo está formado por el dígito cero cuando el denominador es potencia de dos.

El formato de representación propuesto contiene algunos cambios en relación con la codificación posicional que muestra la tabla 2-5 que no alteran la expresión directa del valor del número. Se mantiene un campo de exponente para expresar el orden de magnitud y las mantisas se expresan normalizadas para contener su valor en el intervalo [0,5; 1). El efecto inmediato que se provoca es la reducción de la longitud de la representación binaria.

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110100010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010

Para comprobar las características de la mantisa de los números codificados con el formato se realiza un conjunto de pruebas de representación de números racionales en un amplio intervalo.

11,0010010000111111010101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

El perfil de estos experimentos es el siguiente:

- Codificación de números racionales aleatorios[2] en el modelo de representación propuesto.
- Los valores pertenecen al intervalo (0..10000], donde cada número se genera mediante la construcción de una fracción $\frac{a}{b}$, siendo a y b valores enteros aleatorios en el rango [1..10000].
- Realización de 10^9 representaciones numéricas.

Los resultados obtenidos se muestran en la siguiente tabla:

Prueba a/b	%
Números con mantisa fija	61,10
Números con mantisa periódica distinta de 0	99,58
Números que carecen de algún tipo de mantisa	39,31

Tabla 2-6: Características de la representación racional de a/b en el formato propuesto

Los datos de la tabla anterior muestran que existe un conjunto de números, 38,90 %, para los que no es necesaria la mantisa fija. Este porcentaje es significativamente menor que el 50% obtenido para denominadores impares en la representación posicional directa que muestra la tabla 2-5. En este formato, la representación de la mantisa fija será necesaria cuando los números a codificar sean mayores o iguales a uno o cuando el denominador de la fracción que lo genera sea potencia de dos. También se observa que la cantidad de representaciones con mantisa periódica es muy elevada y prácticamente todos los números disponen de ella. Este hecho se debe a la naturaleza binaria de la base de representación.

[2] Para la generación de números enteros aleatorios se han utilizado las funciones *randomize* y *random* de la librería *stdlib.h* perteneciente al entorno de programación anteriormente mencionado.

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001000101001100111111001100011110100

Se realiza un segundo conjunto de pruebas, del mismo tipo que las anteriores, pero considerando ahora fracciones de la serie armónica con la forma $\frac{1}{b}$, con $b \in [1..10000]$ para evitar simplificaciones entre factores comunes de los numeradores y denominadores de las fracciones generadoras. Los datos que resultan de esta prueba se muestran en la tabla siguiente.

Prueba 1/b	%
Números con mantisa fija	0,14
Números con mantisa periódica distinta de 0	99,86
Números que carecen de algún tipo de mantisa	100

Tabla 2-7: Características de la representación racional de 1/b en el formato propuesto

En la tabla anterior se observa que debido a la normalización de la representación la mayoría de los números carecen de mantisa fija, y sólo la contienen aquellos cuyo denominador es potencia de dos.

Para determinar la complejidad espacial de las representaciones, las pruebas siguientes se orientan a estudiar la relación existente entre la cantidad de dígitos necesarios en la codificación de un número racional y el numerador y el denominador de la fracción que lo genera. Para una mayor claridad se presentan los resultados de forma separada para la mantisa fija y periódica en relación con el numerador y denominador de la fracción generadora respectivamente. En el caso general, la longitud de las mantisas estará en una posición intermedia consecuencia de las simplificaciones entre factores comunes de la fracción.

La figura 2-9 muestra cómo la cantidad de cifras necesaria para contener la mantisa fija aumenta de acuerdo con el logaritmo entero del numerador de la fracción, tal y como sucede en el resto de las codificaciones de coma flotante. La mantisa periódica permanece a cero.

11,0010010000111111011010100010000100010110100011000010001101001100011000110011000101000101110000000111001101000100101001000000100100111000001000100010011001111100110001110100

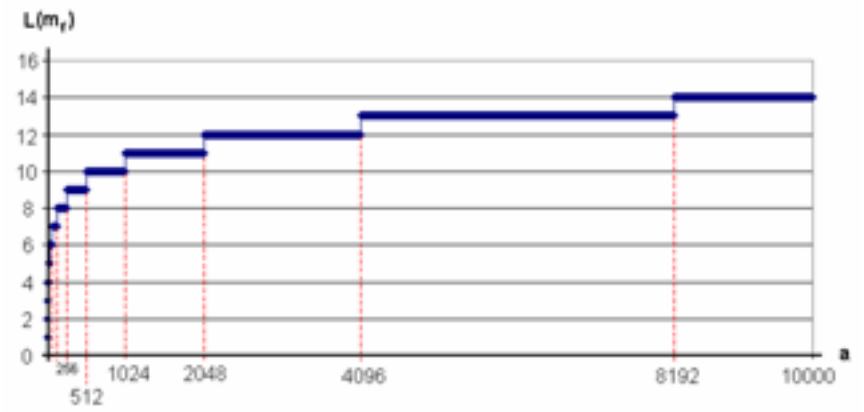


Figura 2-9: Cantidad de cifras necesaria en la representación de la mantisa fija de a/1

La figura 2-10 muestra una tendencia creciente de la cantidad de cifras de la mantisa periódica con respecto al denominador de la fracción. Esta circunstancia provoca que la cantidad de cifras del periodo dependa proporcionalmente de su valor, como era de esperar debido a su naturaleza.

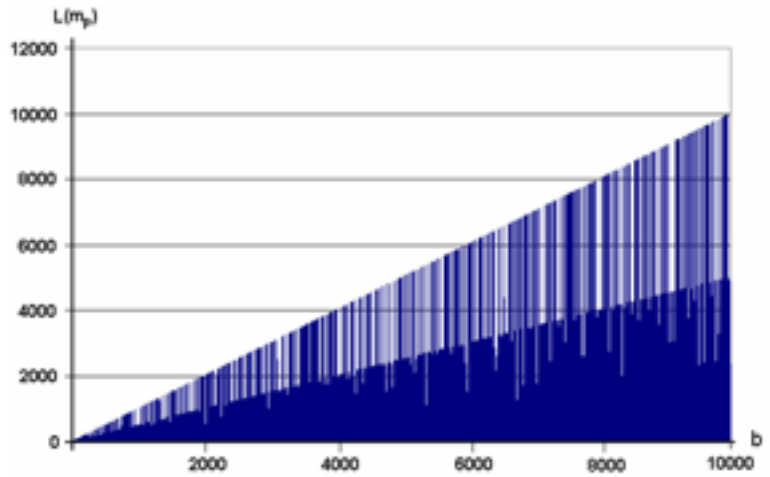


Figura 2-10: Cantidad de cifras necesaria en la representación de la mantisa periódica 1/b

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

Como se observa en las gráficas 2-9 y 2-10, la cantidad de cifras necesarias para la representación exacta de un número está en relación con el tamaño del numerador y denominador de la fracción según la siguiente expresión:

$$\forall\, x \in \mathbb{Q} \;/\; x = \frac{a}{b},\ \text{si}\ d_{L(m)} \geq b + \lg_2 a \Rightarrow \textit{Representación exacta} \qquad [2.53]$$

Esta cantidad de cifras puede ser menor debido a la presencia de factores comunes entre el numerador y el denominador de la fracción así como por denominadores con factores potencia de dos.

Experimentos II

El segundo grupo de pruebas tiene como objeto comparar el método propuesto y los métodos de representación convencionales presentes en la mayoría de los computadores de propósito general, en concreto el formato de representación IEEE-754. Para ello, se codifican números racionales periódicos y no periódicos tanto en el formato de representación propuesto como en el formato de representación IEEE-754 en simple y doble precisión. Los valores se toman en un intervalo normalizado entre 0 y 1 para centrar la atención en la precisión de las codificaciones realizadas de la mantisa y no en su orden de magnitud. De esta prueba se extraen conclusiones acerca de la expresividad de dicho formato y de las desviaciones que produce en la representación numérica racional. El perfil de los experimentos es el siguiente:

- Cada número no periódico pertenece al intervalo $[0, 1)$ y consta de una cantidad de 128 cifras fraccionarias aleatorias significativas.
- Cada número periódico pertenece al intervalo $(0, 1]$ y se construye mediante una fracción $1/b$, donde b es un valor entero aleatorio no potencia de 2 en el rango $[1..10000]$.
- Realización de 10^9 representaciones numéricas para cada caso.

Las pruebas se basan en la capacidad de expresión sin error de los números racionales con el método propuesto. El procedimiento consiste en comparar para cada número la codificación que proporcionan los formatos convencionales con las correspondientes cifras exactas de la

11,00100100001111110110101000100010000101101000110000100011010011000100111000100110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101100

representación en el formato propuesto, de forma que se establezca una medida absoluta del error cometido así como la desviación sobre la codificación correcta. De estas pruebas se obtiene una evaluación de la corrección de las codificaciones realizadas por los formatos de representación de IEEE-754 de simple y doble precisión en su capacidad de representación numérica.

Una primera orientación acerca de la calidad de la representación viene dada por la posición que ocupa el dígito de mayor orden de magnitud con valor distinto respecto al correspondiente de la representación exacta. Si bien no es indicativo de la dimensión del error cometido, sí que representan una medida de la similitud entre el valor exacto y el valor representado. La tabla 2-8 muestra el valor promedio de ésta primera posición con un dígito distinto de la mantisa. Se observa que en todos los casos su valor está muy cerca de la cantidad de cifras que se destinan a la representación.

Formato	Números no periódicos		Números periódicos	
	Posición	σ	Posición	σ
IEEE-754 Simple Precisión	23,36	8,97	23,08	8,34
IEEE-754 Doble Precisión	52,01	15,05	51,82	14,88

Tabla 2-8: Primera posición distinta en promedio

La tabla 2-9 muestra el promedio del error absoluto producido en valor de la mantisa sin considerar su orden de magnitud. Se debe señalar que debido a la naturaleza de formato en coma flotante estos errores se ven afectados por un exponente que puede amplificar su valor [Goldberg, 1991], [Bohlender, 1990].

Formato	Números no periódicos		Números periódicos	
	Error	σ	Error	σ

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

IEEE-754 Simple Precisión	$2{,}979 \cdot 10^{-8}$	$2{,}055 \cdot 10^{-13}$	$2{,}978 \cdot 10^{-8}$	$1{,}186 \cdot 10^{-13}$
IEEE-754 Doble Precisión	$5{,}551 \cdot 10^{-17}$	$1{,}856 \cdot 10^{-33}$	$5{,}549 \cdot 10^{-17}$	$1{,}027 \cdot 10^{-33}$

Tabla 2-9: Error promedio en la codificación IEEE-754 de simple y doble precisión

11,0010010000111111011010100010000100010110100011000010001101001100010011000110010100010111000000011101100000111001101000100101001000000100100111000001000100010010011001111100110001110100

Conclusiones

En este capítulo se ha presentado la función identidad en precisión variable a partir de un formato de representación de números racionales y su implementación a bajo nivel. De la misma se destacan los siguientes aspectos:

- El formato de representación de doble mantisa permite la codificación de números racionales sin error y constituye una alternativa de expresión numérica racional exacta a la representación simbólica.
- La codificación de los números se fundamenta en su representación fraccionaria posicional, lo que permite realizar operaciones aritméticas directamente sobre los campos del formato.
- La instrumentación de la función identidad hace un uso flexible de registros de tamaño fijo y mantiene las características del formato. En la representación aproximada de los datos el esquema se comporta como un formato clásico de coma flotante.
- Las pruebas realizadas ponen de manifiesto la información que se pierde en la codificación fraccionaria de los números racionales mediante las técnicas tradicionales. Los análisis realizados deben tomarse para valorar la conveniencia de su utilización en determinados problemas.

Capítulo III

Metodología de Operación

1. Estructura de los operadores
2. Operaciones aritméticas de números enteros

III - 1

Estructura de los operadores

La expresión sin error de valores racionales junto con la concepción y el diseño de operadores que procesan números expresados en este formato de representación componen los pilares sobre los que se fundamenta la computación exacta sobre los números racionales. Las funciones de interés serán aquellas que constituyen una ley de composición interna entre los elementos del conjunto $\mathbb{Q}$.

Con esos principios, se establece un modelo de computación formado por el esquema de representación numérica formalizado mediante la función identidad y las funciones de suma y producto sobre números racionales. El modelo debe disponer de un funcionamiento flexible para procesar operandos de distinta longitud y considerar, asimismo, elementos en el problema que limiten la precisión del resultado. En relación con este último aspecto, la instrumentación en precisión variable de las funciones incorpora un parámetro $\overline{d}$ que refleja estos requerimientos de acuerdo con la formulación del problema [1.6]:

$$\forall\ \overline{x} \in \text{dominio}(\Gamma_f),\ |\Gamma_f^{VP}(\overline{x},\ \overline{d}) - f(\overline{x})| \leq \varepsilon$$

11,0010010000111111010101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

La metodología de operación que se presenta en este apartado presta especial atención a la estructura de los operadores así como a su implementación. El resultado de este examen determinará la conveniencia de aplicar métodos de cálculo basados en el aumento de tamaño de operación elemental. La operatoria del procesador debe sacar partido de la estructura interna de los operandos y aprovechar sus características, además, su carácter variable favorece el empleo de técnicas iterativas que, por repetición, sean capaces de procesar todos las cifras del número.

Granularidad de los operadores

La estructura de los datos que establece el formato propuesto resulta difícil de encajar en los métodos rígidos que procesan operandos con una cantidad fija de cifras. Esta circunstancia sugiere la necesidad de buscar arquitecturas que coordinen la estructura de los operandos y la metodología de operación y sean capaces de adaptarse a números de distinto tamaño.

Tradicionalmente, la mayoría de operadores elementales consideran al bit como la unidad mínima de información procesable. Un nuevo paso en su evolución consiste en aumentar la granularidad de los operadores y considerar como unidad mínima de cálculo un conjunto de bits.

Como aspecto importante para la construcción de esos operadores se destaca el *retardo temporal* de la operación. También la *complejidad de la unidad aritmética* y el *área ocupada* por los operadores debe considerarse sobretodo en el diseño e implementación del circuito. En esto último influye de manera esencial el *grado de paralelismo* y *reutilización* de los módulos que constituyen la propia unidad aritmética. Finalmente, también son deseables características que aportan calidad al procesamiento, como son la *robustez* de los resultados y la *tolerancia a fallos* de los operadores. Otros aspectos, como el *consumo de potencia*, no se contemplan en este trabajo.

Se denomina *k-operadores*, a los operadores que toman como unidad mínima de proceso una cantidad de k bits. La idea fundamental consiste en obtener ventajas en la instrumentación de las funciones

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100101000010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

aritméticas utilizando en su construcción elementos k-operadores. Este aumento de granularidad contiene una serie de mejoras inherentes en aquellos casos en los que es posible su aplicación frente a la operatoria a nivel de bit. La figura 3-1 muestra esquemáticamente la funcionalidad de un k-operador genérico y de un operador a nivel de bit.

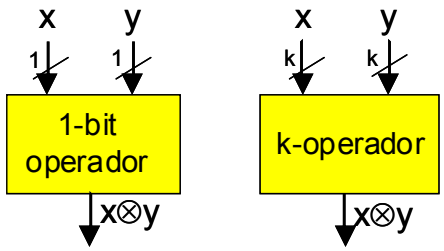


Figura 3-1: Operadores de 1 y k bits

Se simplifica la estructura de la unidad aritmética al utilizar menos unidades de procesamiento individual para operar grupos de bits. Por ejemplo el algoritmo de suma con propagación de acarreo (CPA —*Carry Propagate Adder*) [Zimmermann, 1987] utiliza tantas unidades sumadoras de un bit (FA—*Full-Adder*) como cifras tenga el número. En este caso, como se observa en la figura 3-2, la cantidad de lógica de interconexión debe ser igual a la cantidad de bits de los operandos.

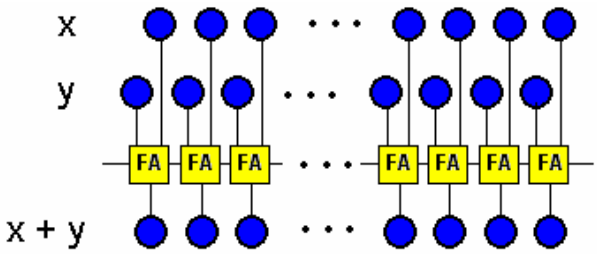


Figura 3-2: Suma CPA bit a bit

La figura 3-3 muestra cómo el diseño mediante k-operadores reduce la complejidad de interconexión al rebajar el número de enlaces entre los módulos de cálculo en un factor k. Alternativamente, con la misma cantidad de lógica de interconexión se opera con datos de mayor

11,0010010000111111010101000100001011010001100001000110101100010011000110010100010111000000011011100000111001101000100101001000000100100111000001000100010011001111100110001110100

tamaño. Esta cualidad puede mejorar algunas de las características de interés en el diseño de los operadores del procesador al reducir la complejidad de la unidad aritmética y favorece la construcción de unidades aritméticas flexibles que procesen distintos tamaños de operandos. En este aspecto la aplicación de métodos iterativos permitirá procesar progresivamente todas las cifras de los operandos, por ejemplo en una unidad de suma para números de distinta longitud [García et al, 2003a], [García et al, 2003b], [Mora, 2001].

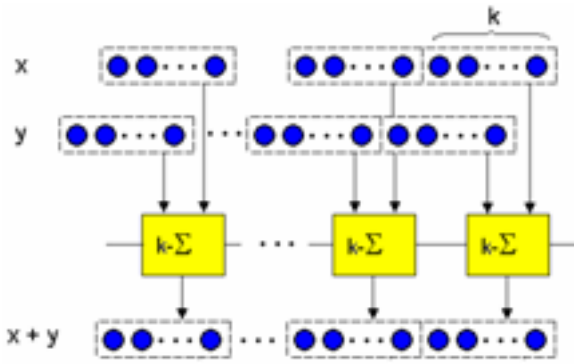


Figura 3-3: Suma CPA en bloques de k bits

Otra ventaja consiste en la mejor predisposición de la unidad para construir operadores paralelos que calculen simultáneamente varias operaciones autónomas similares. La disposición de cauces de entrada independientes y la lógica de conexión adecuada proporcionará un elevado nivel de paralelismo sobre operandos de longitud k o múltiplo de k. Esta idea es la base en el diseño de las unidades aritméticas de los procesadores multimedia [Conte et al, 1987], [Peleg y Weiser, 1996]. Estos diseños integran varias unidades procesadoras similares e independientes que operan en paralelo sobre datos de un tamaño fijo reducido, por ejemplo, muestras de sonido (16 bit), componentes de color de un pixel (8 bit), etc.

Como se observa en la figura 3-4, un operador formado por la misma cantidad de k-operadores que en la figura 3-3 se puede configurar para realizar varias operaciones en paralelo sobre datos independientes.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101000

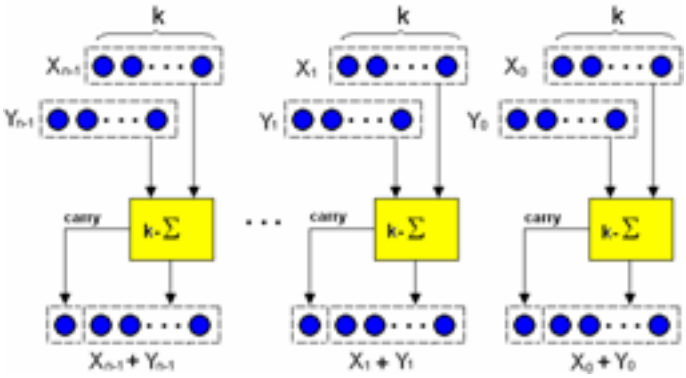


Figura 3-4: Unidad de suma con n módulos sumadores

El estudio de las operaciones de la unidad aritmética determinará aquellas que admiten su construcción mediante k-operadores que actúen sobre grupos de bits independientemente, así como su posterior combinación de los resultados correspondientes.

Para k=1 los k-operadores son idénticos a las unidades que trabajan bit a bit. Por tanto el aumento de la granularidad de un operador supone una generalización de éstos.

Diseño de los k-operadores

El objetivo de diseño de los k-operadores consiste en proporcionar una arquitectura que mejore las prestaciones del microprocesador para el mayor número de funciones. Se plantean las siguientes alternativas:

Diseño basado en lógica combinacional

Esta opción consiste en construir un k-operador mediante un circuito combinacional que calcule el resultado de la operación para k bits. Éste circuito puede estar compuesto a su vez por la conjunción de k operadores elementales bit a bit. La figura 3-5 muestra el sumador CPA con unidades k-procesadoras formadas por sumadores completos de un bit. En este caso, la interconexión entre ellos será la misma que la que se aplica entre los propios k-operadores.

11,0010010000111111010101000100010000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010100100000010010011100000100010001010011001111110011000111010 0

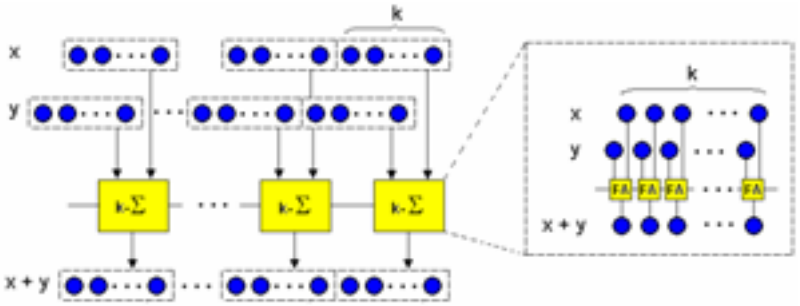


Figura 3-5: Diseño combinacional de los k-operadores

Aunque este diseño siempre será posible de realizar, no garantiza ventajas en las prestaciones de la unidad aritmética frente a las operaciones bit a bit.

Alternativamente, se plantean métodos de cálculo basados igualmente en circuitos combinacionales que obtengan el resultado completo de la función a computar. Por ejemplo para la función suma, los algoritmos de suma de anticipación de acarreo (CLA —*Carry Look Ahead Adder*) y suma condicional (COSA —*Conditional Sum Adder*) [Zimmermann, 1987]. Cada diseño posee unas características propias que pueden mejorar algunos de los aspectos de la operación que aconsejan su utilización en determinadas situaciones.

Diseño basado en lógica almacenada

Los avances en la tecnología de fabricación de circuitos y dispositivos permiten concebir nuevos métodos de operación que hubieran sido prohibitivos tiempos atrás. El procedimiento que se propone para diseñar los nuevos operadores consiste en la utilización de memorias de acceso rápido, también llamadas tablas look-up (LUT —*Look-Up Table*) como medio para realizar el cálculo efectivo.

La estructura de memoria contiene, para cualquier par de bloques de k bits, el resultado precalculado de la operación. Estas tablas look-up almacenan todos los resultados para operandos de tamaño k, de manera que tan sólo hay que seleccionar la celda en la que se encuentra el resultado en función del propio valor de los operandos sin realizar

11,00100100001111110110101001000100001011010001100001000110100110001001100011001101000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

ningún otro procesamiento más [García et al, 2003b], [Mora, 2001]. La figura 3-6 muestra esquemáticamente un k-operador suma basado en esta arquitectura.

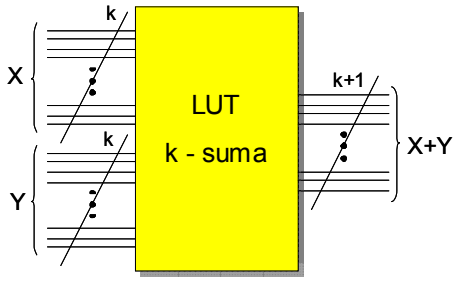


Figura 3-6: Unidad procesadora para la operación suma basada en lógica almacenada

Este procedimiento no es generalizable para todas las operaciones ni produce ventajas en todos los casos, por lo que se debe estudiar cada operación para comprobar sus beneficios. En el siguiente apartado se presenta un modelo de operación basado en este método de cálculo.

Modelo de operación basado en lógica almacenada

La idea principal consiste en implementar las operaciones del procesador mediante k-operadores construidos según un diseño de lógica almacenada.

La realización de los k-operadores mediante tablas look-up permite una mayor densidad de integración en las implementaciones VLSI que los otros métodos combinacionales de cálculo. Su utilización rebaja los costes en el desarrollo del hardware, aporta flexibilidad y reduce la cantidad de módulos requeridos en el diseño de microprocesadores. Además, estos dispositivos pueden incorporar elementos de detección de errores y corrección de los datos y, por tanto, de los resultados que se producen [Parhami, 2000].

Las posibilidades de construcción de las memorias pueden ser aprovechadas para proporcionar solidez y flexibilidad al sistema. El uso

11,0010010000111111011010100010000100010110100011000010001101001100011001100110010100010111000000011011100000111001101000100101001000000100100111000001000100010100110011111001100011101100

de memorias de sólo lectura es una opción más robusta que los circuitos combinacionales y, alternativamente, el uso de memorias de lectura y escritura permite configurar varias funciones distintas en el mismo circuito lógico facilitando su mantenimiento y reparación [Wong y Goto, 1995], [Tang, 1991].

La naturaleza de los dispositivos de memoria facilita tanto su reutilización como un alto grado de paralelismo. Memorias multipuerto con varios canales de acceso obtienen cauces de resultados paralelos en la misma pastilla de almacenamiento. En este sentido se pueden tomar decisiones de diseño y escoger elementos de acceso múltiple o estructuras compuestas por varios elementos similares con acceso concurrente.

En relación con el coste temporal, el retardo del circuito está formado por el tiempo necesario en acudir a la tabla y tomar el dato y depende de las vías de comunicación con la memoria, su estructura interna y la tecnología de fabricación [Nambu et al, 1998], [Wilton y Jouppi, 1994], [Wada et al, 1992]; además es constante e independiente del valor de los operandos. Si éstos tienen una longitud mayor que k se deberá tener en cuenta también el tiempo de combinación de los resultados parciales para conformar el resultado final.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111101 00

Sea:

$T(\Gamma_f^n)$: Tiempo en ejecutar la función f para operandos de tamaño n según la implementación Γ.

$T(\Delta M_f^k)$: Tiempo de respuesta de una memoria según un diseño Δ que contiene todos los resultados parciales de la operación f para operandos de tamaño k.

$T_{\Delta M}$: Tiempo de acceso a la memoria de diseño Δ.

$T(C_f^{n/k})$: Tiempo de composición de los resultados parciales de la función f para operandos de longitud n calculados en bloques de tamaño k.

Los métodos basados en resultados almacenados representan una mejora en rendimiento cuando el retardo conjunto empleado en el cálculo de la función sea inferior al de otras implementaciones de f, tal como formula la expresión siguiente.

$$T(\Delta M_f^k) + T_{\Delta M} + T(C_f^{n/k}) < T(\Gamma_f^n) \qquad [3.1]$$

Las distintas arquitecturas y tecnologías de memorias poseen expresiones de $T(\Delta M_f^k)$ diferentes. En este aspecto, el desarrollo de la tecnología juega un papel determinante en la mejora de las prestaciones y la reducción del coste. Los sucesivos avances tecnológicos permiten aumentar el tamaño de k para el cálculo de operandos cada vez mayores y disminuir el retardo de combinación $T(C_f^{n/k})$. Asimismo, la ubicación de las tablas de resultados en la propia unidad aritmética junto al resto de la lógica de la operación minimiza el coste de acceso a los datos $T_{\Delta M}$ [Carr, 1993].

La complejidad espacial de los k-operadores basados en tablas look-up crece exponencialmente con la longitud de los operandos. Esta circunstancia limita su uso para el caso general y afecta al valor deseable de k que maximiza el rendimiento, lo que da lugar a una situación de compromiso entre el valor de k y el tamaño de la memoria necesaria [García et al, 2003b], [Mora, 2001]. Una solución para reducir

11,0010010000111111011010100010000100010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010100100000010010011100000100010100100111110011000111010

el coste espacial consiste en aplicar etapas de preprocesamiento y postprocesamiento que reduzcan la cantidad de entradas necesarias, con el consiguiente aumento de la complejidad temporal.

III - 2

Operaciones aritméticas de números enteros

En los objetivos de este trabajo se encuentra la concepción de un procesador flexible que contenga una instrumentación de funciones con características de precisión variable. Estas operaciones tienen la consideración de primitivas del procesador, lo que impone severas restricciones de eficiencia y de complejidad.

La traducción de los números al esquema de representación propuesto condiciona fuertemente el diseño de las funciones. Los operandos se componen de una serie de campos de distinta longitud de tipo entero. La primera consecuencia de ello será la naturaleza de la relación entre la implementación realizada y las funciones matemáticas que definen.

El desarrollo de los operadores recoge la reflexión sobre esos aspectos y se sustenta tanto en la metodología de cálculo que se ha descrito como en la operatoria con números enteros. Por esta razón, se presenta en primera instancia los métodos de suma y producto para valores de esta naturaleza como base de las operaciones individuales entre las partes de los operandos.

11,0010010000111111011010100010001000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001000101001100111111001100011101001

Suma de números enteros

El método de cálculo de la función suma para operandos de distinta longitud consiste en un esquema iterativo que va construyendo el resultado de forma incremental en cada paso. A grandes rasgos la operación consiste en dividir los datos en partes manejables del mismo tamaño, sumar por separado cada parte y componer finalmente los resultados parciales por combinación. En este método se obtienen ventajas del aumento de granularidad de las operaciones y de la operatoria basada en LUT.

En la siguiente figura se observa el esquema de fragmentación y suma parcial de los operandos. Los números se alinean desde la derecha para hacer corresponder las cifras significativas del mismo orden de magnitud. Las posiciones que queden vacías por la izquierda se completan con una extensión del signo del número más corto.

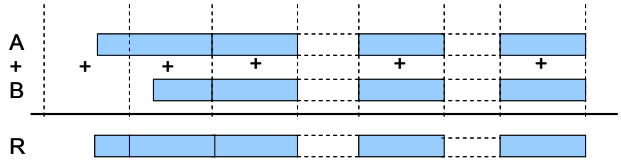


Figura 3-7: Fragmentación de los operandos y suma parcial

Las operaciones de suma de las partes se ejecutan secuencialmente comenzando desde las posiciones menos significativas. De este modo, el proceso de combinación y de formación del resultado se simplifica al considerar el acarreo serie y propagarlo de una parte hacia la siguiente siguiendo un modelo de sumador CPA. La cantidad de iteraciones necesarias para completar el procesamiento del número dependerá del tamaño de los sumandos parciales y de la longitud total de los números. Este aspecto debe buscar un equilibrio entre la complejidad espacial y temporal del operador completo intercambiando número de iteraciones por complejidad del sumador de cada parte.

El coste temporal de la operación completa depende del tiempo empleado en cada suma parcial así como del número de iteraciones que se realizan. El retardo de la operación parcial es constante debido a que

se suma una cantidad de cifras fija en cada iteración mientras que la cantidad de iteraciones es directamente proporcional a la longitud total de los sumandos. Por estos motivos, la expresión asintótica de la complejidad temporal de la operación completa es lineal con la cantidad de cifras del número mayor e independiente del método utilizado para la concatenación de las sumas parciales de bloques. Su retardo sólo influirá en la constante multiplicativa de la complejidad de la suma completa. La siguiente expresión ilustra el coste temporal,

$$T_{suma\text{-}\mathbb{Z}} \in O(n_A, n_B) \qquad [3.2]$$

donde n_A y n_B representan la cantidad de cifras de cada número respectivamente.

Suma de operandos de longitud fija

La suma de números enteros de longitud fija se puede instrumentar mediante cualquier método presente en la literatura especializada [Cheng et al, 2000], [Takagi y Horiyama, 1999], [Parhi, 1997], [Srinivas y Parhi, 1992], [Quach y Flynn, 1990], [Wei y Thompson, 1990], [Zimmermann, 1987], [Brent y Kung, 1982]. En esta memoria se propone realizar esta operación siguiendo la metodología de cálculo descrita en el apartado anterior. Esta técnica contempla un aumento de la granularidad de la operación agrupando las cifras a sumar en bloques de k dígitos y el uso de lógica almacenada.

El método que se utiliza para ejecutar la suma parcial se compone de tres etapas [García et al, 2003c], [Mora, 2001]: fragmentación de los sumandos en bloques de tamaño k, suma de todos los pares de bloques mediante acceso a una memoria con resultados almacenados y la posterior combinación ordenada de los resultados obtenidos considerando la lógica de los acarreos. La figura siguiente muestra la etapa de obtención de los resultados precalculados.

11,0010010000111111011010100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

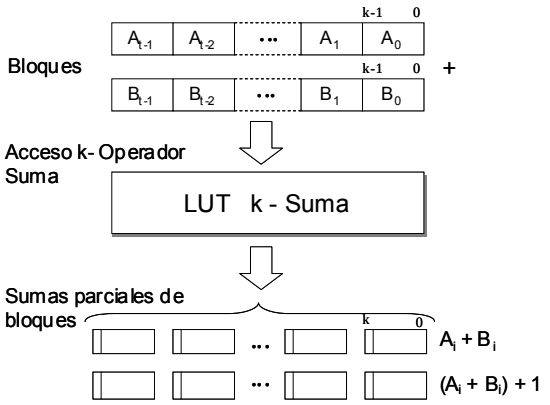


Figura 3-8: Obtención de los resultados precalculados de una suma parcial

Como se observa en la figura anterior, para cada par de bloques se extrae la suma y su sucesor de una LUT k-suma. La combinación de estas sumas parciales se realiza según una estructura que puede ser secuencial o en árbol [García et al, 2003c], [Mora, 2001]. Será decisión de diseño decidir el procedimiento a utilizar. Las figuras 3-9 y 3-10 muestran esquemáticamente el proceso de combinación en cada caso.

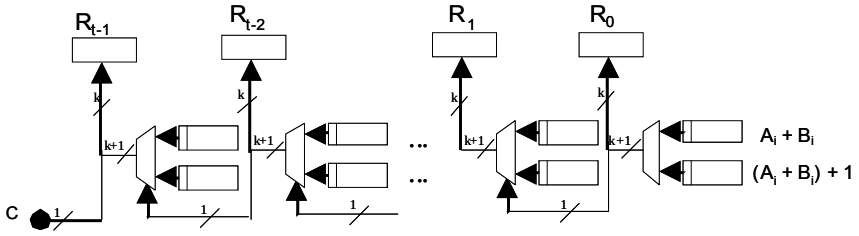


Figura 3-9: Combinación secuencial de los resultados precalculados de la suma parcial

11,00100100001111110110101010001000100001011010001100001000110101001100011001100110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011110100

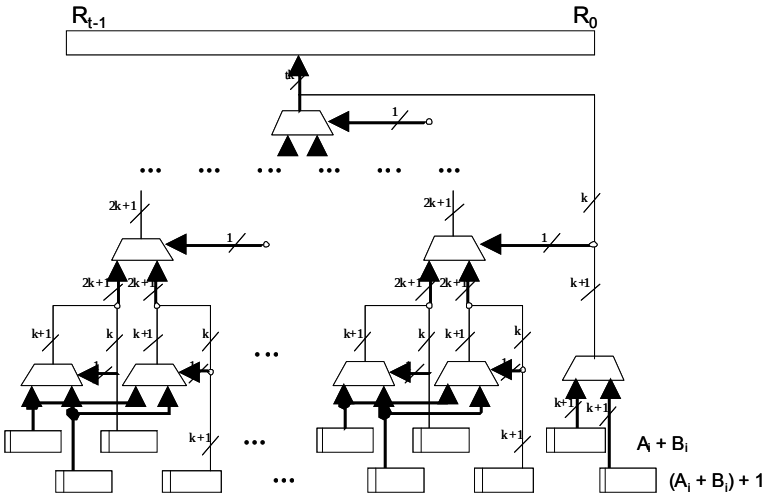


Figura 3-10: Combinación en árbol de los resultados precalculados de la suma parcial

La complejidad espacial de la operación depende en gran medida del tamaño del k-operador basado en lógica almacenada. Éste se calcula considerando las líneas de dirección de la memoria y la longitud de cada celda. Para almacenar la suma de todos los números de k cifras se necesita una memoria de $2^k \cdot 2^k \cdot (k+1)$ bits. Sin embargo, como la LUT es simétrica sólo es necesario almacenar la mitad más la diagonal principal. La siguiente expresión muestra la cantidad total de memoria necesaria.

$$M_{k\text{-suma}} = 2^{k-1} \cdot (2^k + 1) \cdot (k+1) \text{ bits} \qquad [3.3]$$

La siguiente tabla muestra la complejidad espacial del k-operador suma para distintas instancias de k.

k	$M_{k\text{ - suma}}$
1	6 bits
2	30 bits
4	680 bits
6	1,77 KB
8	36,14 KB

11,0010010000111111011010100010000100010110100011000010001101001100011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

16	≈ 4,4MB
32	>1 GB

Tabla 3-1: Tamaño del k-operador suma basado en lógica almacenada

Como se observa, la memoria aumenta de forma exponencial con el valor de k. Debido a esto, se tiene que buscar un equilibrio entre la memoria necesaria y la complicación del circuito que se requiere.

Multiplicación de números enteros

La importancia del operador de multiplicación se constata por su frecuente utilización en la mayoría de aplicaciones [Ing-Jer y Tzu-Chin, 1998], [Oberman, 1996]. A lo largo de la historia se han propuesto numerosos métodos y algoritmos de multiplicar con la finalidad de mejorar su rendimiento. Entre las propuestas más significativas se encuentran aquellas que se centran en reducir la complejidad asintótica del procedimiento de cálculo mediante la reducción sucesiva del tamaño de los operandos y el uso de la transformada rápida de Fourier (FFT – *Fast Fourier Tramsform*).

11,00100100001111110110101001000100001011010001100001000110101100010011000110010100010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

La siguiente tabla muestra algunas propuestas representativas:

Método	Complejidad
[Karatsuba y Ofman, 1963]	$O(n^{1,585})$
[Toom, 1963]	$O(n^{1,465})$
[Bailey, 1993] [Schönhage y Strassen, 1971]	$O(n \lg(n) \lg(\lg(n)))$

Tabla 3-2: Complejidad temporal de diversos algoritmos del producto

Todos esos métodos de multiplicar obtienen una mejor expresión analítica del coste que el algoritmo clásico, $O(n^2)$ [Parhami, 2000], [Bewick y Flynn, 1992]; sin embargo, poseen una excesiva complejidad de su metodología de cálculo que los convierten en difíciles de implementar en un computador. El estudio realizado por Zuras [Zuras, 1994] demuestra que, tras su implementación hardware, los métodos rápidos anteriores no proporcionan ventajas en el rendimiento, ya que la complejidad del circuito resultante provoca unos retardos que no compensan la mejora de la expresión asintótica [Zuras, 1994].

Por esas consideraciones se toma como punto de partida el método clásico de multiplicar adaptándolo a operandos de longitud variable. La operación clásica se compone de las conocidas etapas de generación y reducción de productos parciales y suma final [Wen-Chang y Chein-Wei, 2000], [Oberman, 1996], [Bewick, 1994]. El algoritmo que se utiliza aplica un esquema iterativo para construir el resultado progresivamente mediante una multiplicación por columnas [Chen-Ying, 1996]. De forma resumida, el procedimiento de cálculo consiste en fragmentar los operandos en partes manejables, multiplicar cada una de esas porciones y combinar sucesivamente los resultados parciales para obtener el resultado final de la operación. La cantidad de partes en las que se fragmentan los operandos determina el número de multiplicaciones y sumas parciales según una relación cuadrática.

La figura 3-11 muestra de forma esquemática el proceso de multiplicación, donde se observa la formación del resultado desde las

11,0010010000111111010101000100010000101101000110000100011010011000100011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101000

posiciones menos significativas. En el ejemplo se considera, sin pérdida de generalidad, que ambos operandos tienen el mismo número de cifras.

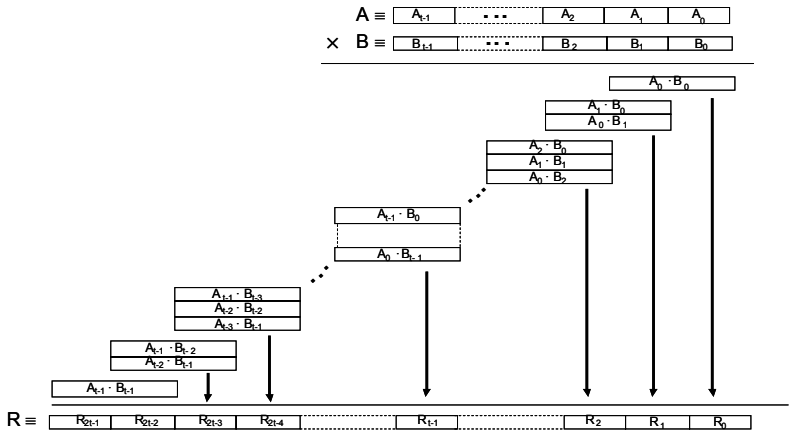


Figura 3-11: Multiplicación por columnas

Cada uno de estos productos parciales se suma al resultado parcial hasta el momento de la operación completa teniendo en cuenta el desplazamiento a la izquierda relativo a su posición. Tras cada operación, un conjunto de cifras en la zona menos significativa del resultado final son correctas. La figura 3-12 muestra la formación progresiva del resultado final en cada producto parcial.

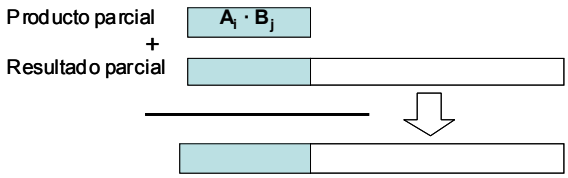


Figura 3-12: Formación del resultado final con la suma de productos parciales

La implementación de este proceso de formación del resultado debe considerar algunos aspectos de diseño con implicaciones en el rendimiento del circuito resultante en relación con el esquema iterativo y al grado de paralelismo de su implementación.

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010100

La longitud de los operandos iniciales y la longitud de los factores de las operaciones parciales son determinantes para establecer la cantidad de iteraciones necesarias. La siguiente figura muestra dos ejemplos de operandos del mismo tamaño fragmentados en distintas partes donde se observa las diferencias en la cantidad de operaciones necesarias.

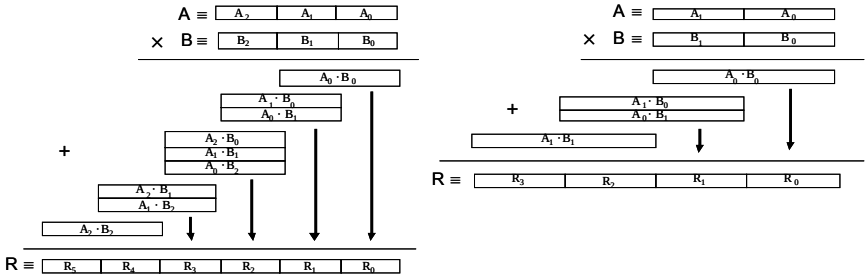


Figura 3-13: Esquema de cálculo de la multiplicación con operandos fragmentados en 2 y 3 partes

Con respecto a su implementación, se favorece la construcción de cauces segmentados entre las operaciones. Se presentan múltiples posibilidades de diseño, por ejemplo, la siguiente figura muestra estructuras segmentadas entre las operaciones de suma y producto que intervienen en la formación del resultado.

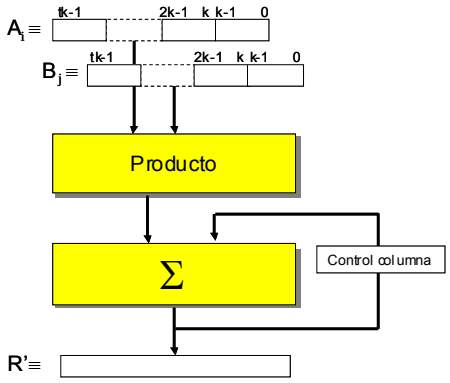


Figura 3-14: Cadena segmentada entra las operaciones de producto y suma

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110100101000100001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

Con respecto a la complejidad temporal de la operación, las decisiones de diseño se reflejan en las constantes multiplicativas de la expresión del coste pero no reducen el orden de complejidad de la operación. La expresión asintótica del coste mantiene una relación cuadrática con la cantidad de cifras de los operandos.

$$T_{\text{Multiplicación-}\mathbb{Z}} \in O(n_A \cdot n_B) \qquad [3.4]$$

Siendo n_A y n_B la cantidad de cifras de cada uno de los números enteros.

Multiplicación de operandos de longitud fija

Se conocen multitud de algoritmos para realizar la multiplicación de números de longitud fija [Parhami, 2000], [Oberman, 1996], [Bewick, 1994], [Omondi, 1994]. El procedimiento que se utiliza en esta memoria pone en práctica las ideas relativas a la metodología de operación propuesta y consta de las mismas tres etapas que la operación completa. Como se observa en la figura siguiente, en cada multiplicación se multiplican dos conjuntos de t bloques de k cifras.

11,0010010000111111011010100010001000010110100011000010001101001100010011001100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

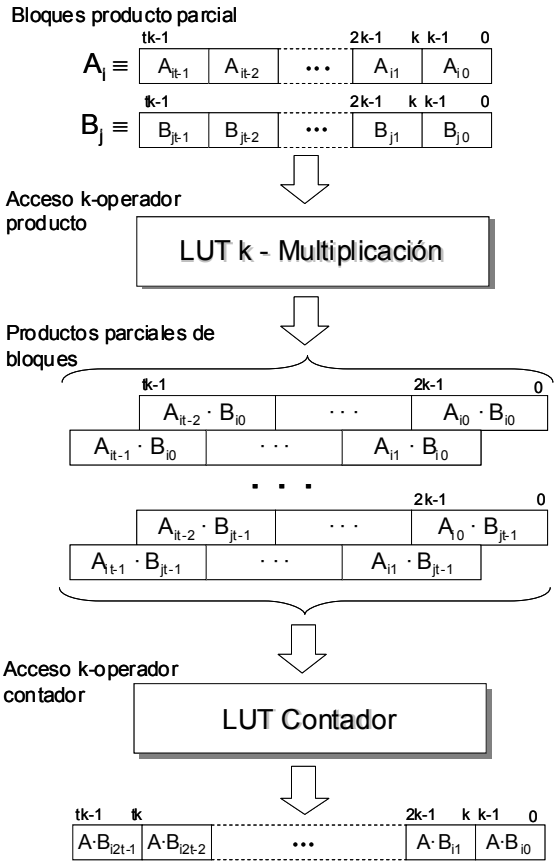


Figura 3-15: Cálculo de los productos parciales

La generación de productos parciales se realiza tomando el resultado de la operación directamente de una LUT para bloques de cifras de tamaño k, la combinación de los productos parciales reduce su número hasta un total de dos mediante el empleo de etapas de *operadores contadores* [García et al, 2003a], [Mora, 2001], [Zimmermann, 1997], [Dadda, 1965], [Wallace, 1964] y la suma final se puede realizar con cualquier método de suma de números enteros.

Al igual que en la operación de suma, la complejidad espacial de la operación depende del tamaño de los operadores basados en memorias con resultados almacenados.

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100010100010111000000011011100000111001101000100101001000000100100111000001000100010010011001111100110001110100

La memoria que implementa el operador producto contiene 2k líneas de dirección para celdas de 2k bits de longitud. Además, el tamaño total necesario tan sólo debe considerar la mitad de la tabla y la diagonal principal, por tanto, la expresión resultante de su tamaño corresponde con la siguiente:

$$M_{k\text{-multiplicación}} = (2^k + 1)\cdot 2^k \cdot k \text{ bits} \qquad [3.5]$$

En la tabla siguiente se muestra el tamaño del k-operador producto para diferentes valores de k.

k	$M_{k\text{ - multiplicación}}$
2	40 bits
4	136 B
6	3,04 KB
8	64,25 KB
12	≈ 24,5MB

Tabla 3-3: Tamaño k-operador producto basado en lógica almacenada

Para el operador contador se dispone de una memoria que contiene la cuenta del número de unos que hay entre los bits de la entrada. Para una cantidad de k líneas de entrada el número de unos se puede representar con *s* bits, siendo,

$$s = \lg_2(k+1) \qquad [3.6]$$

El tamaño total del k-operador contador es:

$$M_{k\text{-contador}} = 2^k \cdot \lg_2(k+1) \text{ bits} \qquad [3.7]$$

La tabla siguiente refleja el tamaño del operador contador para varios valores de k.

k	$M_{k\text{ - contador}}$
3	2 B

7	48 B
15	16 KB
31	>1 GB

Tabla 3-4: Tamaño k-operador contador basado en lógica almacenada

Conclusiones

En este capítulo se ha desarrollado la metodología de cálculo de las operaciones y presentado los algoritmos para la suma y el producto de operandos de tipo entero. Entre los aspectos destacables se mencionan los siguientes:

- El aumento de la granularidad de los operadores abre nuevos caminos en la concepción de procedimientos de operación y el diseño de procesadores.
- La construcción de los operadores mediante lógica almacenada proporciona ventajas inherentes a la propia estructura de la memoria relativas a flexibilidad, robustez, paralelismo y reutilización.
- La algoritmia basada en esquemas iterativos permite abordar el procesamiento de operandos de una longitud variable así como aplicar técnicas que mejoren el rendimiento y la reutilización del hardware.

Una vez presentados estos principios se está en condiciones de desarrollar algoritmia para el procesamiento de números racionales expresados en el formato propuesto. En los siguientes capítulos se proponen los métodos de cálculo de la suma y el producto racional exacto.

Capítulo IV

Suma en Precisión Variable

1. Suma de números racionales
2. Instrumentación de la función suma

IV - 1

Suma de números racionales

El desarrollo de un método de operación que obtenga el resultado exacto de la suma de dos números racionales constituye uno de los objetivos establecidos en esta investigación. La representación de los datos en el formato de doble mantisa y la consideración de elementos de gestión de la precisión de los resultados condicionan fuertemente la naturaleza de la operatoria.

En esta primera parte del capítulo se presenta el algoritmo general de suma que obtiene el resultado exacto para datos expresados en el formato propuesto sin atender las restricciones que impone el espacio material de representación y aceptando longitudes arbitrarias de los campos que conforman los números. Esta asunción permite aislar el método de cálculo de su instrumentación y logra una independencia de la capa hardware o software en la que se implemente. En el apartado siguiente, se propone una arquitectura del operador que tiene en cuenta la implementación a bajo nivel del formato de representación. En esta instrumentación entran en juego las restricciones sobre la

precisión de los operandos y resultados que se concretan en el tamaño disponible de los registros que los contienen.

La importancia de la operación de suma para el cálculo numérico centra la atención de numerosos estudios que tratan de mejorar ciertos aspectos de la misma [Bruguera y Lang, 2000], [Nielsen et al, 2000], [Bruguera y Lang, 1999], [Beaumont-Smith et al, 1998], [Suzuki et al, 1996], [Hokenek y Montoye, 1990]. La mayoría de éstos métodos de operación se rigen por un esquema en coma flotante de acuerdo con el estándar IEEE-754 [IEEE, 1985] y su clásica notación de *signo, mantisa* y *exponente*. Todos ellos indican la manera de realizar la operación mediante la manipulación de los campos que componen los números: desplazamientos, suma de mantisas, tratamiento de los exponentes, normalización, redondeo, etc.

El algoritmo que se presenta en esta memoria consiste en una extensión del método tradicional de suma en coma flotante expuesto en numerosos trabajos [Oberman, 1996], [Omondi, 1994], [Quach y Flynn, 1990], [Goldberg, 1990]. En su concepción se consideran cuestiones derivadas de la representación de los números que influyen en el diseño global de la operación, así, los sumandos y resultados se expresan de acuerdo con el esquema de representación de doble mantisa, esto es mediante *signo, exponente, mantisa fija* y *mantisa periódica*. La existencia de dos mantisas diferenciadas conlleva procedimientos distintos de cálculo cuyo resultado conformará la mantisa de la suma. Adquiere especial interés el procesamiento que atañe a la parte periódica de los números y que constituye la principal aportación en este apartado. La longitud arbitraria y no acotada de los sumandos induce a aplicar métodos iterativos para su procesamiento. A este respecto, la suma de números enteros de longitud variable que se ha descrito en el capítulo anterior juega un papel destacado en el desarrollo del cálculo para operar con los campos del número.

A lo largo de la exposición se consideran los operandos A y B según la estructura que describe la siguiente figura.

11,0010010000111111011010100010001000010110100011000010001101001100011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

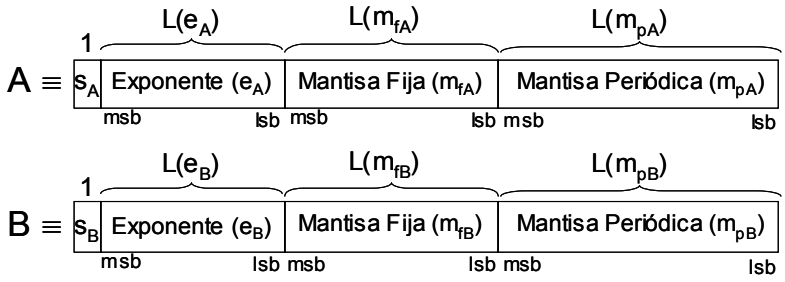


Figura 4-1: Estructura de los operandos

Teniendo en cuenta lo anterior, el algoritmo propuesto para la realización de la suma se muestra esquemáticamente en la figura siguiente:

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010100100000010010011100000100010001010011001111110011000111010 0

Figura 4-2: Etapas de la suma en coma flotante para números racionales

En esta figura se aprecian diferencias significativas en relación con el método clásico de operación. Además de las ya mencionadas, la alineación de las cifras del mismo orden de magnitud de ambas mantisas considera también la longitud variable de los operandos y la etapa de normalización realiza las acciones necesarias para adaptar los resultados al formato y evitar ambigüedades de representación. Este procedimiento de cálculo obtiene el resultado exacto de la suma para operandos racionales, por tanto, el redondeo sólo será conveniente cuando las exigencias del problema o la arquitectura establezcan un límite a la precisión de los resultados. A continuación se aborda con

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

detalle la descripción de cada una de las etapas de que consta la operación.

Cálculo del desplazamiento de mantisas

En los formatos en coma flotante de longitud de palabra fija se desplaza la mantisa del número de menor exponente hacia la derecha para considerar las cifras más significativas en el resultado de la suma. La diferencia entre los exponentes de ambos números indica la cantidad de posiciones a desplazar, tal como muestra la figura 4-3. [Oberman, 1996], [Omondi, 1994], [Quach y Flynn, 1990], [Goldberg, 1990].

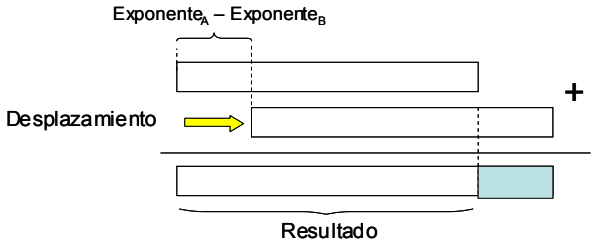
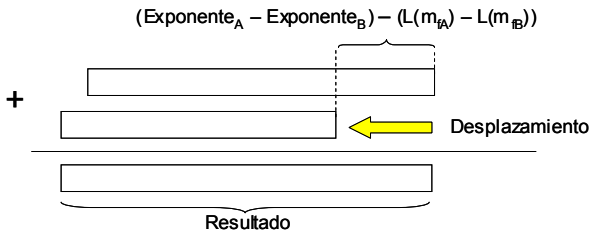


Figura 4-3: Desplazamiento de mantisas en un esquema de longitud fija

En la suma de números con una cantidad variable de cifras se debe tener en cuenta todas las cifras significativas de los sumandos para obtener el resultado exacto. Por este motivo, el cálculo del desplazamiento considera, además de la diferencia de exponentes, la relación entre la cantidad de cifras de la mantisa fija de cada número.

El procedimiento de cálculo consiste en restar la diferencia de los exponentes menos la diferencia de la cantidad de cifras significativas de la mantisa fija de ambos números, como se ilustra en la figura 4-4.



11,0010001000011111101101010001000100001011010001100001000110101100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

Figura 4-4: Desplazamiento de mantisas en un esquema de longitud variable

El campo del exponente de cada número es un valor entero de longitud variable codificado en representación sesgada. Este convenio de representación abarca a números enteros tanto positivos como negativos. Para proceder a su diferencia se realiza la suma del primero de ellos más el complemento del segundo. El sesgo de cada operando se anula y queda el valor de su diferencia expresado igualmente en complemento. El signo del resultado vendrá dado por el dígito más significativo de la operación y la existencia de acarreo.

La resta de cifras de la mantisa fija se puede procesar en paralelo con la diferencia de exponentes. En el procesamiento de datos de longitud variable es conveniente disponer del tamaño de los campos de los números para permitir su operación directa sin tener que calcular previamente su longitud. En consecuencia, el cálculo consiste en una suma en complemento de esas longitudes.

Tras la diferencia de los exponentes y de las longitudes de las mantisas se restan sus resultados para obtener la cuantía final del desplazamiento mediante otra operación similar a la primera.

Las operaciones de suma se realizan según el método de cálculo propuesto en el capítulo anterior para operandos de longitud variable. Como es sabido en la aritmética en complemento a uno, si la operación produce acarreo se debe sumar uno al resultado para obtener el valor correcto. Por este motivo, el procesamiento de los exponentes mantiene simultáneamente dos cauces de combinación de los resultados parciales para la formación del resultado: la suma y su sucesor. Esta estrategia cubre la situación de la existencia de acarreo en la última posición y evita el cálculo de una suma adicional. El resultado final se selecciona según la lógica que muestra la figura siguiente en la que interviene el acarreo de la operación.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

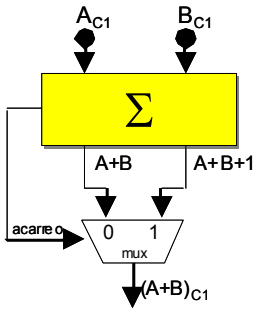


Figura 4-5: Lógica de selección del resultado de las operaciones de suma en complemento

El coste temporal de la etapa de cálculo del desplazamiento de mantisas se determina a partir de las sumas enteras de longitud variable que intervienen. El resto de operaciones de complemento y desplazamiento de bits poseen un retardo constante que no influye en la expresión asintótica del coste. Por esta razón, la complejidad temporal mantiene una relación lineal con la cantidad de cifras de los exponentes, como indica la expresión siguiente:

$$T_{Cálculo\_Desplaz\_Mantisas} \in O(L(e_A), L(e_B)) \qquad [4.1]$$

Desplazamiento de mantisas

El cometido de esta etapa es el de situar las cifras del mismo orden de magnitud de ambas mantisas en la misma posición relativa. Para ello, una de las mantisas se desplaza hacia la izquierda tantos lugares como indique el valor calculado en la etapa anterior. El signo del desplazamiento marcará el operando que se desplaza según el siguiente criterio: si es positivo se desliza A sobre B y si es negativo B sobre A.

La distinta naturaleza de las mantisas del número condiciona el tipo transformación sobre ellas. La mantisa fija se desplaza hacia la izquierda mientras que la mantisa periódica realiza una rotación en el mismo sentido.

El hueco que se produce a la derecha de la mantisa fija desplazada se completa con dígitos correspondientes a la mantisa periódica del mismo número. Si el desplazamiento fuera mayor que la cantidad de cifras de la mantisa periódica se forma un ciclo con ellas hasta completar todas las posiciones. Tras estos movimientos, la longitud de la mantisa fija aumenta una cantidad de cifras igual al desplazamiento realizado mientras que la longitud de la mantisa periódica permanece constante.

Debido a que la cuantía del desplazamiento es un valor de longitud variable hay que prestar especial atención a la suma de números de órdenes muy distintos. Esta circunstancia sugiere la consideración en esta etapa de elementos de control sobre la cantidad de cifras de los operandos.

El coste de la operación está directamente relacionado con la cantidad de posiciones a desplazar, por lo que la expresión asintótica establece una relación lineal con el tamaño de los exponentes:

$$T_{Desplaz\_Mantisas} \in O(L(e_A), L(e_B)) \qquad [4.2]$$

Suma de mantisas

La suma efectiva de números en coma flotante consiste en sumar su parte significativa. En esta etapa, el resultado de la suma de mantisas fijas y periódicas constituyen la mantisa fija y periódica respectivamente del número resultado. Ambas operaciones poseen procedimientos de cálculo distintos que admiten su cálculo en paralelo.

La gestión del signo de los operandos se realiza mediante aritmética de complemento a uno al igual que en la resta de exponentes. En este caso, el complemento de un operando afecta tanto a la mantisa fija como a la periódica. Finalmente, el signo del resultado queda en función de los operandos y del acarreo producido en las sumas fija y periódica.

Suma de mantisas fijas

Las mantisas fijas contienen una cantidad finita y conocida de cifras que, tras la etapa de desplazamiento de mantisas, se encuentran alineadas

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010100000001001001110000010001000101001100111111001100011110100

según su orden de magnitud. En caso de que las longitudes de ambos operandos no coincidan, se extiende el signo del número menor para igualar sus tamaños. La figura siguiente muestra esquemáticamente los elementos que intervienen. Para una mayor claridad de la exposición del método de cálculo, se muestra también el complemento del resultado, aunque no es necesario mantener los tres valores simultáneamente ya que cualquiera de ellos puede ser obtenido a partir de los demás.

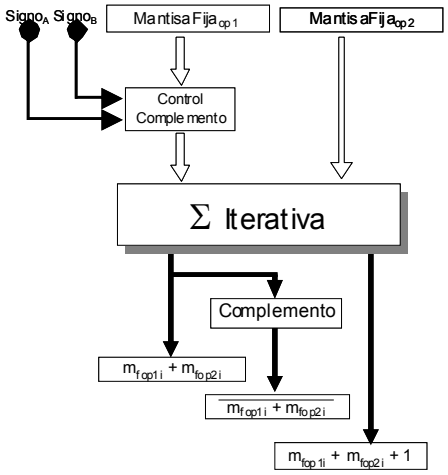
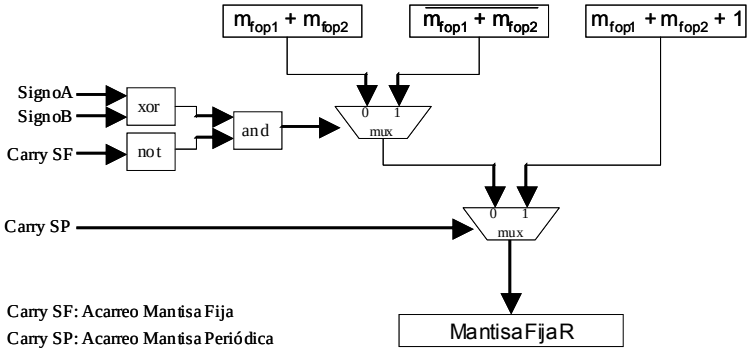


Figura 4-6: Esquema funcional de la suma de las mantisas fijas

El resultado final se selecciona según el signo inicial de los operandos y el acarreo producido en la propia suma fija y en la suma periódica, como se observa en la figura siguiente:



11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000011011100000011100110100010010100100000010010011100000100010100110011111001100011110100

Figura 4-7: Lógica de selección de la mantisa fija del resultado

La complejidad temporal de esta etapa depende linealmente de la cantidad de cifras de la mantisa fija más larga.

$$T_{Suma\_Mantisas\_Fijas} \in O(L(m_{fA}), L(m_{fB})) \quad [4.3]$$

Suma de mantisas periódicas

La mantisa periódica representa una secuencia de cifras que se repite indefinidamente. Su operación debe contemplar el ajuste previo de las longitudes de ambos sumandos y la propagación cíclica de los acarreos que se produzcan.

Debido a su naturaleza periódica se opta por la reproducción completa de los operandos para igualar sus longitudes como muestra la siguiente figura. Se puede comprobar fácilmente que esta acción aumenta el tamaño de los sumandos hasta el mínimo común múltiplo (m.c.m. —*mínimo común múltiplo*) de las cantidades de cifras de ambas mantisas.

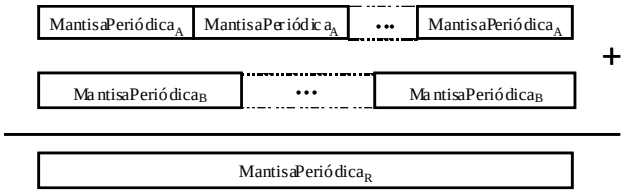


Figura 4-8: Formación de sumandos del mismo tamaño concatenando las mantisas periódicas de los operandos

La operación debe gestionar las longitudes relativas de las mantisas para cumplir la relación anterior, para ello, se establece un proceso iterativo que realiza un control dinámico de sus tamaños relativos conforme avanza el proceso de concatenación. La idea consiste en concatenar en cada paso la mantisa periódica de la cadena de menor longitud hasta que las longitudes ambas se igualen. De este modo se evita el cálculo previo del m.c.m. de las longitudes. El crecimiento del tamaño de los datos en esta etapa justifica su control para la gestión de la precisión.

11,00100100001111110110101000100010000101101000110000100011010011000100111000110001101000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

La metodología de operación que se viene utilizando en este trabajo para el procesamiento de datos de longitud variable dispone directamente del resultado de la suma y de su sucesor. La siguiente figura muestra un esquema funcional de esta etapa.

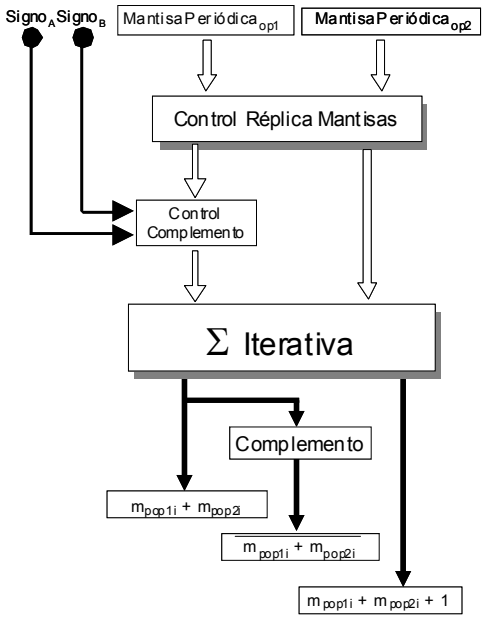


Figura 4-9: Esquema funcional de la suma de mantisas periódicas

La naturaleza periódica de los sumandos debe ser tenida en cuenta ante la presencia de acarreo, el cual se debe propagar hacia la suma de mantisas fijas y hacia la posición menos significativa del resultado. En la figura 4-10 se observa la relación entre las operaciones de las mantisas y la doble propagación del acarreo.

11,0010010000111111011010100010000100001011010001100001000110100110001001100011001101000100101000000100100111000001000100010100110011111100110001110100

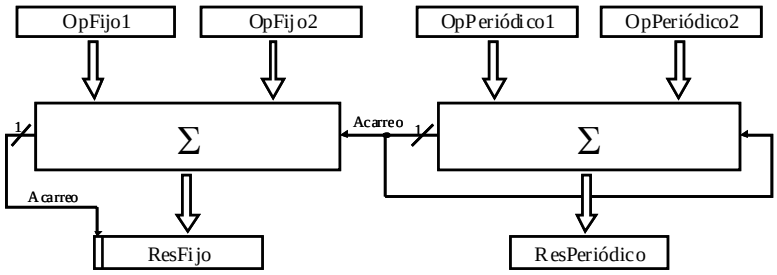


Figura 4-10: Propagación del acarreo entre mantisa periódica y fija

La selección del resultado final de la mantisa periódica se realiza en función del signo inicial de los operandos y la presencia de acarreo tanto en la suma de mantisas periódicas como en las fijas. La figura 4-11 muestra la lógica de selección del resultado según estos parámetros, donde se observa que es necesario esperar a que finalice la suma de mantisas fijas para conocer el resultado final de la mantisa periódica.

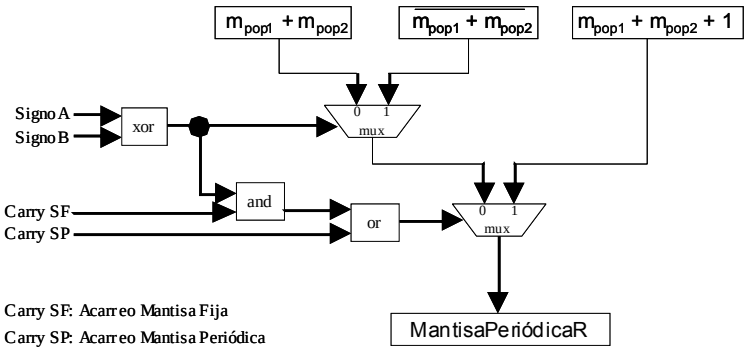


Figura 4-11: Lógica de selección de la mantisa periódica del resultado

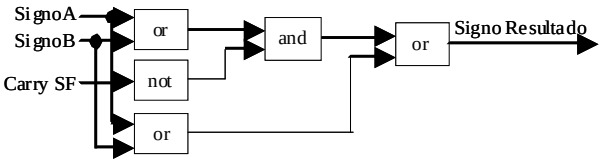
El coste temporal de la operación es directamente proporcional al mínimo común múltiplo de la cantidad de cifras de las mantisas periódicas de los operandos. Esta relación da lugar a un coste asintótico lineal con sus longitudes, como muestra la siguiente expresión:

$$T_{Suma\_Mantisas\_Periódicas} \in O(L(m_{pA}), L(m_{pB})) \qquad [4.4]$$

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

Determinación del signo

El signo del resultado está en relación con los signos iniciales de los operandos así como del acarreo producido en la suma de mantisas fijas según el procedimiento habitual para la operación de suma.



Carry SF: Acarreo de la operación de suma fija

Figura 4-12: Lógica de cálculo del signo del resultado

Su cálculo se realiza en cualquier momento tras la obtención del resultado de la mantisa fija.

Finalmente, el coste temporal total de la etapa de suma de mantisas se calcula a partir de los costes individuales de la suma de mantisas fijas y periódicas. Ambas operaciones se pueden realizar en paralelo, salvo la selección del resultado final que tiene un coste constante y despreciable frente a la propia suma. La complejidad temporal corresponde con el máximo de las complejidades anteriores y es lineal con la cantidad de cifras de las mantisas según muestra la expresión siguiente:

$$T_{Suma\_Mantisas} \in O(L(m_A), L(m_B)) \qquad [4.5]$$

Normalización

El resultado de la etapa anterior debe ajustarse a las reglas del formato de representación. En este apartado se plantea su normalización para comprobar y corregir la expresión del número y evitar ambigüedades o múltiples codificaciones del mismo valor.

Las acciones que hay que realizar consisten en situar el uno más significativo de las mantisas a la derecha de la coma fraccionaria,

actualizar consecuentemente el exponente y detectar las duplicidades de periodicidad que se pueden encontrar en las mantisas fijas y periódicas.

Detección del uno más significativo y desplazamiento de mantisas

Esta acción sitúa el uno más significativo de la mantisa a la derecha de la coma fraccionaria. Debido al carácter de longitud no fija de las mantisas, la detección y el desplazamiento emplea un procedimiento iterativo que procesa en cada paso una parte de la mantisa fija y periódica del resultado empezando desde las cifras más significativas.

El desplazamiento de la mantisa fija se realiza sin alterar la mantisa periódica. En caso de que todos los bits de la mantisa fija sean cero, el campo quedará vacío y se establecerá una longitud de campo nula. Si esto ocurre, es necesario entonces localizar el uno más significativo de la mantisa periódica y proceder a su desplazamiento. Para esta mantisa se realiza una rotación hacia la izquierda en tantas posiciones como ceros consecutivos existan en sus lugares iniciales. Las dos mantisas se pueden examinar y desplazar en paralelo escogiendo al final del proceso su valor definitivo. Si todas las cifras de ambas mantisas son cero entonces el número será cero independientemente de su exponente.

Como se observa en la figura siguiente, el desplazamiento de la mantisa fija, $desp_{MF}$, indica la cantidad de posiciones y el signo marca el sentido del movimiento: $desp_{MF}>0$ desplazamiento hacia la izquierda; $desp_{MF}<0$ desplazamiento hacia la derecha. La rotación de la mantisa periódica, siempre es hacia la izquierda, $desp_{MP}\geq 0$.

11,00100100001111110110101001000100001011010001100001000110100110001001100011001101000101000101110000000110111000001110011010001001010010000001001001110000010001 0001010011001111110011000111010 0

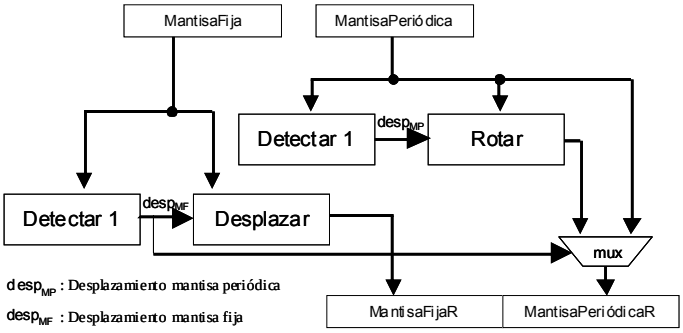


Figura 4-13: Esquema funcional del proceso de detección del uno más significativo y desplazamiento de mantisas

El retardo empleado en esta etapa está linealmente relacionado con las cifras de las mantisas de los operandos, como muestra la expresión del coste siguiente:

$$T_{Detección\_Uno} \in O(L(m_A), L(m_B)) \qquad [4.6]$$

Actualización del exponente

Posteriormente al desplazamiento de la mantisa se ajusta el exponente para reflejar el nuevo orden de magnitud del número en coma flotante. Se debe contemplar la cantidad de cifras desplazadas de la mantisa fija y, en su caso, las rotaciones realizadas en la mantisa periódica.

El valor del nuevo exponente se obtiene según el siguiente criterio:

si $(MantisaFija_R \neq 0)$ $\quad Exponente_R = Exponente_R - desp_{MF}$

si no $\quad Exponente_R = Exponente_R - L(m_{fR}) - desp_{MP}$

Esas operaciones se ejecutan mediante el esquema de suma iterativa de longitud variable debido al carácter indefinido del tamaño del exponente. Su cálculo se puede realizar en paralelo con el desplazamiento de mantisas una vez determinada la cantidad de posiciones a desplazar.

El coste de esta etapa dependerá linealmente con la cantidad de cifras del exponente.

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000011011100000111001101000100101001000000100100111000001000100010010011001111110011000111010 0

$$T_{Actualización\_Exponente} \in O(L(e_A), L(e_B)) \quad [4.7]$$

Detección de periodicidad

La operación de suma de los números puede provocar en las mantisas fijas y periódicas configuraciones de cifras inapropiadas para la correcta codificación del resultado según el formato de representación. Esta situación se produce cuando la mantisa periódica está formada por grupos de cifras que constituyen un subperiodo dentro de la propia mantisa o bien cuando la mantisa fija contiene en su extremo menos significativo algún grupo de cifras similar al de la mantisa periódica. En estos casos se debe simplificar la representación para producir una única expresión de cada número.

La mantisa periódica tiene que estar formada por el subperiodo constituyente más pequeño. Un método para comprobar la existencia de subperiodos consiste en rotar y comparar sucesivamente. Al encontrar una correspondencia, el número de cifras que se haya rotado hasta el momento indicará el tamaño final del periodo. Este proceso se implementa mediante un esquema iterativo que compara las cifras de la mantisa con las sucesivas rotaciones de la misma. Si todas las cifras del periodo son cero, se obtiene un periodo nulo y se establece el campo vacío con longitud cero. Igualmente, si todas las cifras del periodo son uno, también se elimina, pero en este caso hay que sumar una unidad a la mantisa fija en su posición menos significativa.

Con respecto a la mantisa fija, se debe eliminar del resultado las cifras redundantes ya recogidas en la mantisa periódica. El procedimiento compara la mantisa periódica con la parte final de la mantisa fija, en caso de coincidencia se elimina y se continua con el proceso de comparación.

El coste de la etapa es lineal con la longitud de la mantisa del resultado, la cual es proporcional a la cantidad de cifras de las mantisas de los operandos como indica la expresión siguiente:

$$T_{Detección\_Per} \in O(L(m_A), L(m_B)) \quad [4.8]$$

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010010000001001001110000010001000101001100111111001100011101100

El retardo total de la etapa de normalización se obtiene a partir de los costes individuales de las subtareas. La actualización del exponente se puede realizar en paralelo con la detección de periodicidad. Las expresiones [4.6], [4.7] y [4.8] muestran que el coste de esta etapa depende linealmente con la cantidad de cifras de los operandos y se rige por la siguiente expresión asintótica:

$$T_{Normalización} \in O(L(n_A),L(n_B)) \qquad [4.9]$$

donde n_A y n_B corresponden con la cantidad de cifras de cada operando respectivamente.

11,0010010000111111011010100010000100001011010001100001000110100110001001100011001100010100100000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

Coste temporal total de la operación de suma

Una vez se han descrito cada una de las etapas que componen la operación de suma para números racionales se obtiene la expresión de la complejidad temporal de la operación completa.

La organización secuencial de las etapas (figura 4-2) permite calcular el coste de la operación como la suma de los costes individuales de cada etapa.

$$T_{suma\text{-}\mathbb{Q}} = T_{Cálculo\_Desplaz\_Mantisas} + T_{Desplaz\_Mantisas} + \\ + T_{Suma\_Mantisas} + T_{Normalización} \qquad [4.10]$$

La expresión asintótica corresponde con el máximo de estos costes, que como se demuestra, pertenece a un orden superior de complejidad lineal con la cantidad de cifras de los operandos iniciales, tal y como ilustra la expresión siguiente:

$$T_{suma\text{-}\mathbb{Q}} \in O(L(n_A),L(n_B)) \qquad [4.11]$$

Con esta expresión se constata que la complejidad de la operación de suma para números racionales pertenece al mismo orden que la complejidad de la suma para números enteros a pesar del mayor número de etapas de procesamiento. La diferencia en tiempo estará en el tamaño de las constantes multiplicativas que afectan a las operaciones.

IV - 2

Instrumentación de la función suma

El procedimiento de cálculo que se ha presentado obtiene el resultado exacto de la suma de dos números codificados según el esquema de representación en coma flotante de doble mantisa. Como se ha comprobado en la instrumentación del formato, el espacio material disponible en un computador impone limitaciones a la longitud de los números, debiéndose utilizar registros de longitud fija para contenerlos. En la misma medida, la instrumentación de la función suma contempla también los requisitos para contener los resultados parciales y finales que se generan.

En línea con el planteamiento de precisión ajustable, la definición de la función suma necesita un parámetro adicional a los operandos que determina la cantidad de cifras del resultado, cumpliéndose que:

$$|\Gamma_{suma}^{VP}(\bar{x},\bar{d}) - suma(\bar{x})| \leq \varepsilon \qquad [4.12]$$

La configuración de $\bar{d}$ mantiene el significado dado en la función identidad y representa la cantidad de cifras que contiene el exponente y la mantisa en su conjunto. Su interpretación, en este contexto,

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000011011100000111001101000100101001000000100100111000001000100010100111001111100110001110100

corresponde con las limitaciones impuestas por la arquitectura del sistema y, más concretamente, por el almacenamiento de los datos.

El perfil de la función $\Gamma^{VP}_{suma}(\bar{x}, \bar{d})$ adopta la expresión siguiente:

$$\Gamma^{VP}_{suma} : \mathbb{Q} \times \mathbb{Q} \times \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{Q} \qquad [4.13]$$

donde,

$$\bar{x} \equiv (x, y) \in \mathbb{Q} \times \mathbb{Q}$$
$$\bar{d} \equiv \langle d_{L(e)}, d_{L(m)} \rangle, \text{ con } d_{L(e)} \in \mathbb{N} \wedge d_{L(m)} \in \mathbb{N}$$

Las características de expresión del formato de representación y del algoritmo propuesto en el apartado anterior garantizan que para cualquier par de sumandos racionales se puede obtener sin error el resultado de su suma con un tamaño adecuado de registro.

$$\forall x, y \in \mathbb{Q}, \exists \bar{d} \in \mathbb{N}\times\mathbb{N} \;/\; \Gamma^{VP}_{suma}(x, y, \bar{d}) = suma(x, y) \qquad [4.14]$$

Cuando la longitud de los registros no es suficiente para contener los datos se producirá un resultado aproximado, al igual que ocurre en los métodos de suma convencionales sobre operandos en coma flotante.

A continuación se describe la instrumentación a bajo nivel de la función suma, Γ^{VP}_{suma}, según los requerimientos hardware de la arquitectura del procesador y fundamentada en el algoritmo descrito anteriormente. Los operadores elementales que componen cada etapa de la suma se aplican sobre el contenido de los registros y consisten, en su mayor parte, en comparaciones, complementos, desplazamientos y rotaciones sobre los mismos. Estas operaciones, junto con algunas otras más elaboradas que intervienen, se instrumentan según los diseños comúnmente conocidos para ello. Los operadores de suma pueden construirse conforme los principios de diseño basado en lógica almacenada, lo que aporta ventajas a la instrumentación de la operación completa.

Los sumandos tienen la estructura que ilustra la figura 4-14, en la que se incluyen los punteros asociados a cada registro que delimitan el comienzo de la parte periódica así como la longitud de cada una de las mantisas del número.

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101111000000011011100000111001101000100101001000000100100111000000100010100110011111100110001110100

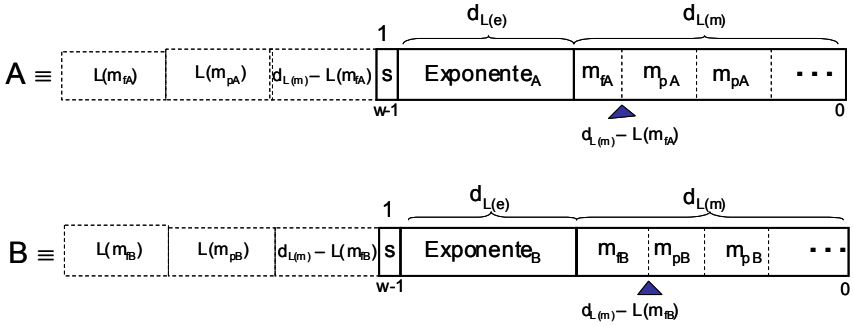


Figura 4-14: Estructura de los operandos en la instrumentación de la función

Calculo del desplazamiento de mantisas

La instrumentación de esta etapa tiene en cuenta el tamaño de los registros y de los datos. En el cálculo intervienen la longitud de las mantisas fijas y el valor de los exponentes que, en la arquitectura que se describe, están contenidos en registros de longitud fija. La siguiente figura muestra el diseño funcional de la etapa donde se observan las operaciones de suma en complemento que proveen el resultado y su sucesor así como la lógica de selección empleada.

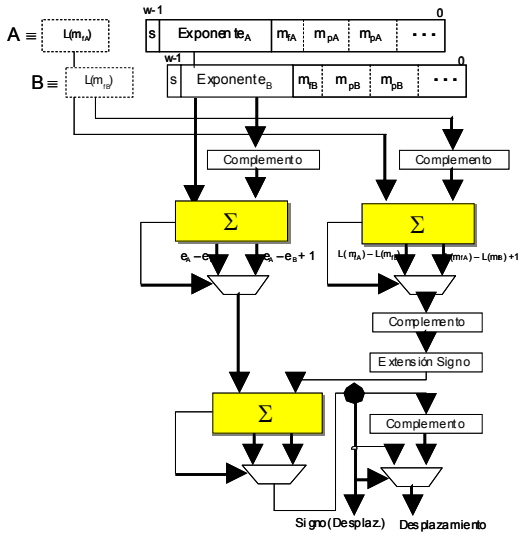


Figura 4-15: Esquema de la etapa del cálculo del desplazamiento de mantisas

Desplazamiento de mantisas

Las restricciones acerca de la cantidad de cifras de los operandos y resultados imponen un tamaño de registro fijo en las operaciones. En este contexto, las mantisas fijas se desplazan hacia la derecha para considerar las cifras más significativas.

El movimiento se realiza sobre ambos operandos y prepara las mantisas fijas para su suma eliminando las cifras periódicas de los números y situando a las mantisas en la posición menos significativa de los registros. El dato calculado en la etapa anterior corresponde con la cantidad de cifras periódicas que pasan a formar parte de la mantisa fija, lo que se traduce en menos posiciones a desplazar. Mediante multiplexores y el signo del desplazamiento se selecciona éste operando, para el que es necesario modificar la longitud de su mantisa fija y el puntero de separación entre la parte fija y periódica. Observar que la rotación del periodo es implícita a esta última operación. La figura siguiente muestra el esquema funcional de esta etapa donde se observan los detalles de la instrumentación.

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010000001001001110000010001
0001010011001111110011000111010100

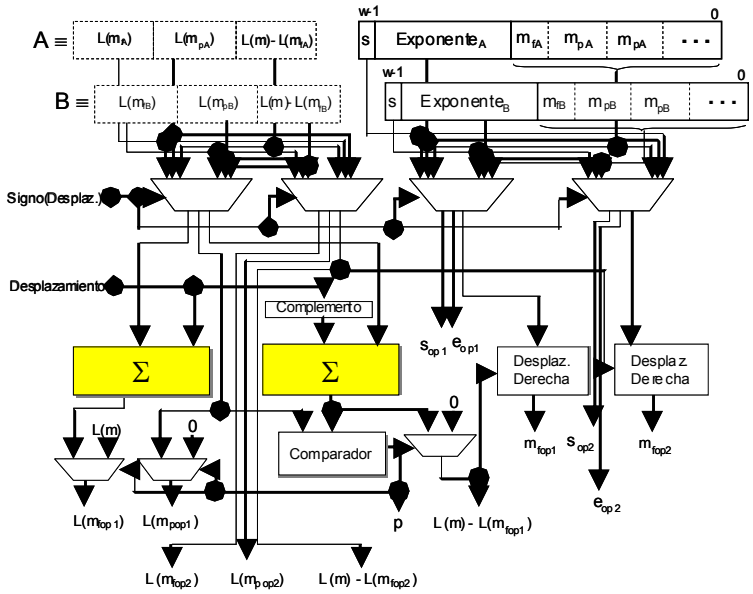


Figura 4-16: Esquema de la etapa de desplazamiento de mantisas

Las operaciones de suma que intervienen presentan una pequeña complejidad por el tamaño reducido de sus operandos. La gestión de la precisión se realiza comprobando el valor de la posición del puntero que marca la separación entre los tipos de mantisas del operando que señalado. En caso de que sea negativo se toma toda la mantisa del operando como fija y se elimina la mantisa periódica. Esta circunstancia tiene consecuencias en el resto de la operación que se ejecutará como una suma en coma flotante clásica obviando todo el procesamiento de la parte periódica de los sumandos. En esta situación no es posible alcanzar el resultado exacto de la suma y se obtendrá un resultado aproximado. Como se observa en la figura 4-16 se incluye un indicador en el diseño (p) que almacena el estado de la precisión de la operación (0: exacto; 1: aproximado).

Suma de mantisas

Esta etapa contempla el procesamiento de la parte fija y periódica de los sumandos. En el desplazamiento de mantisas se situaron a las mantisas

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100101000101110000000011011100000011100110100010010010000001001001110000010001
000101001100111111001100011101100

fijas listas en las posiciones menos significativas de registros auxiliares. A partir de ahí, su suma se realiza sin más según la instrumentación que muestra la siguiente figura.

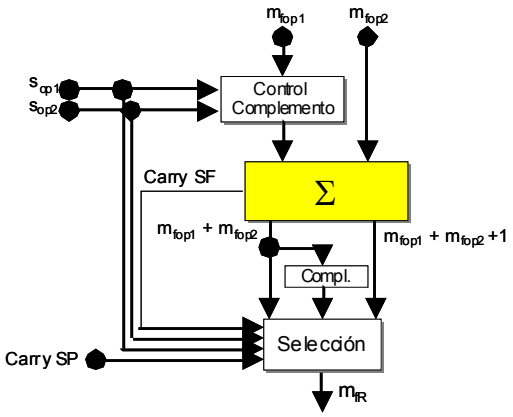


Figura 4-17: Esquema de la etapa de suma de mantisas fijas

La selección del resultado se realiza conforme la lógica descrita en la figura 4-7 en la que intervienen el signo de los operandos y el acarreo producido en la propia suma de mantisas fijas y periódicas.

Con respecto a la suma de mantisas periódicas, es preciso ajustar la longitud de los sumandos mediante la concatenación sucesiva del periodo de los operandos antes de proceder a su adición. La operatoria debe gestionar la precisión de los datos y resultados para no exceder el tamaño de registro establecido, en cuyo caso se obtendrá un resultado aproximado. En la figura siguiente se muestra el detalle de la instrumentación de esta operación con los elementos necesarios para realizar estas acciones.

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

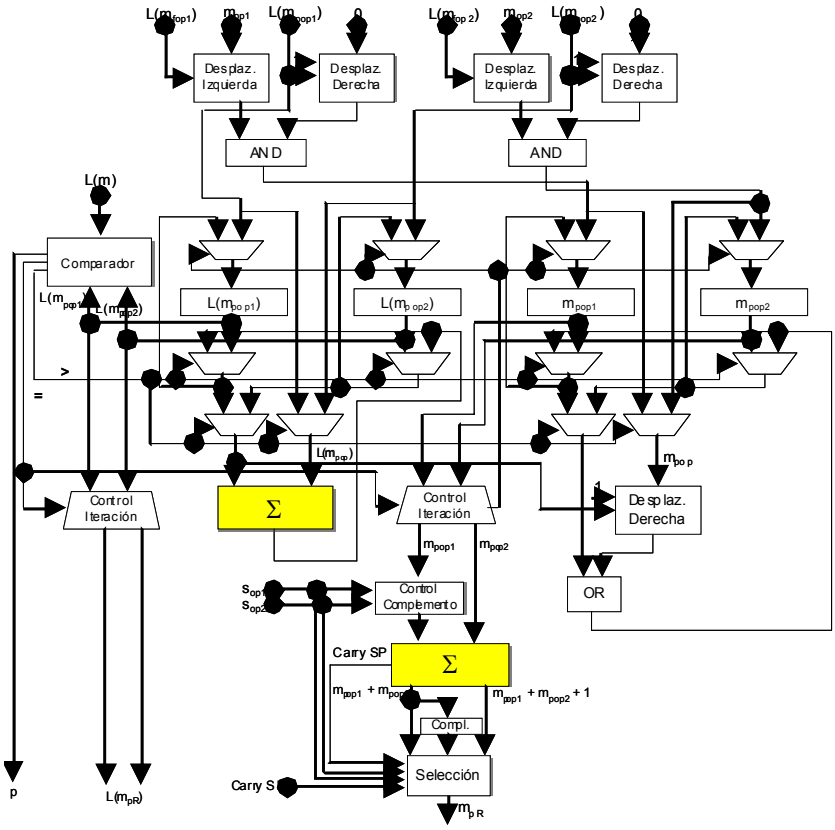


Figura 4-18: Esquema de la etapa de suma de mantisas periódicas

La instrumentación propuesta que se observa en la figura anterior separa la mantisa periódica del conjunto de la mantisa de cada operando y la coloca en las posiciones más significativas. A continuación va concatenando sucesivamente los periodos hasta que las longitudes de los sumandos se igualen. Como se aprecia en la figura 4-18, este proceso sigue un esquema iterativo en el que, en cada iteración, se comprueba sus longitudes relativas. El resultado de la comparación se señala en dos líneas de un bit (**>**, **=**): la primera de ellas informa si el primer sumando es mayor que el segundo y la segunda línea informa si ambos son iguales, en cuyo caso se detiene el proceso y se suman los datos hasta el momento. El esquema de la suma sigue el mismo

procedimiento que en la operación de mantisas fijas y es necesario su acarreo para conocer el resultado final.

La gestión de la precisión tiene un tratamiento sencillo en esta etapa. En caso de que la suma de las mantisas fijas ocupara todo el espacio disponible (p=1) el valor de la suma periódica será ignorado en la construcción del resultado. En otro caso, el control de la longitud se realiza dentro del mismo módulo comparador de longitudes en la concatenación de periodos. En cada paso comprueba que las longitudes hasta el momento no sobrepasen el tamaño establecido L(m). Si esto ocurre, se detiene el proceso y se indica, aunque no lo sean, que las longitudes de los sumandos son iguales para proceder a su suma y aprovechar su resultado posteriormente.

Normalización

La mayor parte de operadores aritméticos sobre datos en coma flotante necesitan un proceso de normalización que ajuste el resultado a los requerimientos del formato. La instrumentación que se propone contempla esta etapa en la que se incluye la actualización de las longitudes de los campos asociados a los registros que contienen los números.

Las acciones de detectar uno significativo y actualizar exponente se realizan según el método habitual para ello. Se emplea un procedimiento iterativo para la detección del periodo del número así como el refinamiento de la mantisa fija. Ésta última se compara parcialmente en sus posiciones menos significativas con el periodo y en caso de coincidencia se modifica su longitud y la posición de comienzo de la mantisa periódica. Finalmente, la formación del resultado concatena la mantisa fija y periódica de manera cíclica para ocupar toda la parte disponible del registro. El diseño general se ilustra en la figura siguiente, en la que se resaltan en áreas punteadas, cada una de las acciones que se realizan.

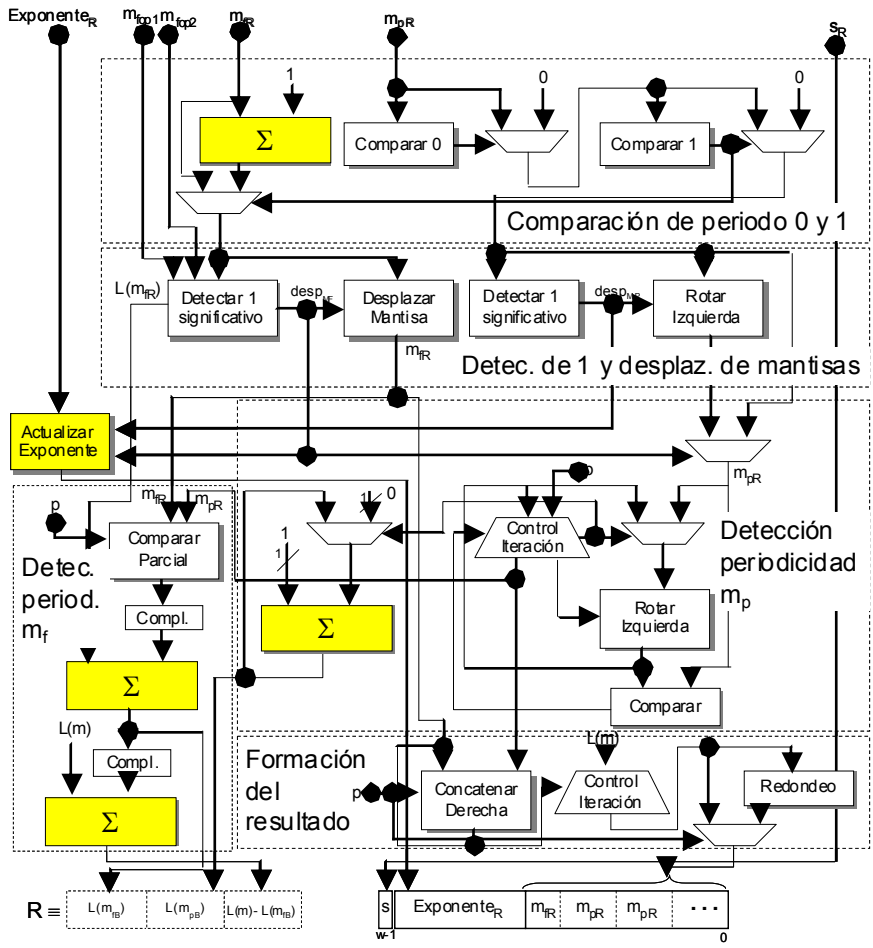


Figura 4-19: Esquema de la etapa de normalización

La gestión de la precisión tiene en cuenta el valor del indicador de exactitud p del resultado. En caso de que no sea posible la obtención de su valor exacto es necesaria una etapa de redondeo para aproximarlo al dato representable más cercano. Este procedimiento se implementa por cualquiera de los métodos conocidos y no será tratado en este trabajo.

Evaluación empírica

Los experimentos se han realizado en las mismas condiciones que las pruebas de la función identidad en lo referente al entorno de trabajo, a la generación aleatoria y a la monitorización de los resultados. Se presentan tres conjuntos de experimentos que estudian distintos aspectos de la operación y de los resultados.

Experimentos I

El objetivo de estas pruebas es el de analizar la oportunidad de la etapa de normalización en entornos donde el coste temporal sea un aspecto crítico. Los experimentos se centran en las acciones de simplificación de la mantisa periódica y en la existencia de submantisas periódicas en la mantisa fija. Se determina la frecuencia en la que estas acciones son necesarias para reducir la longitud de los resultados y evitar una múltiple representación de los números. Para comprobar la existencia de alguna relación con el rango de generación de los números racionales se consideran distintos intervalos cuya amplitud sigue una progresión geométrica de razón dos.

El perfil de este conjunto de pruebas es el siguiente:

- Suma de números racionales periódicos representados en el formato propuesto.
- Los valores pertenecen al intervalo $(0, 256]$, donde cada número se construye mediante la fracción $\frac{a}{b}$, siendo a y b números enteros aleatorios generados en los siguientes intervalos:

  $a, b \in [1..i]$, para $i = \{16, 32, 64, 128, 256\}$
- Realización de 10^6 sumas independientes en cada intervalo.

11,00100100001111110110101001000100001011010001100001000110100110001001100011001010001011100000001101110000011100110100010010010000001001001110000010001
000101001100111111001100011110100

Los resultados obtenidos se muestran en porcentaje en la tabla siguiente:

Prueba a/b	16	32	64	128	256
Simplificación de la mantisa periódica	2,65%	2,18%	1,65%	1,33%	1,11%
Existencia de submantisas periódicas en la mantisa fija	6,68%	3,14%	1,22%	0,62%	0,15%

Tabla 4-1: Frecuencia de normalización de las mantisas fijas y periódicas

El porcentaje de veces donde la etapa de normalización reduce la longitud del resultado de la suma es pequeño y su frecuencia disminuye conforme aumenta el rango de generación de los valores a sumar. Se contemplan varias estrategias para mejorar el rendimiento temporal de la operación que pasan por no realizar estos dos pasos de la normalización o realizarlos tan solo al final de una secuencia encadenada de operaciones suma.

Experimentos II

El objetivo de este conjunto de pruebas consiste en obtener una estimación de la complejidad espacial del resultado tras secuencias de operaciones encadenadas estudiando el crecimiento de la cantidad de cifras de las mantisas. Para detectar su relación con el rango de generación se consideran intervalos crecientes según una progresión aritmética de diferencia diez.

El perfil de las pruebas que se realizan es el siguiente:

- Suma de números racionales periódicos representados en el formato propuesto.

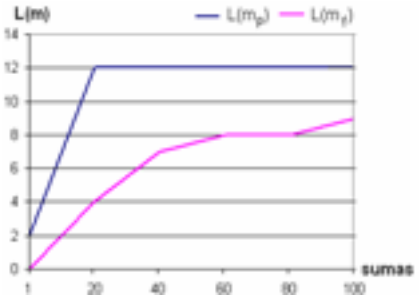
11,0010010000111111011010100010001000010110100011000010001101001100011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001000101001100111111001100011101000

- Los valores pertenecen al intervalo (0, 80], donde cada número se construye mediante la fracción $\frac{a}{b}$, siendo a y b números enteros aleatorios generados en los siguientes intervalos:

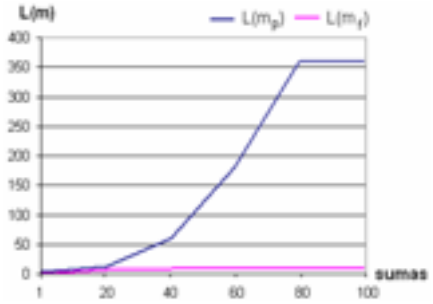
  a, b ∈ [1..i], para i = {10, 20, …, 80}

- Realización de 10^2 sumas sucesivas en cada intervalo.

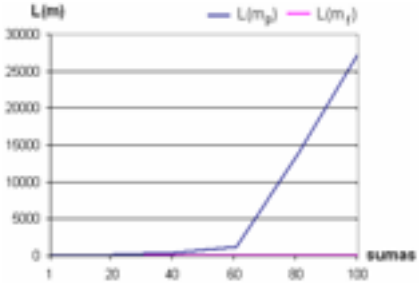
Los resultados obtenidos se muestran en las gráficas de la figura 4-20. Cada gráfica muestra el crecimiento de la mantisa fija y periódica para el intervalo de generación indicado.



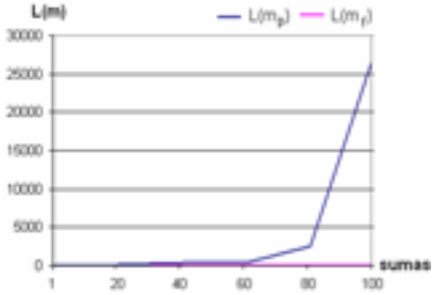
a, b ∈ [1..10]



a, b ∈ [1..20]



a, b ∈ [1..30]



a, b ∈ [1..40]

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

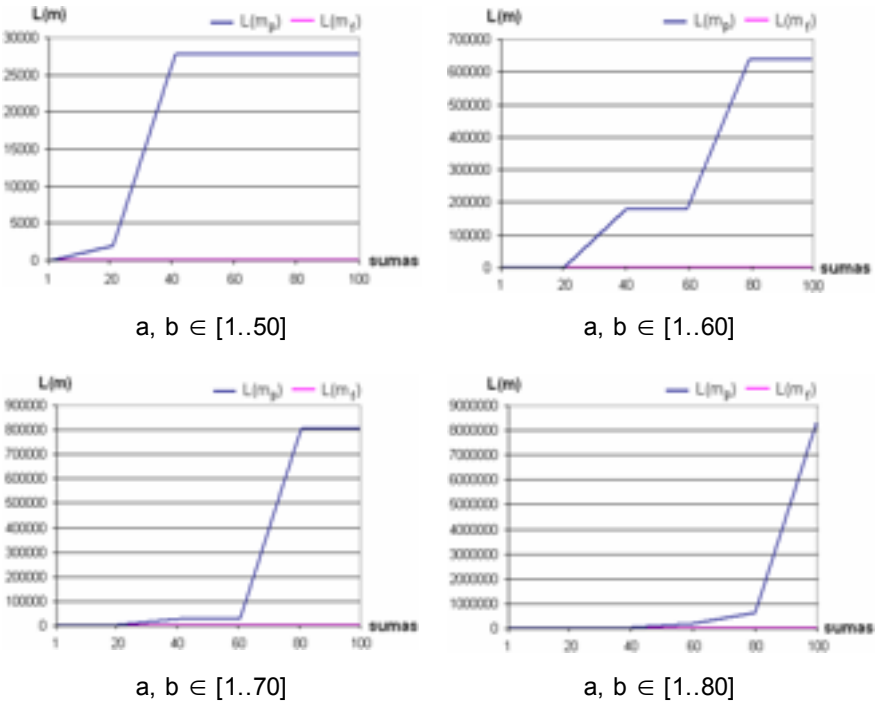


Figura 4-20: Crecimiento de la longitud de las mantisas del resultado de operaciones encadenadas

En la figura anterior se observa que el crecimiento de las mantisas fijas es mucho menor que el de las mantisas periódicas y aparentemente independiente del rango de generación de las fracciones.

Las gráficas muestran un crecimiento de las mantisas periódicas relacionado con la amplitud del rango de obtención de los denominadores. Esta correspondencia se debe al proceso de formación de la mantisa periódica en el método de suma de números racionales donde su longitud, antes de normalizar, es igual al m.c.m. de las longitudes de las mantisas periódicas de los operandos.

Aunque la cantidad de operaciones sucesivas influye en primera instancia sobre el aumento del número de cifras, existe una longitud máxima para cada intervalo que viene dada por el m.c.m. de las longitudes de los periodos de todos los números del intervalo. La tabla

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

4-2 expone estos valores límite para cada uno de los intervalos de representación de los números en las pruebas realizadas.

a/b	Factores primos	m.c.m.
a, b ∈ [1..10]	2^2 3	12
a, b ∈ [1..20]	2^3 3^2 5	360
a, b ∈ [1..30]	2^3 3^2 5 7 11	27.720
a, b ∈ [1..40]	2^3 3^2 5 7 11	27.720
a, b ∈ [1..50]	2^3 3^2 5 7 11 23	637.560
a, b ∈ [1..60]	2^3 3^2 5 7 11 13 23 29	240.360.120
a, b ∈ [1..70]	2^3 3^2 5 7 11 13 23 29	240.360.120
a, b ∈ [1..80]	2^3 3^2 5 7 11 13 23 29	240.360.120

Tabla 4-2: Límite de la longitud de la mantisa periódica

Para confirmar estos resultados se realiza otro conjunto de pruebas con una mayor cantidad de operaciones sucesivas bajo un rango de generación más reducido, cuyo perfil es el siguiente:

- Suma de números racionales periódicos representados en el modelo propuesto.
- Los valores pertenecen al intervalo (0, 40], donde cada número se construye mediante la fracción $\frac{a}{b}$, siendo a y b números enteros aleatorios generados en intervalos crecientes:

  a, b ∈ [1..i], para i = {10, 20, 30, 40}
- Realización de 10^6 sumas sucesivas en cada intervalo.

La presentación de los resultados se realiza de forma separada para las mantisas fijas y periódicas en las figuras 4-21 y 4-22 respectivamente.

Mantisas fijas:

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101 00

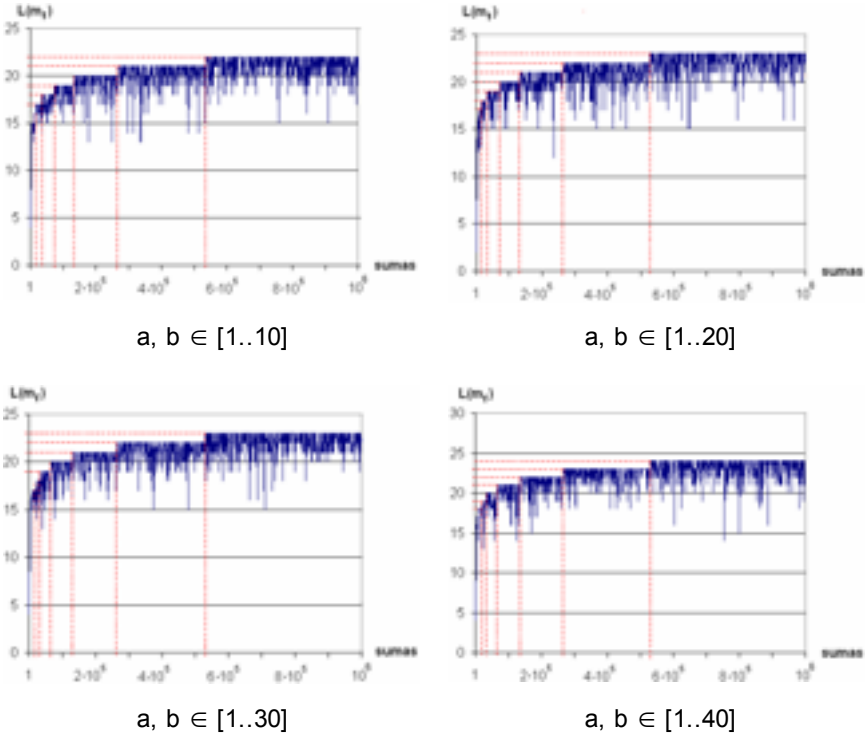


Figura 4-21: Crecimiento de la longitud de las mantisas fijas del resultado de operaciones encadenadas

En la figura anterior se observa claramente que el crecimiento de las mantisas fijas es logarítmico con la cantidad de sumas a realizar aunque no independiente del rango de generación de las fracciones. La longitud máxima de la mantisa fija para un determinado rango corresponde con la cantidad de cifras necesarias para representar en binario el extremo superior del intervalo.

Mantisas periódicas:

11,0010010000111111010101000100010000101101000110000100011010011000100111000110011000101000101110000000011011100000011100110100010010100100000010010011100000100010100110011111001100011110100

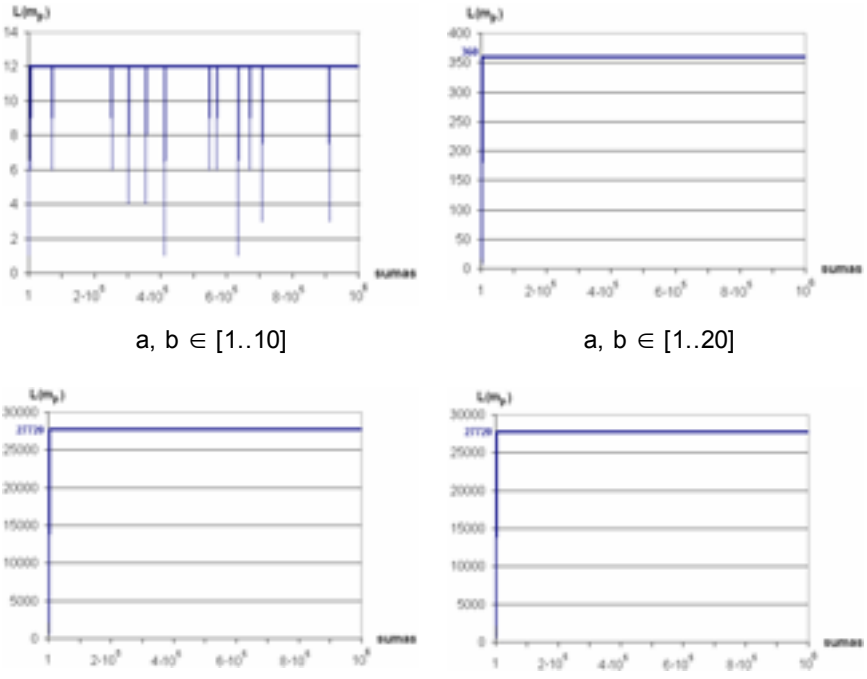


Figura 4-22: Crecimiento de la longitud de las mantisas periódicas del resultado de operaciones encadenadas

La figura 4-22 muestra el crecimiento de la mantisa periódica en extensas secuencias de operaciones consecutivas. Se observa que al alcanzar el tamaño límite el crecimiento se detiene y su longitud permanece fija a este valor salvo simplificaciones realizadas en la normalización.

Experimentos III

El objetivo de este conjunto de experimentos consiste en extraer conclusiones acerca de la calidad de los resultados de los métodos de suma convencionales basados en la norma IEEE-754. Para ello se realizan comparaciones entre los valores que proporcionan y los

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

correspondientes resultados exactos obtenidos con el algoritmo propuesto. Con objeto de estudiar únicamente la representación de la mantisa se escogen los números aleatorios en un intervalo contenido entre 0 y 1. En todos los casos el exponente se representa correctamente y en su totalidad. Se analizan múltiples escenarios de generación de los sumandos que combinan aspectos relativos al redondeo, periodo, precisión y cantidad de operaciones sucesivas.

El perfil de este conjunto de pruebas es el siguiente:

- Realización de secuencias de operaciones encadenadas mediante el método de suma propuesto para operandos racionales y mediante la suma de acuerdo con la representación IEEE-754 en simple y doble precisión. La relación entre la cantidad de sumas de cada secuencia corresponde con una progresión geométrica de razón 10 según la siguiente expresión:

$$R_{\mathbb{Q}} = \sum_{i=1}^{10^t} q_i \text{ , donde } t \in \{0..7\} \qquad [4.15]$$

- Cada número no periódico pertenece al intervalo $[0, 1)$ y consta de una cantidad de 128 cifras fraccionarias significativas aleatorias.
- Cada número periódico pertenece al intervalo $(0, 1]$ y se construye mediante la fracción $\frac{1}{b}$, donde b es un valor entero no potencia de 2 en el siguiente rango: $b \in [1..50]$
- Realización de 10^3 pruebas de cada caso.

La comparación se realiza en términos de la primera cifra diferente así como en el error absoluto que se produce en el resultado en relación con el valor exacto.

La tabla 4-3 muestra el valor promedio de la posición de la primera cifra incorrecta para números representados en simple precisión.

IEEE-754 simple precisión

11,00100100001111110110101001000100001011010001100001000110100110001001100011001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011110100

Número de operaciones	Números no periódicos		Números periódicos		Números periódicos con redondeo por exceso	
	Posición	σ	Posición	σ	Posición	σ
10^0	23,32	9,52	23,03	9,09	23,01	9,11
10^1	22,95	8,36	22,86	8,89	22,74	8,90
10^2	22,30	8,55	22,71	9,01	21,71	8,23
10^3	20,52	8,15	20,49	9,39	20,02	8,12
10^4	18,98	7,91	18,18	8,14	17,85	8,38
10^5	17,51	7,67	17,71	7,71	15,85	6,66
10^6	15,12	8,09	15,53	7,09	14,06	7,46
10^7	13,80	6,91	13,75	6,55	12,56	5,94

Tabla 4-3: Primera posición incorrecta en operaciones sucesivas con el formato IEEE-754 en simple precisión

11,00100100001111110110101000100010000101101000110000100011010011000100111000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011110100

La figura siguiente muestra la evolución de la primera cifra incorrecta con el número de sumas consecutivas.

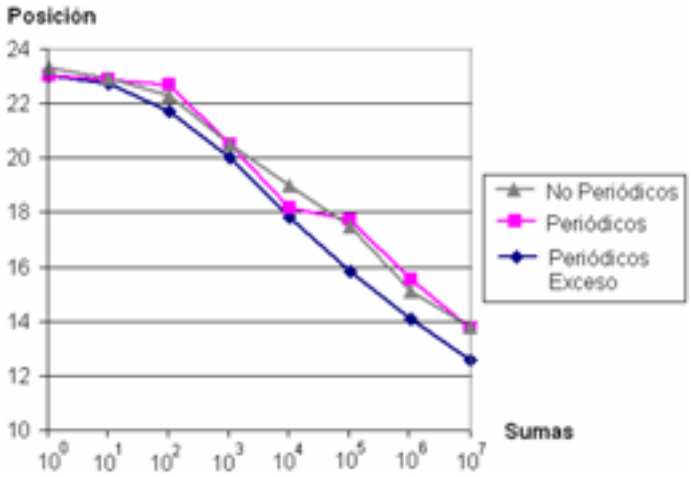


Figura 4-23: Primera posición incorrecta en operaciones sucesivas con el formato IEEE-754 en simple precisión

La tabla 4-4 muestra el valor promedio de la posición de la primera cifra incorrecta para números representados en doble precisión.

IEEE-754 doble precisión						
Número de operaciones	Números No Periódicos		Números periódicos		Números periódicos con redondeo por exceso	
	Posición	σ	Posición	σ	Posición	σ
10^0	52,12	15,59	51,75	16,44	52,02	15,44
10^1	51,95	14,76	51,96	16,31	51,76	15,91
10^2	51,07	14,55	50,98	15,02	50,53	15,02
10^3	49,42	14,80	49,21	15,21	48,12	14,83
10^4	47,57	14,03	47,39	14,85	47,31	14,36
10^5	46,48	14,72	46,68	13,97	45,15	14,02
10^6	44,08	14,58	44,23	14,69	43,56	14,12
10^7	42,23	13,48	42,28	14,01	41,17	13,55

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

Tabla 4-4: Primera posición incorrecta en operaciones sucesivas con el formato IEEE-754 en doble precisión

La figura 4-24 ilustra gráficamente los resultados de la tabla anterior.

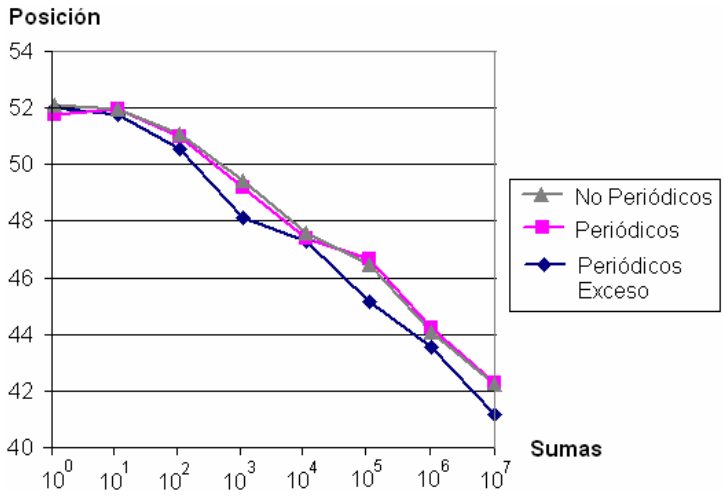


Figura 4-24: Primera posición incorrecta en operaciones sucesivas con el formato IEEE-754 en doble precisión

En las tablas 4-3 y 4-4 y las figuras 4-23 y 4-24 se observa que las operaciones sobre números no periódicos y periódicos obtienen resultados similares en los dos esquemas de precisión. Éstos últimos presentan, en general, un resultado de peor calidad en el caso de valores expresados con redondeo en exceso debido a la no compensación del error. En todos los casos la posición de la primera cifra incorrecta de la mantisa sigue una tendencia decreciente con el número de sumas consecutivas que producirán, con una cantidad suficiente de operaciones, una mantisa significativa compuesta en su totalidad por cifras incorrectas.

Las siguientes tablas y figuras muestran el valor absoluto del error cometido respecto al valor exacto del resultado.

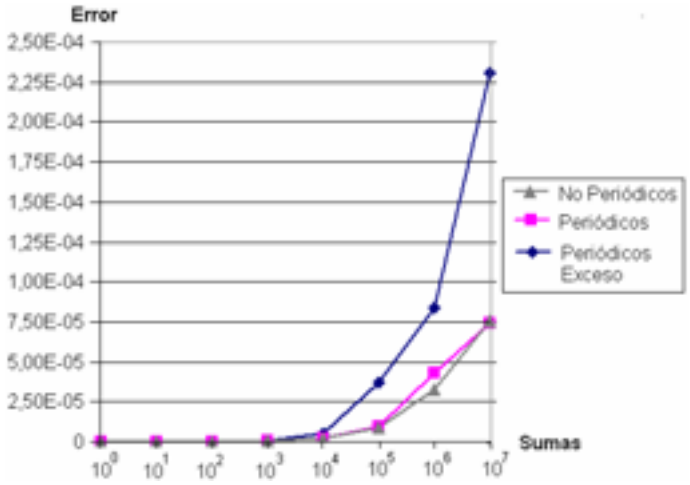
11,00100100001111110110101001000100001011010001100001000110100110001001100011001010001011100000001101110000011100110100010010010000001001001110000010001 0001010011001111110011000111010 0

Para datos en *simple precisión*:

IEEE-754 simple precisión						
Número de operaciones	Números No Periódicos		Números periódicos		Números periódicos con redondeo por exceso	
	|Error|	σ	|Error|	σ	|Error|	σ
10^0	$3{,}050 \cdot 10^{-8}$	$2{,}624 \cdot 10^{-12}$	$3{,}049 \cdot 10^{-8}$	$3{,}491 \cdot 10^{-12}$	$2{,}991 \cdot 10^{-8}$	$2{,}478 \cdot 10^{-12}$
10^1	$5{,}535 \cdot 10^{-8}$	$4{,}964 \cdot 10^{-10}$	$5{,}941 \cdot 10^{-8}$	$2{,}070 \cdot 10^{-10}$	$6{,}885 \cdot 10^{-8}$	$7{,}582 \cdot 10^{-10}$
10^2	$1{,}463 \cdot 10^{-7}$	$6{,}200 \cdot 10^{-8}$	$1{,}653 \cdot 10^{-7}$	$5{,}026 \cdot 10^{-8}$	$2{,}428 \cdot 10^{-7}$	$1{,}172 \cdot 10^{-8}$
10^3	$7{,}569 \cdot 10^{-7}$	$1{,}219 \cdot 10^{-8}$	$7{,}399 \cdot 10^{-7}$	$2{,}241 \cdot 10^{-8}$	$1{,}021 \cdot 10^{-6}$	$4{,}224 \cdot 10^{-7}$
10^4	$1{,}999 \cdot 10^{-6}$	$2{,}120 \cdot 10^{-7}$	$1{,}950 \cdot 10^{-6}$	$7{,}592 \cdot 10^{-7}$	$4{,}865 \cdot 10^{-6}$	$1{,}038 \cdot 10^{-7}$
10^5	$9{,}251 \cdot 10^{-6}$	$2{,}324 \cdot 10^{-6}$	$9{,}303 \cdot 10^{-6}$	$6{,}740 \cdot 10^{-6}$	$3{,}739 \cdot 10^{-5}$	$1{,}367 \cdot 10^{-5}$
10^6	$3{,}276 \cdot 10^{-5}$	$3{,}723 \cdot 10^{-5}$	$4{,}318 \cdot 10^{-5}$	$2{,}381 \cdot 10^{-5}$	$8{,}370 \cdot 10^{-5}$	$3{,}405 \cdot 10^{-5}$
10^7	$7{,}586 \cdot 10^{-5}$	$1{,}983 \cdot 10^{-5}$	$7{,}405 \cdot 10^{-5}$	$1{,}645 \cdot 10^{-5}$	$2{,}311 \cdot 10^{-4}$	$5{,}372 \cdot 10^{-4}$

Tabla 4-5: Error promedio en operaciones sucesivas con el formato IEEE-754 en simple precisión

La figura siguiente muestra gráficamente el crecimiento del error.



11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

Figura 4-25: Error promedio en operaciones sucesivas con el formato IEEE-754 en simple precisión

Para datos en *doble precisión*:

IEEE-754 doble precisión						
Número de operaciones	Números No Periódicos		Números periódicos		Números periódicos con redondeo por exceso	
	|Error|	σ	|Error|	σ	|Error|	σ
10^0	$5{,}805 \cdot 10^{-17}$	$7{,}470 \cdot 10^{-28}$	$5{,}671 \cdot 10^{-17}$	$7{,}470 \cdot 10^{-28}$	$5{,}830 \cdot 10^{-17}$	$4{,}375 \cdot 10^{-26}$
10^1	$1{,}012 \cdot 10^{-16}$	$1{,}086 \cdot 10^{-19}$	$1{,}100 \cdot 10^{-16}$	$1{,}086 \cdot 10^{-19}$	$1{,}138 \cdot 10^{-16}$	$2{,}709 \cdot 10^{-20}$
10^2	$2{,}727 \cdot 10^{-16}$	$2{,}911 \cdot 10^{-16}$	$2{,}973 \cdot 10^{-16}$	$2{,}911 \cdot 10^{-16}$	$5{,}994 \cdot 10^{-16}$	$4{,}529 \cdot 10^{-16}$
10^3	$1{,}046 \cdot 10^{-15}$	$1{,}092 \cdot 10^{-15}$	$1{,}180 \cdot 10^{-15}$	$1{,}092 \cdot 10^{-15}$	$2{,}401 \cdot 10^{-15}$	$2{,}911 \cdot 10^{-15}$
10^4	$3{,}680 \cdot 10^{-15}$	$2{,}771 \cdot 10^{-15}$	$3{,}277 \cdot 10^{-15}$	$2{,}771 \cdot 10^{-15}$	$8{,}152 \cdot 10^{-15}$	$5{,}528 \cdot 10^{-15}$
10^5	$8{,}663 \cdot 10^{-15}$	$1{,}103 \cdot 10^{-14}$	$7{,}969 \cdot 10^{-15}$	$1{,}103 \cdot 10^{-14}$	$3{,}604 \cdot 10^{-14}$	$3{,}121 \cdot 10^{-14}$
10^6	$5{,}512 \cdot 10^{-14}$	$6{,}739 \cdot 10^{-14}$	$3{,}835 \cdot 10^{-14}$	$6{,}739 \cdot 10^{-14}$	$9{,}823 \cdot 10^{-14}$	$2{,}316 \cdot 10^{-14}$
10^7	$2{,}460 \cdot 10^{-13}$	$3{,}157 \cdot 10^{-13}$	$1{,}753 \cdot 10^{-13}$	$3{,}157 \cdot 10^{-13}$	$7{,}981 \cdot 10^{-13}$	$1{,}932 \cdot 10^{-13}$

Tabla 4-6: Error promedio en operaciones sucesivas con el formato IEEE-754 en doble precisión

La figura 4-26 ilustra gráficamente los resultados de la tabla anterior.

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

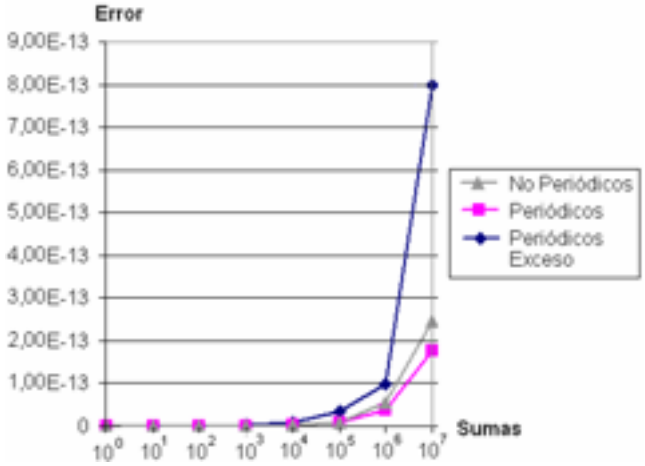


Figura 4-26: Error promedio en operaciones sucesivas con el formato IEEE-754 en doble precisión

Los resultados relativos al error absoluto corroboran las observaciones realizadas sobre la primera posición incorrecta y están en consonancia con las conclusiones de otros trabajos de estudio de las imprecisiones cometidas por el estándar de representación IEEE-754 [Goldberg, 1991], [Bohlender, 1990], [Goldberg, 1967]. El error crece a un ritmo proporcional al número de operaciones, como así lo evidencian las figuras 4-25 y 4-26. Para una mayor claridad de esta circunstancia se dubuja gráficamente el error en escala logarítmica en la figura siguiente:

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001010001011100000001101110000011100110100010010100100000010010011100000100010001010011001111110011000111010 0

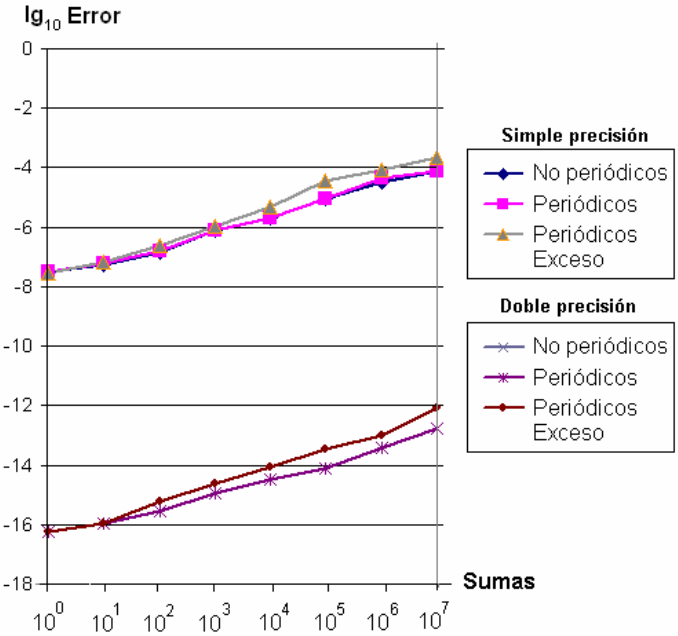


Figura 4-27: Comparación del logaritmo del error promedio de las sumas en operaciones sucesivas con el formato IEEE-754 en simple y doble precisión

Los errores que muestran las tablas y gráficos anteriores manifiestan claramente la incapacidad del formato de representación IEEE-754 de proveer resultados exactos. Si bien para una cantidad reducida de operaciones el procesamiento de datos codificados en este formato proporciona unos resultados de una calidad comparable a la propia precisión de representación de los números, queda probado que puede alcanzar cualquier grado de error con una cantidad de operaciones suficiente. Además, no hay que olvidar que las desviaciones producidas se ven amplificadas por el orden de magnitud de los números.

Conclusiones

En este capítulo se ha desarrollado un método de cálculo de la suma de números racionales del que destacan los siguientes aspectos.

- El método de operación proporciona el resultado exacto de la suma de cualquier par de números racionales representados en el formato

11,00100100001111110110101010001000100001011010001100001000110101001100010011000110010100010111000000011011100000111001101000100101001000000100100111000000100010100100110011111001100011101100

de representación propuesto, lo que constituye una evaluación exacta de la función suma para números racionales.

- El coste temporal asintótico de la operación completa y la longitud de la representación fraccionaria posicional del resultado exacto de la suma es proporcional al tamaño inicial de los operandos.

- La instrumentación se apoya en una arquitectura con registros de tamaño fijo y esquemas de cálculo iterativos. La operatoria incluye elementos de control sobre la precisión de los resultados y, en caso de una representación aproximada, es equivalente a la de los métodos en coma flotante.

- Los experimentos realizados demuestran que las operaciones de suma con números expresados según la norma IEEE producen desviaciones respecto al resultado exacto. La dimensión de estos errores aumenta proporcionalmente con la cantidad de operaciones consecutivas encadenadas que se realicen.

Capítulo V

Multiplicación en Precisión Variable

1. Multiplicación de números racionales
2. Instrumentación de la función multiplicación

V - 1

Multiplicación de números racionales

Este operador se fundamenta de forma similar al planteamiento realizado para la suma. En primer lugar se desarrolla la función de multiplicación para operandos racionales representados en el formato de doble mantisa. Posteriormente se describe pormenorizadamente la instrumentación a bajo nivel que se propone en el marco de la arquitectura que se viene utilizando en este trabajo.

El método de cálculo de la multiplicación parte del algoritmo clásico para números en coma flotante desarrollado en numerosas investigaciones [Even y Müller, 2000], [Park et al, 1999], [Altwaijry, 1997], [Oklobdzija et al, 1996], [Bewick, 1994], [Bewick y Flynn, 1992]. La operatoria incorpora las características de la representación de los números racionales así como la metodología de operación propuesta en este trabajo y se apoya en un conjunto de teoremas referentes al producto de números racionales expresados en notación fraccionaria.

A continuación se formulan y demuestran los teoremas que se aportan, los cuales permiten conocer la cantidad y naturaleza de las cifras del

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001000101001100111111001100011110100

resultado de las multiplicaciones en las que intervienen números racionales periódicos.

Teorema 1

El producto entre un número fraccionario periódico puro y un número fraccionario no periódico, ambos normalizados, presenta una dualidad con respecto a la cantidad de cifras periódicas de los números que se multiplican, siempre que la cantidad de cifras significativas de ambos números sea la misma.

$$\forall A, B \in [0, 1) \subset \mathbb{Q},\ A = 0,m_{fA}\widehat{m}_{pA},\ B = 0,m_{fB}\widehat{m}_{pB}\ /$$
$$/\ L(m_{fA})=0 \wedge m_{pA}\neq 0 \wedge L(m_{fB})>0 \wedge m_{pB} = 0 \wedge L(m_{pA}) = L(m_{fB}) \Rightarrow$$
$$\Rightarrow A \cdot B = A' \cdot B'\ /\ L(m_{fA'})= L(m_{fB}) \wedge m_{pA'} = 0 \wedge L(m_{fB'})=0 \wedge \qquad [5.1]$$
$$\wedge\ L(m_{pB'}) = L(m_{pA}) \wedge\ m_{fA'} = m_{pA} \wedge m_{pB'} = m_{fB}$$

Demostración:

$$\text{Sea } A \in [0, 1) \subset \mathbb{Q}\ /\ L(m_{fA}) = 0 \wedge m_{pA}\neq 0 \Rightarrow$$

$$\Rightarrow A = 0,\widehat{m}_{pA} = \frac{m_{pA}}{\underbrace{(B-1)\cdots(B-1)}_{L(m_{pA})}}$$

$$\text{Sea } B \in [0, 1) \subset \mathbb{Q}\ /\ L(m_{fB}) > 0 \wedge m_{pB} = 0 \Rightarrow$$

$$\Rightarrow B = 0,m_{fB} = \frac{m_{fB}}{\underbrace{10\ \cdots\ 0}_{L(m_{fB})}}$$

El producto de los números, donde $L(m_{pA}) = L(m_{fB})$, se calcula con las siguientes operaciones en notación simbólica:

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001011000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

$$A \cdot B = 0,\widehat{m}_{pA} \cdot 0,m_{fB} =$$

$$= \underbrace{\frac{m_{pA}}{(B-1)\cdots(B-1)}}_{L(m_{pA})} \cdot \underbrace{\frac{m_{fB}}{10\ \cdots\ 0}}_{L(m_{fB})} =$$

$$= \underbrace{\frac{m_{pA}}{10\ \cdots\ 0}}_{L(m_{fB})} \cdot \underbrace{\frac{m_{fB}}{(B-1)\cdots(B-1)}}_{L(m_{pA})} =$$

$$= 0,m_{pA} \cdot 0,\widehat{m}_{fB}$$

c.q.d. □

Ejemplos:

En base 10

Ej.1: $A = 0,\bar{7}\ ;\ B = 0,4 \Rightarrow A \cdot B = 0,\bar{7} \cdot 0,4 = 0,7 \cdot 0,\bar{4}$

Ej.2: $A = 0,\widehat{1038}\ ;\ B = 0,4892 \Rightarrow$

$$\Rightarrow A \cdot B = 0,\widehat{1038} \cdot 0,4892 = 0,1038 \cdot \widehat{0,4892}$$

Ej.3: $A = 0,\widehat{125}\ ;\ B = 0,5 = 0,500 \Rightarrow$

$$\Rightarrow A \cdot B = 0,\widehat{125} \cdot 0,500 = 0,125 \cdot \widehat{0,500}$$

Este teorema enuncia que la cantidad de cifras periódicas de los operandos se mantiene constante cuando la cantidad de cifras significativas de ambos es la misma independientemente del operando en el que se encuentren. Sin embargo, cuando el número de cifras significativas difiere, o bien, el número periódico contiene también mantisa fija son necesarios algunos ajustes en el valor de los números. En los siguientes ejemplos se muestra una prueba de ello.

Ej.4: $A = 0,7\widehat{391}\ ;\ B = 0,1234 \Rightarrow$

$$\Rightarrow A \cdot B = 0,7\widehat{391} \cdot 0,1234 = 0,7318 \cdot 0,\widehat{1246}$$

11,0010010000111111010101000100001000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

Ej.5: $A = 0,\overline{7}$; $B = 0,417 \Rightarrow A \cdot B = 0,\overline{7} \cdot 0,417 = 0,7 \cdot 0,\overline{463}$

Teorema 2

El producto de dos números racionales normalizados, donde un número es periódico y el otro es no periódico, contiene como máximo, la cantidad de cifras periódicas del número periódico inicial.

$$\forall\, A, B \in [0, 1) \subset \mathbb{Q} \,/\, m_{pA} \neq 0 \wedge m_{pB} = 0 \Rightarrow L(m_{pAB}) \leq L(m_{pA}) \qquad [5.2]$$

Demostración:

Sea el número racional A periódico:

$$A \in [0, 1) \subset \mathbb{Q}\ ,\ \exists\, x_a, y_a \in \mathbb{Z} \,/\, A = \frac{x_a}{y_a} \wedge mcd(x_a, y_a) = 1$$

$$L(m_{pA}) = \Im\,(y_a)$$

Sea el número racional B no periódico:

$$B \in [0, 1) \subset \mathbb{Q}\ ,\ \exists\, x_b, y_b \in \mathbb{Z} \,/\, B = \frac{x_b}{y_b} \wedge mcd(x_b, y_b) = 1$$

$$L(m_{pB}) = \Im\,(y_b)$$

donde,

mcd: Máximo común divisor.

$\Im$: Función que determina la cantidad de cifras periódicas del número.

El producto de ambos números en notación de fracción se expresa como:

$$A \cdot B = \frac{x_a}{y_a} \cdot \frac{x_b}{y_b} = \frac{1}{y_b} \frac{x_a x_b}{y_a} \qquad [5.3]$$

De la expresión [5.3] se conoce que:

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001 0001010011001111110011000111010 0

$\frac{1}{y_b}$: Es un valor no periódico.

$\frac{x_a x_b}{y_a}$: El producto $x_a x_b$ puede incorporar nuevos factores divisores de y_b.

La cantidad de cifras periódicas de $\frac{x_a x_b}{y_a}$ se determina según el siguiente criterio:

- Si $mcd(x_a x_b, y_a) = 1 \Rightarrow$ La cantidad de cifras periódicas coincide con las del número periódico inicial: $L(m_{pAB}) = L(m_{pA}) = \Im(y_a)$
- Si no si $mcd(x_a x_b, y_a) \neq 1 \Rightarrow$ La simplificación de la fracción produce un denominador y'_a, tal que, $y_a = m \cdot y'_a$, donde, de acuerdo con las reglas de la aritmética modular:

  $$y_a = m \cdot y'_a \wedge Base^s \equiv 1 \bmod (y_a) \Rightarrow$$
  $$\Rightarrow Base^s \equiv 1 \bmod (m) \wedge Base^s \equiv 1 \bmod (y'_a)$$

  por tanto, si s y s' son los menores enteros que cumplen:

  $$Base^{s'} \equiv 1 \bmod (y'_a) \wedge Base^s \equiv 1 \bmod (y_a)$$

  entonces:

  $$s' \leq s$$

luego, $L(m_{pAB}) = \Im(y'_a) \leq L(m_{pA}) = \Im(y_a)$

c.q.d. □

Ejemplos:

En base 10

Ej.1: $A = 0,\widehat{7}$; $B = 0,4 \Rightarrow A \cdot B = 0,\widehat{31}$

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

Ej.2: $A = 0,1038$; $B = 0,4892 \Rightarrow A \cdot B = 0,05078403$

Ej.3: $A = 0,\overset{\frown}{125}$; $B = 0,121 \Rightarrow A \cdot B = 0,01\overset{\frown}{515}$

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100101000100111000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

Corolario:

Si el número B no periódico tiene la forma $\frac{1}{y_b}$, es decir $x_b = 1$, entonces la cantidad de cifras periódicas del producto es igual a la cantidad de cifras del factor periódico inicial.

Demostración:

La demostración es trivial a partir del teorema anterior ya que si el numerador de la fracción, $\frac{x_a x_b}{y_a}$ no incorpora nuevos divisores de y_a, la longitud del periodo del número $A \cdot B$ será la misma que la de A.

Como se quiere demostrar. □

Teorema 3

El producto de dos números periódicos unidad es un número periódico puro cuya cantidad de cifras periódicas es múltiplo de la cantidad de cifras periódicas de cada uno de los factores.

$$\forall A, B \in [0, 1) \subset \mathbb{Q}, A = 0,\widehat{m}_{pA}, B = 0,\widehat{m}_{pB} /$$
$$/ \quad A \text{ es periódico unidad de grado } L(m_{pA}) \wedge$$
$$\wedge B \text{ es periódico unidad de grado } L(m_{pB}) \Rightarrow \qquad [5.4]$$
$$\Rightarrow A \cdot B = 0,\widehat{m}_{pAB} \wedge L(m_{pAB}) = \dot{L}(m_{pA}) \wedge L(m_{pAB}) = \dot{L}(m_{pB})$$

Demostración:

Sea $A \in [0, 1) \subset \mathbb{Q} / A \text{ es periódico unidad de grado } L(m_{pA})$,

$$\exists y_a \in \mathbb{Z} / A = \frac{1}{y_a} \wedge mcd(Base, y_a) = 1 \Rightarrow$$
$$\Rightarrow (L(m_{pA}) = s_A \Leftrightarrow (Base^{sA} \equiv 1 \bmod (y_a))$$

11,00100100001111111011010100010001000010110100011000010001101001100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

se cumple que,

$$\frac{Base^{SA}}{y_a} = C_A + \frac{1}{y_a} \;/\; C_A \in \mathbb{Z}$$

Sea $B \in [0, 1) \subset \mathbb{Q}$ / B es periódico unidad grado $L(m_{pB})$,

$$\exists\, y_b \in \mathbb{Z} \;/\; B = \frac{1}{y_b} \wedge mcd(Base, y_b) = 1 \Rightarrow$$

$$\Rightarrow (L(m_{pB}) = s_B \Leftrightarrow (Base^{sB} \equiv 1 \bmod (y_b))$$

de manera que,

$$\frac{Base^{SB}}{y_b} = C_B + \frac{1}{y_b} \;/\; C_B \in \mathbb{Z}$$

Sea R el producto de los números A y B: $R = A \cdot B$

$$R = \frac{x_a \cdot x_b}{y_a \cdot y_b}$$

y sea s_R la cantidad de cifras periódicas de R, se cumple que,

$$mcd(Base, y_a) = 1 \wedge mcd(Base, y_b) = 1 \Rightarrow mcd(Base, y_a \cdot y_b) = 1$$

La mantisa fija de la representación posicional de un número se produce por la existencia de factores comunes entre la base de la representación y el denominador de la fracción irreducible del número. De la expresión anterior se deduce que el producto es un número periódico puro. Además se tiene que:

$$s_R = L(m_{pR}) \Leftrightarrow (Base^{sR} \equiv 1 \bmod (y_a \cdot y_b))$$

$$\frac{Base^{SR}}{y_a y_b} = C_R + \frac{1}{y_a y_b} \;/\; C_R \in \mathbb{Z}$$

s_R es múltiplo de s_A y de s_B, es decir:

$$SR = \dot{SA} \;\wedge\; SR = \dot{SB}$$

11,0010010000111111011010100010001000010110100011000010001101001100011001100010100010111000000111001101000100101000000100100111000001000100010011001111110011000111010 0

Para comprobarlo se desarrolla la siguiente relación:

$$\frac{Base^{SR}}{y_a} = \frac{Base^{SR}}{y_a} \cdot \frac{y_b}{y_b} =$$

$$= y_b \cdot \frac{Base^{SR}}{y_a y_b} = y_b C_R + \frac{y_b}{y_a y_b} \quad / C_R \in \mathbb{Z}$$

Observando la expresión resultante,

$$\frac{Base^{SR}}{y_a} = y_b C_R + \frac{1}{y_a} \quad / y_b C_R \in \mathbb{Z}$$

se demuestra que la cantidad de cifras periódicas de s_R, es congruente módulo 1 con y_a:

$$Base^{sR} \equiv 1 \bmod (y_a)$$

Conforme las reglas de la aritmética modular, este hecho sólo es posible si s_R es múltiplo de s_A, como se quiere demostrar. Análogamente es posible comprobar que s_R también es múltiplo de s_B. Por tanto, queda completada la demostración del teorema. c.q.d. □

Ejemplos:

En base 2

Ej.1:

$A = 0,\widehat{001}$; $L(m_{pA}) = 3$; Periodo unidad de grado 3 ;

$B = 0,\widehat{0001}$; $L(m_{pB}) = 4$; Periodo unidad de grado 4 ;

$A \cdot B = 0,\overbrace{000000100111}$, $L(m_{pAB}) = 12$;

Ej.2:

$A = 0,\widehat{01}$; $L(m_{pA}) = 2$; Periodo unidad de grado 2 ;

$B = 0,\widehat{001}$; $L(m_{pB}) = 3$; Periodo unidad de grado 3 ;

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001000101001100111111001100011101000

$A \cdot B = 0,\overline{000011}$, $L(m_{pAB}) = 6$;

Teorema 4

Los acarreos producidos, c_n, en el desarrollo de sumas de cada conjunto de cifras periódicas del resultado del producto de dos números periódicos unidad cumplen la siguiente relación:

$$\forall\, n \in \mathbb{N},\ c_n = n \qquad [5.5]$$

Sean A y B dos números periódicos unidad. Su producto adquiere la forma siguiente:

$$A = 0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pA})} \quad ;\quad B = 0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pB})}$$

$$A \cdot B = 0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pA})} \cdot 0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pB})}$$

Debido a que cada uno de los factores periódicos se despliega en una cadena infinita de cifras fraccionarias, esta multiplicación adopta la estructura de un sumatorio infinito de productos.

$$0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pA})} \cdot 0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pB})} =$$

$$= 0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pA})} \cdot 0,\underbrace{00\cdots01}_{L(m_{pB})} +$$

$$+ 0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pA})} \cdot 0,\underbrace{00\cdots01}_{2\cdot L(m_{pB})} + \qquad [5.6]$$

$$+ 0,\underbrace{\overbrace{00\cdots01}}_{L(m_{pA})} \cdot 0,\underbrace{00\cdots01}_{3\cdot L(m_{pB})} +$$

$$\cdots$$

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

Según se pronuncia el tercer teorema, el resultado de ese sumatorio es un número periódico puro,

$$A \cdot B = R = 0,\hat{m}_p = 0,m_p\, m_p\, m_p\, m_p \ldots$$

donde la cantidad de cifras periódicas de su periodo, $L(m_p)$, es múltiplo de la cantidad de cifras periódicas de cada uno de los factores. Sea φ la cantidad de cifras periódicas del resultado: $L(m_p) = \varphi$

$$\varphi = \dot{L}(m_{pA}) \wedge \varphi = \dot{L}(m_{pB}) \qquad [5.7]$$

Cada uno de los productos del sumatorio de la expresión [5.6] corresponde con un desplazamiento del primer factor una cantidad de posiciones proporcional al grado del segundo factor, por tanto, los sumandos se pueden organizar mediante la siguiente estructura triangular.

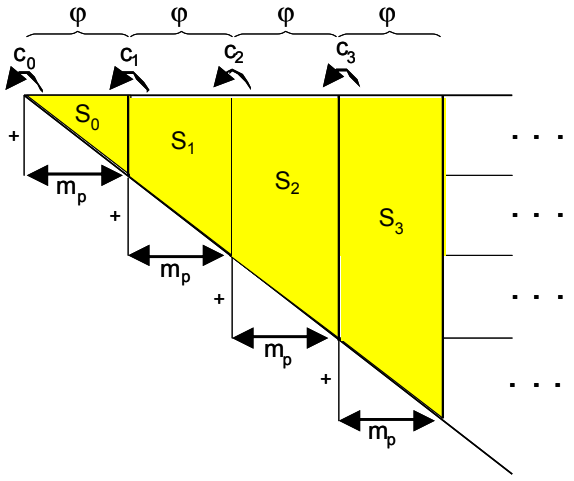


Figura 5-1: Organización de los sumandos en el desarrollo de sumas del producto de dos números periódicos

donde,

S_i : suma parcial de la columna i de φ cifras periódicas.

c_i : acarreo producido en la obtención de S_i.

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101 00

El teorema 4 formula que los acarreos c_i que se producen en la suma S_i forman una progresión aritmética de diferencia uno.

Demostración:

Se emplea el principio de inducción completa sobre preórdenes bien fundados, llamado p*rincipio de inducción Noetheriana* [Leckx y Sieber, 1984], sobre la relación [5.5].

- *Condición de aplicación del principio de inducción*: el conjunto de los números naturales con la relación de orden ≤ forman un preorden bien fundado $(\mathbb{N}, \leq)$, donde el valor cero es el minimal del conjunto.
- *Base de Inducción:* se cumple la condición para los minimales del conjunto. Se comprueba que en el primer bloque no se genera acarreo debido a que el producto de dos números periódicos puros genera un número periódico sin mantisa fija.

$$c_0 = 0 \qquad [5.8]$$

- *Hipótesis de Inducción*: se supone que para todo acarreo menor que n se cumple la relación.

$$\forall\, i \in \mathbb{N} \,/\, i<n,\ c_i = i \qquad [5.9]$$

- *Paso de Inducción*: se demuestra que se cumple la relación para n.

De la estructura formada por el desarrollo del producto de los números periódicos unidad que muestra la figura 5-1 se extrae el siguiente conjunto de ecuaciones:

$$\begin{aligned} m_p &= S_0 + c_1 - Base^{\varphi} \cdot c_0 \\ m_p &= S_1 + c_2 - Base^{\varphi} \cdot c_1 \\ &\dots \\ m_p &= S_i + c_{i+1} - Base^{\varphi} \cdot c_i \\ &\dots \\ m_p &= S_{n-1} + c_n - Base^{\varphi} \cdot c_{n-1} \\ &\dots \end{aligned} \qquad [5.10]$$

Las ecuaciones anteriores relacionan los acarreos que se producen y las sumas de cada bloque. Reorganizando los términos se obtiene:

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001

$$
\begin{aligned}
&Base^{\varphi} \cdot c_0 = S_0 + c_1 - m_p \\
&Base^{\varphi} \cdot c_1 = S_1 + c_2 - m_p \\
&\dots \\
&Base^{\varphi} \cdot c_{i-1} = S_{i-1} + c_i - m_p \\
&\dots \\
&Base^{\varphi} \cdot c_{n-1} = S_{n-1} + c_n - m_p \\
&\dots
\end{aligned}
\qquad [5.11]
$$

La disposición periódica de las cifras del resultado provoca que para cada suma S_i, excepto para la primera, se cumpla la siguiente relación:

$$S_i = S_{i-1} + T_i \qquad [5.12]$$

donde T_i es un conjunto de sumandos de φ cifras cada uno, tal y como se muestra en la siguiente figura:

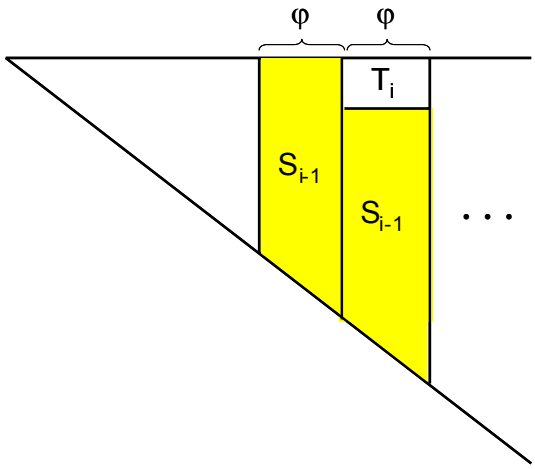


Figura 5-2: Relación entre la suma S_i y la anterior

Tomando las siguientes ecuaciones del conjunto [5.11]:

$$
\begin{aligned}
&Base^{\varphi} \cdot c_{n-2} = S_{n-2} + c_{n-1} - m_p \\
&Base^{\varphi} \cdot c_{n-1} = S_{n-1} + c_n - m_p
\end{aligned}
\qquad [5.13]
$$

Se realiza la sustitución de términos según la relación [5.12] para el índice $n-1$ en las dos ecuaciones anteriores,

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

$$S_{n-1} = S_{n-2} + T_{n-1}$$

y se obtiene:

$$\begin{aligned} Base^{\varphi} \cdot c_{n-2} &= S_{n-2} + c_{n-1} - m_p \\ Base^{\varphi} \cdot c_{n-1} &= S_{n-2} + T_{n-1} + c_n - m_p \end{aligned} \qquad [5.14]$$

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001 0001010011001111110011000111010 0

Por la *Hipótesis de Inducción*, se cumple que $\forall\, i < n\ , c_i = i$, por tanto:

$$c_{n-1} = n-1;\ c_{n-2} = n-2$$

Sustituyendo estos términos en la expresión [5.14]:

$$\begin{aligned} Base^{\varphi} \cdot (n-2) &= S_{n-2} + (n-1) - m_p \\ Base^{\varphi} \cdot (n-1) &= S_{n-2} + T_{n-1} + c_n - m_p \end{aligned} \qquad [5.15]$$

Restando ambas ecuaciones resulta:

$$-Base^{\varphi} = n-1 - c_n - T_{n-1} \qquad [5.16]$$

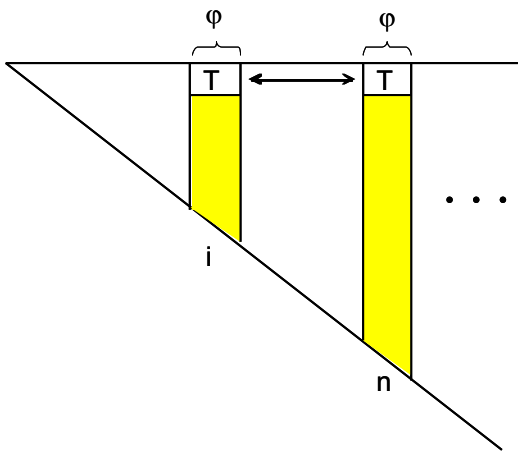
Reordenando los términos:

$$c_n = n - 1 + Base^{\varphi} - T_{n-1} \qquad [5.17]$$

Debido a la estructura periódica de los bloques del resultado, cualquier conjunto de sumandos de la estructura se repite cada cierto número de bloques de φ cifras. Esta característica da lugar a la siguiente relación:

$$\forall\, T_n\ , \exists\, 0 < i < n\ /\ T_i = T_n \qquad [5.18]$$

Es decir, el conjunto de sumandos T es el mismo cada cierto número de columnas de φ cifras. La figura siguiente ilustra esta relación:



11,00100100001111110110101001000100001011010001100001000110100110001001100011001100010100010111000000011011100000111001101000100101001000000100100111000001000100010010011001111110011000111010 0

Figura 5-3: Equivalencia entre sumas parciales

Tomando las siguientes ecuaciones del conjunto [5.11]:

$$\begin{aligned} Base^{\varphi} \cdot c_{i-2} &= S_{i-2} + c_{i-1} - m_p \\ Base^{\varphi} \cdot c_{i-1} &= S_{i-1} + c_i - m_p \end{aligned} \qquad [5.19]$$

Por la *Hipótesis de Inducción*, se cumple que $\forall\, i < n$, $c_i = i$, por tanto:

$$\begin{aligned} Base^{\varphi} \cdot (i-2) &= S_{i-2} + i - 1 - m_p \\ Base^{\varphi} \cdot (i-1) &= S_{i-1} + i - m_p \end{aligned} \qquad [5.20]$$

Sustituyendo en las ecuaciones anteriores la relación [5.12] para el índice $i-1$,

$$S_{i-1} = S_{i-2} + T_{i-1}$$

se obtiene:

$$\begin{aligned} Base^{\varphi} \cdot (i-2) &= S_{i-2} + i - 1 - m_p \\ Base^{\varphi} \cdot (i-1) &= S_{i-2} + T_{i-1} + i - m_p \end{aligned} \qquad [5.21]$$

Restando ambas ecuaciones resulta:

$$-Base^{\varphi} = -1 - T_{i-1}$$

Reordenando los términos se forma la siguiente identidad:

$$T_{i-1} = Base^{\varphi} - 1 \qquad [5.22]$$

Observar que el valor de T_{i-1} es independiente de i. Sustituyendo este resultado en la ecuación [5.17] se tiene el valor de c_n siguiente:

$$c_n = n-1 + Base^{\varphi} - T_n = n-1 + Base^{\varphi} - Base^{\varphi} + 1 = n$$

Por tanto, resulta que efectivamente:

$$\forall\, n \in \mathbb{N},\ c_n = n$$

11,0010010000111111011010100010001000010110100011000010001101001100011001100110100010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

c.q.d. □

11,0010010000111111011010100010001000101101000110000100011010011000100011100110001010001011100000001101110000011100101000100101001000000100100111000001000100010010011001111110011000111010 0

Descripción del método de multiplicar

Una vez establecida la base conceptual necesaria se está en condiciones de desarrollar el algoritmo de multiplicación propuesto. Se realizan las mismas consideraciones que en la operación de suma con respecto a la precisión de los resultados, la estructura de los operandos y el carácter variable de sus campos. En este aspecto adquiere especial importancia los algoritmos de suma y multiplicación descritos para números de longitud variable. La operación se realiza conforme a un esquema general y consta de las etapas que describe la siguiente figura.

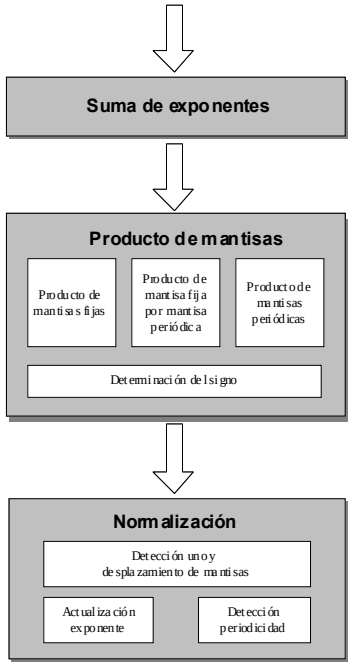


Figura 5-4: Etapas del producto en coma flotante para números racionales

El método de multiplicar que se propone presenta diferencias respecto al método clásico sobre números en coma flotante fundamentadas en el distinto formato de representación de los datos y la exactitud de los resultados. Sólo se considera su redondeo cuando las exigencias del problema impidan su representación sin error. En los siguientes apartados se detalla el procedimiento de cálculo de cada una de las

11,00100100001111110110101000100010000101101000110000100011010011000100111000110010001011100000001110011010001001010000001001001110000010001000101001100111111001100011110100

etapas de que consta la operación considerando, en toda la exposición, la estructura de los operandos que ilustra la figura 4-1.

Suma de exponentes

En esta etapa se calcula el orden de magnitud del resultado de la multiplicación mediante la suma de los exponentes de ambos operandos. Se tiene en cuenta la naturaleza de su representación para restar el sesgo al resultado que se obtiene.

El cálculo consiste en operaciones de suma entera de longitud variable que, debido a la naturaleza iterativa del algoritmo de suma, posee un retardo linealmente proporcional a la cantidad de cifras de los operandos. Asimismo, según el esquema de las etapas de la operación que muestra la figura 5-4 se admite su ejecución en paralelo con el producto de mantisas. La siguiente expresión formula la complejidad de la etapa en términos de las longitudes de los exponentes:

$$T_{Suma\_Exponentes} \in O(L(e_A), L(e_B)) \qquad [5.23]$$

Producto de mantisas

El procesamiento de esta etapa tiene en consideración la existencia de dos mantisas de distinta naturaleza. El método de cálculo que se describe se adapta a la presencia de mantisa periódica significativa y da lugar a tres posibilidades: producto de dos números fraccionarios no periódicos, producto de un número fraccionario periódico por otro no periódico y producto de dos números fraccionarios periódicos. En los siguientes subapartados se describe la operación de cada uno de los casos.

Producto de dos números no periódicos

La siguiente expresión indica las condiciones sobre la configuración de las mantisas de los números para este caso: $A, B \in \mathbb{Q}$.

11,0010010000111111010101000100010000101101000110000100011010011000100111000110011000101000101110000000011011100000111001010010010010000001001001110000010001000100100111100001100111111001100011101000100

$$\text{Sea } A = 0{,}m_{fA}\widehat{m}_{pA} \wedge m_{fA} = \alpha_{L(mfA)-1}\ldots\alpha_0 \wedge m_{pA} = \gamma_{L(mpA)-1}\ldots\gamma_0 \,/$$
$$/\ L(m_{pA}) > 0 \wedge \forall\, i \in \{0..L(m_{pA})-1\},\ \gamma_i = 0$$

$$\text{y } B = 0{,}m_{fB}\widehat{m}_{pB} \wedge m_{fB} = \alpha_{L(mfB)-1}\ldots\alpha_0 \wedge m_{pB} = \gamma_{L(mpB)-1}\ldots\gamma_0 \,/$$
$$/\ L(m_{pB}) > 0 \wedge \forall\, i \in \{0..L(m_{pB})-1\},\ \gamma_i = 0 \qquad [5.24]$$

El segundo teorema referente a la multiplicación de números racionales fraccionarios indica que el producto de dos cantidades no periódicas no contiene cifras periódicas. El resultado estará formado por el producto de las mantisas de los factores y tendrá una longitud igual a la suma de las longitudes de sus mantisas. Esta operación se realiza de acuerdo con el método de multiplicación de números enteros descrito en el capítulo tercero.

El coste del producto para este tipo de operandos pertenece a un orden de complejidad temporal equivalente al producto de la cantidad de cifras de ambos números, tal y como se dedujo en el apartado correspondiente al producto de números enteros de longitud variable. La expresión siguiente formula esta relación considerando los factores concretos de este caso. Se aprecia que si ambos operandos tienen la misma cantidad de dígitos el coste será cuadrático con el número de cifras.

$$T_{Producto\_Mantisas\_a} \in O(L(m_{fA}) \cdot L(m_{fB})) \qquad [5.25]$$

Producto de un número no periódico por otro periódico

La configuración de las mantisas en este caso se especifica en la siguiente expresión: $A, B \in \mathbb{Q}$.

$$\text{Sea } A = 0{,}m_{fA}\widehat{m}_{pA} \wedge m_{fA} = \alpha_{L(mfA)-1}\ldots\alpha_0 \wedge m_{pA} = \gamma_{L(mpA)-1}\ldots\gamma_0 \,/$$
$$/\ L(m_{pA}) > 0 \wedge \forall\, i \in \{0..L(m_{pA})-1\},\ \gamma_i = 0$$

$$\text{y } B = 0{,}m_{fB}\widehat{m}_{pB} \wedge m_{fB} = \alpha_{L(mfB)-1}\ldots\alpha_0 \wedge m_{pB} = \gamma_{L(mpB)-1}\ldots\gamma_0 \,/$$
$$/\ L(m_{pB}) > 0 \wedge \exists\, i \in \{0..L(m_{pB}) - 1\},\ \gamma_i \neq 0 \qquad [5.26]$$

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

Según el segundo teorema formulado anteriormente, el resultado del producto puede tener tantas cifras periódicas como el factor periódico que interviene en la operación. Para el desarrollo de la exposición se consideran los siguientes valores racionales:

$$A, B \in \mathbb{Q} \;/\; A = 0{,}m_{fA} \wedge B = 0{,}m_{fB}\bar{m}_{pB}$$

11,00100100001111110110101001000100001011010001100001000110100110001001100011001101000100101100001110000001101110000011100110100010010100100000010010011100000100010100110011111100110001110100

Cuyo valor simbólico de fracción corresponde a:

$$A = 0,m_{fA} = \frac{m_{fA}}{\underbrace{10\ \cdots\ 0}_{L(m_{fA})}}$$

$$B = 0,m_{fB}\widehat{m}_{pB} = \frac{m_{fB}\, m_{pB} - m_{fB}}{\underbrace{(Base-1)\cdots(Base-1)}_{L(m_{pB})}\ \underbrace{0\ \cdots\ 0}_{L(m_{fB})}}$$

Multiplicado ambos números se obtiene:

$$A \cdot B = \frac{m_{fA}}{\underbrace{10\ \cdots\ 0}_{L(m_{fA})}} \cdot \frac{m_{fB}\, m_{pB} - m_{fB}}{\underbrace{(Base-1)\cdots(Base-1)}_{L(m_{pB})}\ \underbrace{0\ \cdots\ 0}_{L(m_{fB})}} =$$

$$= \frac{m_{fA} \cdot (m_{fB}\, m_{pB} - m_{fB})}{\underbrace{10\ \cdots\ 0}_{L(m_{fA})}\ \underbrace{0\ \cdots\ 0}_{L(m_{fB})}} \cdot \frac{1}{\underbrace{(Base-1)\cdots(Base-1)}_{L(m_{pB})}} \qquad [5.27]$$

Sea C el resultado de la siguiente expresión:

$$C \equiv m_{fA} \cdot (m_{fB} m_{pB} - m_{fB}) \qquad [5.28]$$

Conocida la relación entre el número periódico unidad y la fracción que lo genera,

$$\frac{1}{B^p - 1} = \frac{1}{\underbrace{(B-1)\cdots(B-1)}_{p}} = 0,\overbrace{00\cdots01}^{p} \qquad [5.29]$$

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010100

la multiplicación entre los números fraccionarios A y B se formula según las expresiones [5.30] y [5.31].

$$A \cdot B = \frac{m_{fA} \cdot (m_{fB}\, m_{pB} - m_{fB}) \cdot Base^{-(L(m_{fA}) + L(m_{fB}))}}{(Base^{L(m_{pB})} - 1)} \qquad [5.30]$$

$$A \cdot B = C \cdot 0,\overbrace{\underbrace{00 \cdots 01}_{L(m_{pB})}} \cdot Base^{-(L(m_{fA}) + L(m_{fB}))} \qquad [5.31]$$

En la expresión [5.31] se observa que el producto se compone de tres factores: el valor C que aporta la parte significativa del resultado, el número periódico unidad de grado $L(m_{pB})$ que incorpora la componente periódica y un desplazamiento que contiene el orden de magnitud.

El producto de la componente significativa por la periódica se calcula mediante la siguiente expresión:

$$C \cdot 0,\overbrace{\underbrace{00 \cdots 01}_{L(m_{pB})}} = \frac{\left(\sum_{i=0}^{h} DesplazarDer(C,\, i \cdot L(m_{pB}))\right) + Carry_P}{\underbrace{10 \cdots 0}_{L(m_{pB})}} \qquad [5.32]$$

donde,

$DesplazarDer(C, x)$: Desplazamiento del valor C, x posiciones a la derecha.

$Carry_p$: Acarreo producido a partir de la posición $L(m_{pB})-1$ de la operación:

$$\left(\sum_{i=0}^{h} DesplazarDer(C, i \cdot L(m_{pB}))\right) \qquad [5.33]$$

h: $\lceil L(C) / L(m_{pB}) \rceil - 1$. [5.34]

La expresión [5.32] consiste en una serie de sumas consecutivas entre valores enteros correspondientes a desplazamientos progresivos del

11,0010010000111111011010100010001000100011011010001100001000110100110001100011001100010100010111000000011101100000111001101000100101001000000100100111000001000100010100111001111100110001110100

número inicial C más el acarreo producido desde la posición donde comienza el periodo. El valor C contendrá una cantidad de cifras mayor o igual a $L(m_{pB})$. Asimismo, el resultado final de la expresión será mixto y contendrá una mantisa fija y una periódica con una cantidad de dígitos limitada por el segundo teorema. El procedimiento de cálculo construye un resultado con una cantidad de cifras igual a las del factor periódico. Posteriormente se normalizará el número para eliminar redundancias en el periodo si fuera necesario. El desarrollo de la operatoria considera números racionales de longitud variable expresados en el formato de doble mantisa y cuyos campos tienen una longitud no acotada.

A continuación se describe un ejemplo que ilustra el algoritmo empleado para el cálculo de la expresión [5.32].

Sea el valor C compuesto por ocho cifras:

$$C = C_7\ C_6\ C_5\ C_4\ C_3\ C_2\ C_1\ C_0$$

Y sean dos las cifras periódicas del número periódico: $L(m_{pB}) = 2$

Para este ejemplo, la expresión [5.32] se compone de una serie de sumandos desplazados:

$$C \cdot 0,\widehat{01} = C \cdot 0,01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01 \ldots =$$
$$= C \cdot 0,01 + C \cdot 0,00\ 01 + C \cdot 0,00\ 00\ 01 \ldots$$

En la siguiente figura se dispone la suma múltiple anterior verticalmente. La zona triangular sombreada muestra gráficamente las cifras implicadas en el cálculo de su valor, donde las $L(m_{pB})$ menos significativas forman el periodo. El acarreo que se produzca en su cálculo deberá ser considerado en las cifras fijas y periódicas.

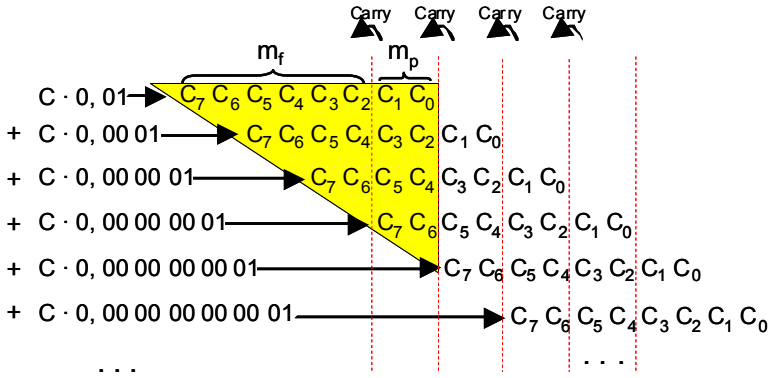


Figura 5-5: Ejemplo de desarrollo del producto periódico mediante una suma múltiple

Hasta ahora todos los cálculos producen resultados enteros sin cifras fraccionarias. La incorporación del orden de magnitud mediante divisiones entre la unidad seguida de ceros se procesa de una forma inmediata desplazando las cifras hacia la derecha de la coma, lo que da lugar a las mantisas fraccionarias del resultado. Esta acción se pospone a la etapa de normalización. De las expresiones [5.30] y [5.32] se deduce que el desplazamiento a realizar corresponde al que muestra la siguiente expresión:

$$\frac{1}{\underbrace{10\cdots0}_{L(m_{fA})}\,\underbrace{0\cdots0}_{L(m_{fB})}\,\underbrace{0\cdots0}_{L(m_{pA})}} \Rightarrow \text{Desplaz.} = L(m_{fA}) + L(m_{fB}) + L(m_{pB}) \qquad [5.35]$$

El cálculo del valor C consta de una multiplicación y una suma entre operandos enteros de longitud variable. Como decisión de diseño se plantea la disyuntiva entre su cálculo serie o paralelo mediante un esquema segmentado. A modo de ejemplo, se presenta en la figura siguiente el modelo segmentado con la partición de los operandos en partes manejables del mismo tamaño. El cálculo se desarrolla de derecha a izquierda según los métodos de suma y producto de números enteros de longitud variable.

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110001010001011100000001101110000011100110100010010100100000010010011100000100010001010011001111110011000111010100

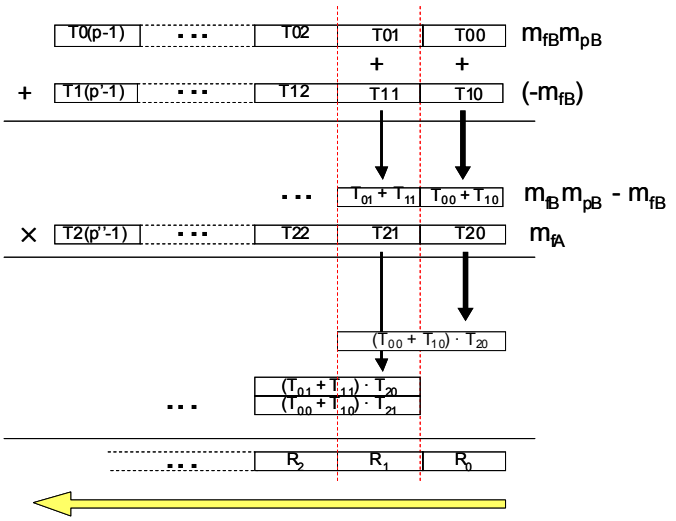


Figura 5-6: Esquema de ejecución de la suma y el producto segmentado

La expresión [5.32] da lugar a un procesamiento más elaborado. Este cálculo contiene una cantidad de sumas de longitud variable proporcional al número de cifras de C entre el número de cifras de la mantisa periódica. A su vez, cada suma se realiza entre valores de un tamaño similar al resultado de la operación producto.

Un posible método de cálculo de esa expresión considera los sumandos desde una perspectiva vertical. De este modo el valor del resultado se obtiene mediante la suma de la sucesión de cantidades de longitud igual al periodo correspondientes a porciones sucesivas del valor C de la misma longitud. La figura 5-7 describe esquemáticamente este proceso, donde las flechas indican el orden de las sumas. Observar que en cada columna sólo hay que hacer una operación de adición, ya que la suma de los demás sumandos es el resultado completo de la columna anterior.

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010100000001001001110000010001 0001010011001111110011000111010 0

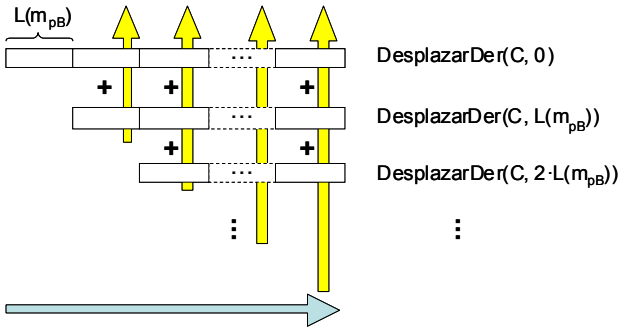


Figura 5-7: Orden de ejecución de las sumas sucesivas por columnas

La siguiente figura muestra este proceso para las cifras del ejemplo anterior. Se observa que todas esas operaciones son de una longitud igual a la cantidad de cifras periódicas $L(m_{pB})$.

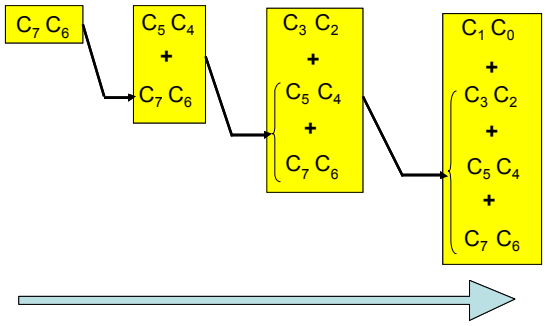


Figura 5-8: Sumas sucesivas por columnas para L(C) = 8; $L(m_{pB})$ = 2

La propagación del acarreo generado se realiza a través de otro conjunto de operaciones de suma similares a las anteriores que comienzan desde las posiciones menos significativas. Se considera acarreo a la cantidad formada por las cifras que excedan de la longitud del periodo, $L(m_{pB})$, en cada una de estas sumas. Las cifras que conforman el periodo del resultado reciben un tratamiento especial, ya que si después de sumar su acarreo se obtuviera uno nuevo, tendría que volver a ser considerado. Por este motivo, como describe la figura siguiente, se obtiene en la misma operación la suma y su sucesor.

11,0010010000111111010101000100001011010001100001000110101000110001001100011001100010100010111000000011101100000111001010001001010010000001001001110000010001000100110011111100110001110100

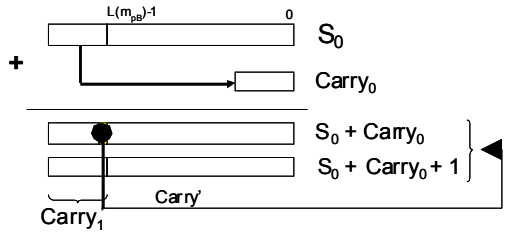


Figura 5-9: Gestión del acarreo en las cifras periódicas del resultado

A partir de ahí y para el resto de partes, el procesamiento consiste en sumar sucesivamente los acarreos producidos al bloque posterior tal y como se observa en la figura siguiente:

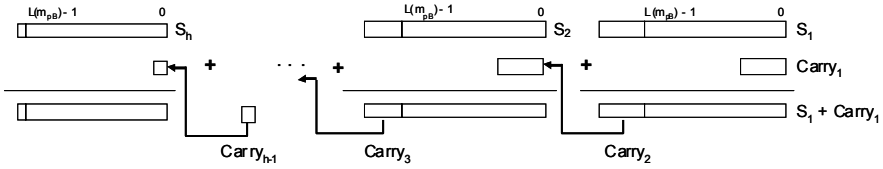


Figura 5-10: Propagación del acarreo

El coste temporal de esta etapa viene dado por el retardo de dos sumas y un producto de longitud variable de una cantidad de cifras similar a la suma de todas las cifras de los operandos. Cada operación de suma tiene un coste lineal con la cantidad de dígitos mientras que el producto tiene una complejidad proporcional al producto de las cifras de los operandos implicados. Aplicando la regla del máximo de la suma de complejidades, el coste temporal asintótico del producto de mantisas corresponde a la expresión siguiente, el cual, sigue siendo cuadrático y proporcional al producto de las cifras que componen los números.

$$T_{Producto\_Mantisas\_b} \in O((L(m_{fA}) \cdot (L(m_{fB})+L(m_{pB})) \quad [5.36]$$

Producto de dos números periódicos

Los factores tienen mantisa periódica significativa: $A, B \in \mathbb{Q}$.

Sea $A = 0,m_{fA}\widehat{m}_{pA} \wedge m_{fA} = \alpha_{L(mfA)-1}\ldots\alpha_0 \wedge m_{pA} = \gamma_{L(mpA)-1}\ldots\gamma_0$ /
/ $L(m_{pA}) > 0 \wedge \exists\, i \in \{0..L(m_{pA})-1\}, \gamma_i \neq 0$

11,0010010000111111011010100010001000010110100011000010001101001100011001100010100010111000000011011100000111001101000100101001000000100100111000000100010010011001111110011000111010 0

$$y\ B = 0,m_{fB}\widehat{m}_{pB} \wedge m_{fB} = \alpha_{L(mfB)-1}\ldots\alpha_0 \wedge m_{pB} = \gamma_{L(mpB)-1}\ldots\gamma_0\ /$$
$$/\ L(m_{pB}) > 0 \wedge \exists\, i \in \{0..L(m_{pB}) - 1\},\ \gamma_i \neq 0 \qquad [5.37]$$

El resultado del producto será un número periódico que, de acuerdo con el tercer teorema del producto de números racionales fraccionarios, tendrá una cantidad de cifras periódicas múltiplo de las cifras periódicas de ambos factores.

La representación simbólica de los factores adopta la expresión general de fracción de los números racionales:

$$A = 0,m_{fA}\widehat{m}_{pA} = \frac{m_{fA}\, m_{pA} - m_{fA}}{\underbrace{(Base-1)\cdots(Base-1)}_{L(m_{pA})}\underbrace{0\ \cdots\ 0}_{L(m_{fA})}}$$

$$B = 0,m_{fB}\widehat{m}_{pB} = \frac{m_{fB}\, m_{pB} - m_{fB}}{\underbrace{(Base-1)\cdots(Base-1)}_{L(m_{pB})}\underbrace{0\ \cdots\ 0}_{L(m_{fB})}}$$

El producto de las fracciones anteriores da lugar a la expresión siguiente:

$$A \cdot B = \frac{(m_{fA}\, m_{pA} - m_{fA}) \cdot (m_{fB}\, m_{pB} - m_{fB})}{\underbrace{(Base-1)\cdots(Base-1)}_{L(m_{pA})}\underbrace{(Base-1)\cdots(Base-1)}_{L(m_{pB})}\underbrace{0\,0\ 0\ \cdots\ 0}_{L(m_{fA})+L(m_{fB})}} \qquad [5.38]$$

11,00100100001111110110101001000100001011010001100001000110101001100010011000110010100010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010100

Reordenando los términos que intervienen en la fracción se obtiene:

$$A \cdot B = \frac{(m_{fA}\, m_{pA} - m_{fA}) \cdot (m_{fB}\, m_{pB} - m_{fB}) \cdot Base^{-(L(m_{fA}) + L(m_{fB}))}}{(Base^{L(m_{pA})} - 1) \cdot (Base^{L(m_{pB})} - 1)} \qquad [5.39]$$

Sean las identidades siguientes:

$$C = (m_{fA}\, m_{pA} - m_{fA}) \cdot (m_{fB}\, m_{pB} - m_{fB}) \qquad [5.40]$$

$$D = \frac{1}{(Base^{L(m_{pA})} - 1) \cdot (Base^{L(m_{pB})} - 1)} \qquad [5.41]$$

Sustituyendo [5.40] y [5.41] en [5.39] la operación adopta la siguiente forma:

$$A \cdot B = C \cdot D \cdot Base^{-(L(m_{fA}) + L(m_{fB}))} \qquad [5.42]$$

De acuerdo con esta expresión, la multiplicación de dos números racionales periódicos se descompone en el producto de tres factores característicos: una cantidad C no periódica que contiene la componente significativa de los números, un valor D que aporta la componente periódica y un desplazamiento que contiene su orden de magnitud. Esta descomposición transforma la multiplicación de dos números racionales periódicos en el producto desplazado de un número no periódico, C, por un número periódico, D. Por consiguiente, conocido el valor de C y D, este caso se puede calcular a partir del método expuesto para el producto de un número no periódico por uno periódico. El desplazamiento se abordará en la etapa de normalización.

El cálculo de C se realiza directamente sobre los operandos mediante operaciones de suma y producto de números enteros de longitud variable. Un posible procedimiento para su cálculo consiste en un esquema segmentado similar al descrito en el caso anterior, en el que se obtiene progresivamente y, de forma alternada, resultados parciales de

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100010100010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

cada operación de suma que serán multiplicados en paralelo con los obtenidos en una iteración anterior.

El procesamiento del factor D requiere de una operatoria más elaborada. La expresión siguiente muestra D en términos de un producto de números periódicos unidad:

$$D = \frac{1}{(Base^{L(m_{pA})} - 1)\cdot(Base^{L(m_{pB})} - 1)} = 0,\overbrace{00\cdots01}^{L(m_{pA})} \cdot 0,\overbrace{00\cdots01}^{L(m_{pB})} \qquad [5.43]$$

De la figura 5-1 se extraen las ecuaciones que determinan el valor de las cifras periódicas de D conforme las expresiones [5.10] y [5.11].

El cuarto teorema del producto de números racionales periódicos formula que los acarreos generados entre los bloques siguen una progresión aritmética de diferencia uno que comienza desde el valor cero para el primero de ellos, es decir: $\forall\, n \in \mathbb{N},\ c_n = n$

A partir de los resultados obtenidos por este teorema, la organización de sumas de la figura 5-1 que produce el desarrollo del producto de los números periódicos se puede expresar mediante una estructura de conjuntos de sumandos T y P tal como se muestra en la siguiente figura:

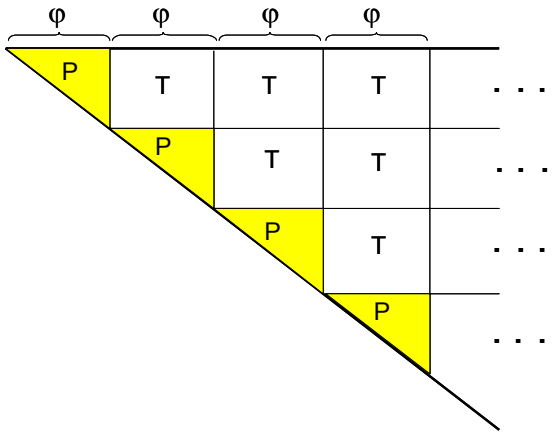


Figura 5-11: Estructura periódica del producto de dos números periódicos

11,00100100001111110110101001000100010110100011000010001101001100011000110011001010001011100000001101110000011100110100010010010000001001001110000010001000101001100111111001100011101000

Como ya se indicó en la expresión [5.22], el valor de cada uno de los bloques T de la figura anterior tiene un valor igual a la base de la representación elevada al número de cifras periódicas menos uno, es decir: $T = Base^{\varphi} - 1$

La figura 5-12 muestra que los sumandos T de una columna, más el acarreo generado por la columna siguiente producen un número potencia de la base más uno. Este valor, más los sumandos residuales P que no constituyen un bloque completo, conforman la cantidad periódica que se repite indefinidamente y da lugar al número racional periódico.

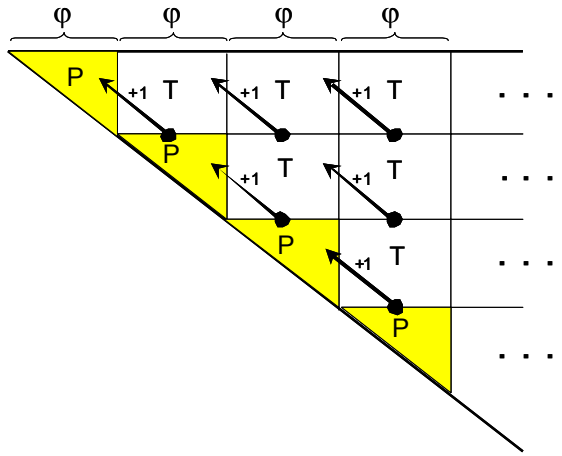


Figura 5-12: Propagación de acarreos producidos

Como consecuencia de esta situación, la cantidad que forma el periodo del número m_{pD} está formada por las sumas de P más uno: $m_{pD} = P+1$.

La longitud del periodo, φ, será aquella para la que la suma de las sucesivas rotaciones de la mantisa del primer número periódico puro, una cantidad de veces múltiplo de la longitud de la mantisa del segundo número, sea igual a la base elevada al número de cifras menos dos. La siguiente expresión describe esta relación:

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001 0001010011001111110011000111010 0

$$\varphi \in \mathbb{N} \;/\; \left[\sum_{i=0}^{\varphi - L(m_{pB})} \text{RotarIzq}\left([p_A]_{L(mpB)+}\,,\, i \cdot L(m_{pB})\right) \right] = \text{Base}^{\varphi} - 2 \qquad [5.44]$$

donde,

p_A: mantisa periódica de $[\text{Base}^{L(mpA)} - 1]^{-1}$.

$[p_A]_{L(mpB)+}$: selección de las $L(m_{pB})$ cifras más significativas de p_A.

$\text{RotarIzq}(y, x)$: rotación del valor y, x posiciones a la izquierda.

Se observa claramente que la cantidad de sumas de T y P coincide con la cantidad de sumas desplazadas del periodo, como muestra la figura siguiente:

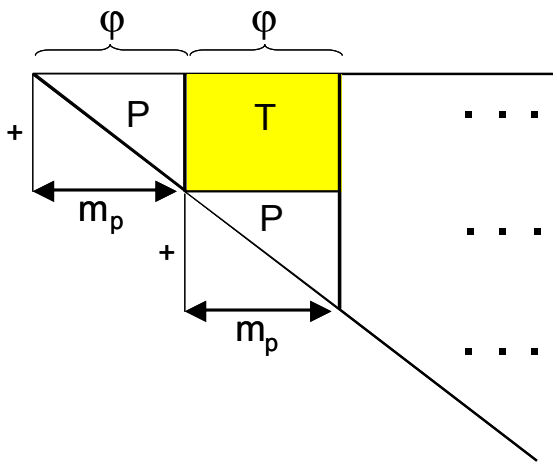


Figura 5-13: Relación entre m_p y T

Teniendo en cuenta que el número φ es múltiplo de la cantidad de cifras periódicas de ambos operandos, un procedimiento para calcular el periodo del resultado consiste en realizar sumas desplazadas de uno de los periodos unidad de los factores respecto al otro hasta alcanzar la cota $\text{Base}^{\varphi} - 1$. El conjunto de sumandos que intervienen constituirá el bloque T. De este modo, el cálculo de las cifras periódicas sigue el esquema de sumas sucesivas que describe la expresión siguiente:

11,0010010000111111010101000100010000101101000110000100011010011000100111000110011000101000101110000000011011100000111001010010010000001001001110000010001000101001100111111001100011101000

$$D = \left(\sum_{i=0}^{h} DesplazarDer\left(p'_A , i \cdot L(m_{pB}) \right) \right) + 1 \qquad [5.45]$$

donde,

p'_A :	$\varphi - L(m_{pB})$ cifras más significativas de la parte fraccionaria de $[Base^{L(mpA)} - 1]^{-1}$.	
$DesplazarDer(y, x)$:	desplazamiento del valor y, x posiciones a la derecha.	
h:	$\lceil \varphi / L(m_{pB}) \rceil - 1$.	[5.46]

El método propuesto para el cálculo del factor D calcula las cifras del periodo directamente sin tener que calcular previamente su cantidad de cifras. Las continuas rotaciones de la mantisa unidad del primer número periódico corresponden con secuencias de cifras correlativas de longitud $L(m_{pB})$ pertenecientes a la expansión periódica de dicho operando. La siguiente figura muestra esta observación gráficamente, donde p_A es el desarrollo del número periódico unidad del primer factor.

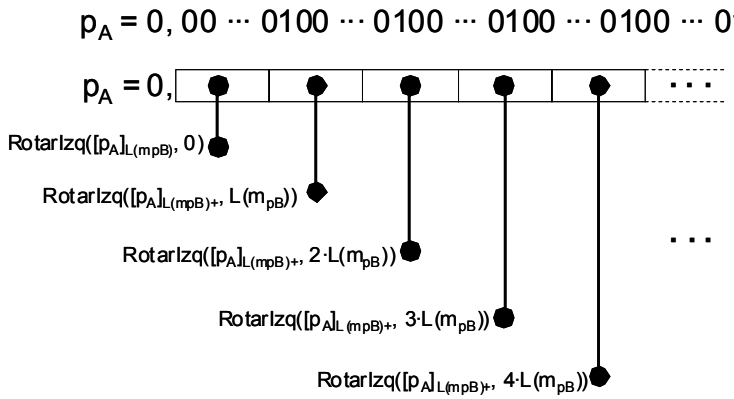


Figura 5-14: Rotaciones sucesivas de la mantisa periódica

Las sumas sucesivas de tales rotaciones corresponden verticalmente con las $L(m_{pB})$ cifras de menor peso de las sumas desplazadas en el calculo del periodo. Esta observación permite calcular directamente y de forma gradual la mantisa del número periódico D siguiendo el siguiente

11,0010010000111111011010100010001000010110100011000010001101001100011001100101000101110000000110111000001110011010001001010010000001001001110000010001 0001010011001111110011000111010 0

procedimiento iterativo: tomando como primer término una secuencia de $L(m_{pB})$ ceros, se concatenan los resultados de las sumas entre el valor anterior y la sucesiva rotación hacia la izquierda de un número periódico unidad de grado igual a $L(m_{pA})$. El proceso termina cuando tras una suma se alcanza el valor $Base^{L(mpB)} - 1$. Esa cantidad no se concatena al resultado, en su lugar se incrementa en una unidad el anterior valor obtenido.

La siguiente figura muestra que los resultados de esas sumas forman parte del periodo. En cada suma se obtiene una nueva porción de $L(m_{pB})$ cifras de D de forma incremental. Debido a la naturaleza del valor límite no hay posibilidad de que se produzca acarreo durante el procedimiento y solo en la última iteración se suma uno al resultado.

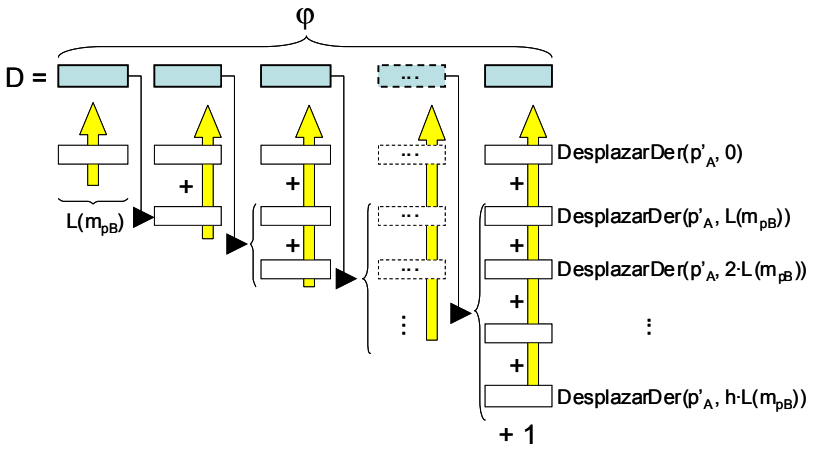


Figura 5-15: Sumas sucesivas por columnas

Una vez obtenido el valor de C y D se multiplican entre sí según el procedimiento de multiplicación de un número periódico por otro no periódico. Con respecto al orden de magnitud del resultado, será en la etapa de normalización donde se ajuste según la magnitud y el sentido del desplazamiento.

Las sumas y productos que intervienen en el cálculo de los factores C y D son operaciones de longitud variable y se realizan según los algoritmos que se han descrito para ellas. La complejidad temporal de

11,0010010000111111011010100010000100010110100011000010001101001100010011000110011000010100010111000000011011100000111001010001001010010000001001001110000010001
0001010011001111110011000111010 0

este apartado está en función del tiempo empleado en el cálculo de C y D y de su multiplicación.

- *Cálculo de C*: el coste resulta proporcional a la cantidad de cifras de los operandos:

$$T_{Producto\_Mantisas\_c1} \in O(L(m_A) \cdot L(m_B)) \quad [5.47]$$

- *Cálculo de D:* la sucesión de rotaciones y sumas necesaria para su cálculo es proporcional a la cantidad de cifras del periodo, el cual a su vez es múltiplo de las longitudes de los periodos de ambos números.

$$T_{Producto\_Mantisas\_c2} \in O(L(m_{pA}) \cdot L(m_{pB})) \quad [5.48]$$

- *Producto C · D*: esta multiplicación tiene un coste asintótico proporcional al producto de la cantidad de cifras de C y D, es decir, lineal con la cantidad total de cifras fijas de las mantisas y cuadrático con las periódicas.

$$T_{Producto\_Mantisas\_c3} \in O((L(m_A) + L(m_B)) \cdot L(m_{pA}) \cdot L(m_{pB})) \quad [5.49]$$

Las expresiones anteriores se combinan mediante la regla del máximo de complejidades, y queda una complejidad asintótica temporal según se muestra en la siguiente expresión:

$$T_{Producto\_Mantisas\_c} \in O((L(m_A) + L(m_B)) \cdot L(m_{pA}) \cdot L(m_{pB})) \quad [5.50]$$

Determinación del signo

El signo del resultado dependerá únicamente de los signos de los operandos y se obtiene mediante el método tradicional para el producto. Su cálculo se puede realizar en cualquier momento durante toda la operación y en paralelo con otra etapa de la misma.

Como se observa en las expresiones de la complejidad, el coste total de la etapa de producto de mantisas depende de la naturaleza de los operandos iniciales. En ello, existen tantas variables independientes como campos significativos tienen los números, lo que impide una

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001 0001010011001111110011000111010 0

expresión sencilla de la complejidad de la operación completa. De las expresiones [5.25], [5.36] y [5.50] se deduce que el retardo de esta etapa corresponde a la siguiente expresión:

$$T_{Producto\_Mantisas} \in O((L(m_A)\cdot L(m_B)) , (L(m_A)+L(m_B))\cdot L(m_{pA}) \cdot L(m_{pB}))) \quad [5.51]$$

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

Normalización

En esta etapa se procesa la mantisa y el exponente generados en las etapas anteriores para adecuarlos al formato propuesto en este trabajo y evitar ambigüedades de la representación. Se realizan las mismas tareas que en la operación de suma, sin embargo, al no existir compensación entre los dos operandos en esta operación, la búsqueda del primer uno significativo necesitará de un menor tiempo de procesamiento. Por otra parte, la actualización del exponente debe tener en cuenta no sólo los desplazamientos originados por la detección del primer uno sino también los movimientos pendientes tras el producto de mantisas.

La complejidad temporal de estas operaciones para el producto tendrá una expresión similar a la de la suma.

$$T_{\text{Normalización}} \in O(L(e_A), L(e_B), (L(m_A) + L(m_B)) + L(m_{pA}) \cdot L(m_{pB})) \qquad [5.52]$$

Coste total de la operación de multiplicación

Las etapas de cálculo que forman la operación de multiplicación pueden ordenarse según distintos criterios de diseño. Desde un punto de vista de las dependencias entre los datos su ejecución se puede realizar en serie o bien en paralelo, en concreto el cálculo de los exponentes junto con el producto de mantisas. Sin embargo, el coste asintótico de la operación completa no se ve afectado por estas decisiones y tomará el máximo coste de las etapas que intervienen.

La independencia en la longitud de los campos que constituyen el número provoca una expresión en términos condicionales. La complejidad temporal del producto de números racionales de longitud variable corresponde a la que se muestra en la expresión siguiente:

$$T_{\text{multiplicación-}\mathbb{Q}} \in O(T_1, T_2, T_3) \qquad [5.53]$$

donde,

$$T_1 \in O(L(e_A), L(e_B))$$

$T_2 \in O((L(m_A) \cdot L(m_B)) , (L(m_A) + L(m_B)) \cdot L(m_{pA}) \cdot L(m_{pB})))$

$T_3 \in O(L(e_A), L(e_B), (L(m_A) + L(m_B)) + L(m_{pA}) \cdot L(m_{pB}))$

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

Ejemplo del producto de dos números periódicos

Con el objetivo de clarificar el procedimiento de multiplicación de dos números racionales periódicos se desarrollan dos ejemplos para números en decimal y en binario.

Ejemplo 1: Base 10

Sea:

$$A = 0,9\widehat{020} \quad \rightarrow m_{fA} = 90;\ m_{pA} = 20;\ L(m_{fA}) = 2;\ L(m_{pA}) = 2$$

$$B = 0,\widehat{67} \quad \rightarrow m_{fB} = 6;\ m_{pB} = 7;\ L(m_{fB}) = 1;\ L(m_{pB}) = 1$$

Cálculo de C:

$$C = (9020 - 90) \cdot (67 - 6) = 544730$$

Cálculo de D:

$$D = \frac{1}{99 \cdot 9} = 0,\widehat{01} \cdot 0,\widehat{1}$$

De acuerdo con el tercer teorema del producto de números periódicos, la cantidad de cifras que componen el periodo será múltiplo de dos. A continuación se presenta el proceso de cálculo del factor D, según el procedimiento expuesto.

El método consiste en realizar sumas sucesivas sobre las rotaciones del periodo de A y comprobar, tras cada operación, su igualdad con el patrón $Base^{L(mpB)} - 1$. Cada suma parcial obtenida, excepto la última, pasará a formar parte del valor D hasta que se cumpla la condición de finalización y se sume uno al resultado. El número así constituido por la concatenación de tales sumas corresponde con la mantisa periódica de D.

La figura 5-16 muestra todo el proceso de composición de D a partir de las sumas y las rotaciones realizadas. El valor límite de las sucesivas sumas es: $10^1 - 1 = 9$.

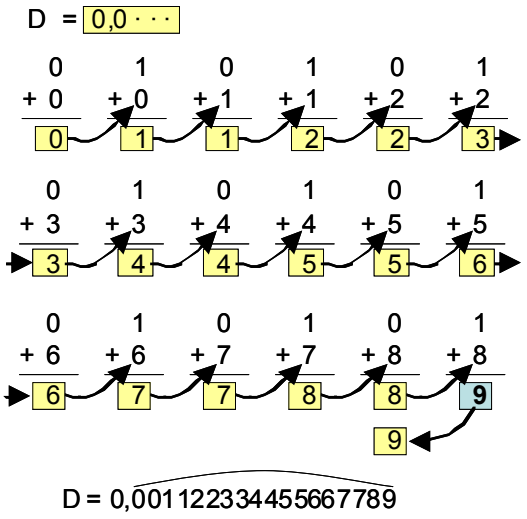
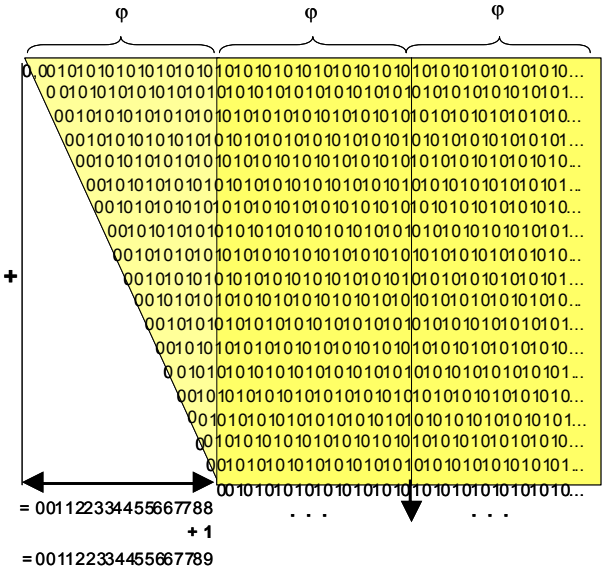


Figura 5-16: Composición de las cifras del valor D en el ejemplo decimal

La figura 5-17 muestra la estructura de sumas generada:



11,00100100001111110110101001000100001011010001100001000110101001100010011000110010100010111000000011011100000111001101000100101001000000100100111000001000100010100110011111001100011101100100

Figura 5-17: Desarrollo de sumas para la obtención de D en el ejemplo decimal

En la figura anterior se observa cómo se forma el valor del periodo a partir de los sumandos señalados por el triángulo de la izquierda de la figura. Se marcan los conjunto de sumas T, cuyo valor es de $10^{\varphi} - 1$.

La operación se transforma en un producto entre un número no periódico y un número periódico:

$$A \cdot B = 544730 \cdot 0,\overbrace{0011223344556677 89} \cdot 10^{-3}$$

Esta operación se realiza según el procedimiento descrito para ella que ilustra la siguiente figura:

$$544730 \cdot 1122334455667789 = 611369248035914701970$$

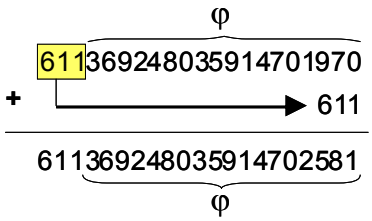


Figura 5-18: Cálculo del producto de un número periódico por otro no periódico en el ejemplo decimal

Finalmente, se normaliza el resultado según el desplazamiento requerido y la posición de la primera cifra mayor que cero.

$$R = A \cdot B = 0,\overbrace{611369248035914702581}$$

donde,

$m_{fR} = 611$; $m_{pR} = 369248035914702581$; $L(m_{fR}) = 3$; $L(m_{pR}) = 18$

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010010000001001001110000010001 0001010011001111110011000111010 0

Ejemplo 2: Base 2:

Sea:

$A = 0,10\overset{\frown}{1001} \rightarrow \quad m_{fB} = 10;\ m_{pB} = 1001;\ L(m_{fB}) = 2;\ L(m_{pB}) = 4$

$B = 0,110\overset{\frown}{0011} \rightarrow \quad m_{fA} = 1100;\ m_{pA} = 011;\ L(m_{fA}) = 4;\ L(m_{pA}) = 3$

Cálculo de C:

$$C = (101001 - 10) \cdot (1100011 - 1100) = 110101000001$$

Cálculo de D:

$$D = \frac{1}{1111 \cdot 111} = 0,\overset{\frown}{0001} \cdot 0,\overset{\frown}{001}$$

La cantidad de cifras que componen el periodo será múltiplo de tres y de cuatro, es decir, múltiplo de doce.

Al igual que en el ejemplo, anterior se obtiene el valor de D según el método propuesto. En este caso el valor límite es de:

$$Base^{L(mpB)} - 1 = 2^3 - 1 = 111_2$$

En la figura siguiente se muestra todo el proceso de formación de D:

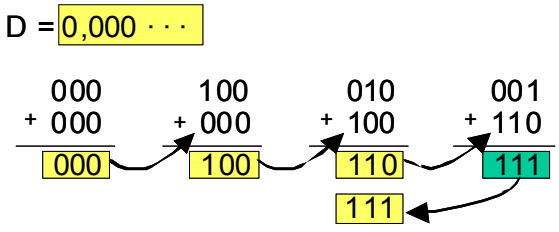


Figura 5-19: Composición de las cifras del valor D en el ejemplo binario

De esta forma:

$$D = 0,\overset{\frown}{000000100111}$$

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110001010001011100000001101110000011100110100010010100100000010010011100000100010001010011001111110011000111010100

El producto queda:

$$A \cdot B = 110101000001 \cdot 0,\overline{000000100111} \cdot 2^{-6}$$

La siguiente figura muestra la estructura de sumas generada para la formación de D. Se observan los conjuntos de sumas T y los sumandos que constituyen las cifras periódicas.

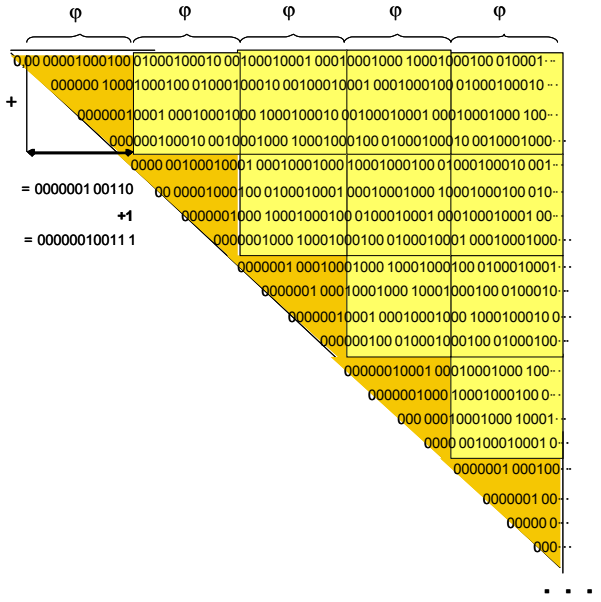


Figura 5-20: Desarrollo de sumas para la obtención de D en el ejemplo binario

Realizando el producto del número periódico por el no periódico:

$$110101000001 \cdot 100111 = 100000010011100111$$

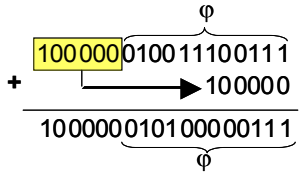


Figura 5-21: Cálculo del producto de un número periódico por otro no periódico en el ejemplo binario

Y finalmente, tras la normalización se obtiene el resultado:

V - 2

Instrumentación de la función multiplicación

Conocido el procedimiento para multiplicar dos números racionales expresados en el formato propuesto, a continuación se describe la instrumentación de la función multiplicación en precisión variable que tiene en cuenta restricciones acerca de la cantidad de cifras significativas del resultado. Posteriormente se analizan experimentalmente algunos aspectos del método de multiplicación y de su complejidad y se comprueba la calidad de los resultados de otros métodos de multiplicar convencionales mediante su comparación con los resultados exactos que produce el método propuesto.

La función en precisión variable admite control sobre la cantidad de cifras mediante un parámetro adicional a los factores, lo que permite distintos grados de aproximación al valor exacto según describe la siguiente expresión:

$$|\Gamma^{VP}_{producto}(\overline{x}, \overline{d}) - producto(\overline{x})| \leq \varepsilon \qquad [5.54]$$

11,00100100001111110110101001000100001011010001100001000110101100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001000100110011111100110001110100

El perfil de la instrumentación $\Gamma^{VP}_{producto}(\overline{x}, \overline{d})$ tiene la siguiente formulación:

$$\Gamma^{VP}_{producto}(\overline{x}, \overline{d}) : \mathbb{Q} \times \mathbb{Q} \times \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{Q} \qquad [5.55]$$

donde los parámetros mantienen el mismo significado que en la función anterior:

$$\overline{x} \equiv (x, y) \in \mathbb{Q} \times \mathbb{Q}$$
$$\overline{d} \equiv \langle d_{L(e)}, d_{L(m)} \rangle, \text{ con } d_{L(e)} \in \mathbb{N} \wedge d_{L(m)} \in \mathbb{N}$$

Como se ha comprobado en la primera parte de este capítulo, la función obtiene el resultado exacto del producto de factores racionales expresados en el formato propuesto con un tamaño de registro finito, tal como se formula en la siguiente expresión:

$$\forall x, y \in \mathbb{Q}, \exists \overline{d} \in \mathbb{N}x\mathbb{N} / \Gamma^{VP}_{producto}(x, y, \overline{d}) = producto(x, y) \qquad [5.56]$$

En los siguientes apartados se presenta la instrumentación de la función producto, $\Gamma^{VP}_{producto}$, donde se describe la propuesta de diseño de cada una de las etapas del algoritmo con el mismo criterio que el empleado para la suma. Se vigila el crecimiento de los números para proveer un valor aproximado en caso de que no sea posible alcanzar el resultado sin error. Los operadores elementales de suma y multiplicación que intervienen se construyen mediante lógica almacenada por coherencia con el resto del modelo. Los factores mantienen la estructura que muestra la figura 4-14 para la suma.

Suma de exponentes

La operación se realiza mediante el método propuesto en este trabajo y calcula la suma y su sucesor. La resta del sesgo del resultado se implementa complementando el bit más significativo del sucesor de la suma de exponentes, tal como se muestra en la siguiente figura:

11,0010010000111111010101000100010000101101000110000100011010011000100110001100110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111100110001111010

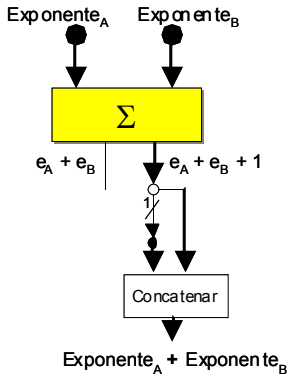
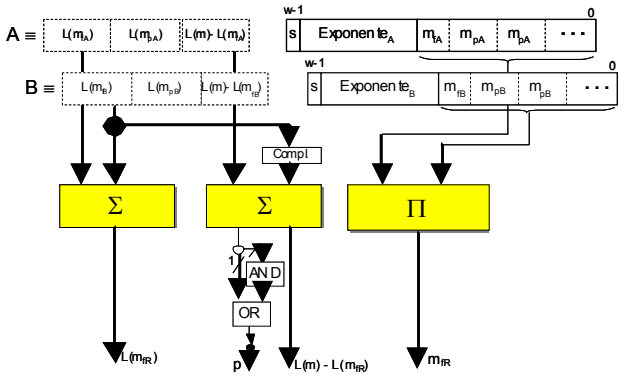


Figura 5-22: Esquema de la etapa del cálculo de la suma de exponentes

Producto de mantisas

En el producto de dos números no periódicos, la multiplicación de las mantisas de ambos factores conforma directamente la mantisa del resultado. La instrumentación basada en el formato de representación que se propone debe calcular también su longitud así como la separación entre la parte fija y periódica, para lo cual son necesarias sendas operaciones de suma sobre los valores correspondientes de los operandos. En el caso de que el resultado sobrepase la zona dedicada a contenerlo, el indicador p de exactitud marcará la presencia de un resultado aproximado. En la figura siguiente se aprecia el diagrama funcional de esta etapa.



11,0010010000111111011010100010001000101101000110000100011010011000100111000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

Figura 5-23: Esquema de la etapa del producto de mantisas en la multiplicación de dos números no periódicos

La instrumentación del producto de un número no periódico por otro periódico considera el carácter cíclico de las cifras de uno de los operandos. En primer lugar se separan las mantisas fijas de las periódicas y se colocan en la zona menos significativa de los registros mediante desplazamientos hacia la derecha. La multiplicación de su producto por la componente periódica del número se implementa a través de una estructura iterativa que va calculando progresivamente el valor de la expresión [5.33] y del acarreo producido a partir de las cifras periódicas. Éste cálculo se realiza en otra estructura de suma paralela dentro de la iteración anterior.

Como se deduce de figura 5-24, la instrumentación que se propone difiere ligeramente del método de cálculo que se ha presentado para operar con números de longitud variable, algo esperado al manipular registros de longitud fija. En relación con el mantenimiento de las longitudes de los registros, tan sólo se precisa la longitud del periodo, ya que los otros datos serán calculados en la etapa de normalización.

11,0010010000111111011010100010001000010110100011000010001101001100011001100011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

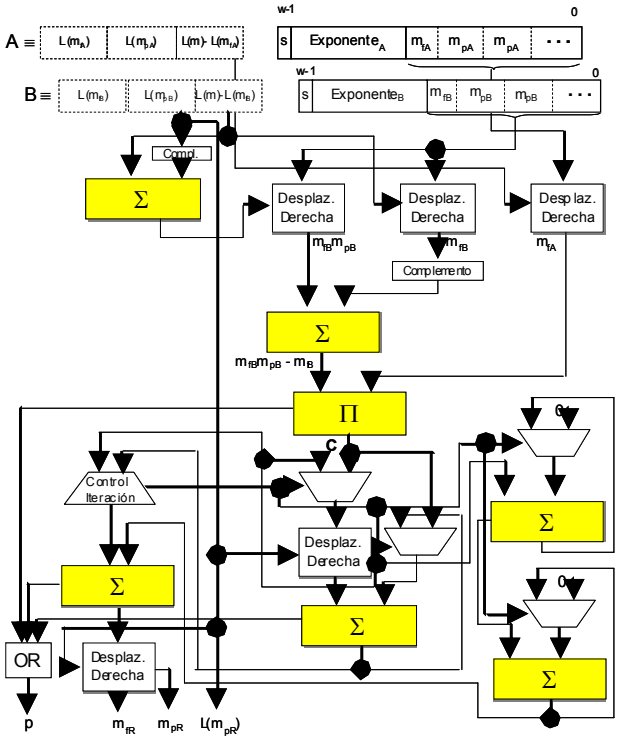


Figura 5-24: Esquema de la etapa de producto de mantisas en la multiplicación de un número no periódico y otro periódico

En la figura anterior se observa que la gestión de la precisión está presente en los módulos que implican crecimiento del contenido de los registros. El operador de multiplicación incorpora un indicador que avisa de la existencia de desbordamiento, aunque su activación no impide continuar con el resto del proceso hasta completar el registro con el propósito de obtener el mejor resultado. En las operaciones de suma se considera el acarreo que se produzca en la misma medida.

Finalmente, el producto de dos números periódicos debe resolver la instrumentación de la ecuación [5.42] para convertir la operación en la multiplicación de un número no periódico por otro periódico. El cálculo del factor no periódico emplea los operadores de suma y producto entre datos alojados en registros de longitud fija. Para la obtención del factor periódico se emplea una estructura que sigue fielmente el

11,00100100001111111011010100010001000101101000110000100011010011000100111000101000101110000000111011010001001010010000001001001110000010001000101001100111111001100011101000

procedimiento que ilustra la figura 5-15. En su construcción se utiliza un módulo de generación de periodo unidad, un desplazador, un sumador, un concatenador de cifras en un registro y un módulo de control para terminar el proceso y vigilar el desbordamiento. La siguiente figura ilustra la organización de todos los componentes necesarios que obtienen el valor del los factores mencionados.

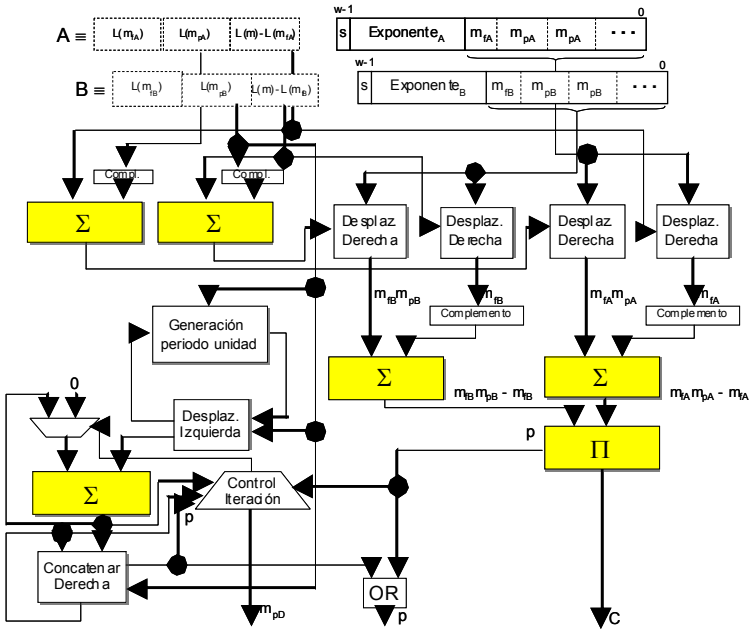


Figura 5-25: Esquema de la etapa de producto de mantisas en la multiplicación de dos números periódicos

La gestión de la precisión sigue un procedimiento similar al del caso anterior. Su control es necesario en las operaciones de multiplicación y en la generación de la mantisa periódica, ya que las sumas que intervienen no producen acarreo ni desbordamiento. Cuando no sea posible mantener el resultado en los registros disponibles se activa el indicador de exactitud y se detiene el proceso de cálculo de la parte periódica del resultado.

Normalización

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

Las acciones que se realizan en esta etapa son iguales a las que se ejecutan en la operación de suma por lo que se mantiene la instrumentación que muestra la figura 4-19.

Evaluación empírica

Los experimentos sobre la multiplicación se realizan en las mismas condiciones de entorno que en las pruebas anteriores de las funciones identidad y suma. Se sigue el mismo esquema que en esta última operación al clasificar las pruebas en tres conjuntos en función de su ámbito de estudio.

Experimentos I

Este primer conjunto de experimentos se centra en estudiar la frecuencia de simplicación de las mantisas periódicas y de eliminación de submantisas periódicas en la mantisa fija del resultado. Se consideran distintos intervalos de generación de valores racionales.

El perfil de este conjunto de pruebas es el siguiente:

- Producto de números racionales periódicos representados en el formato propuesto.
- Los valores pertenecen al intervalo $(0, 256]$, donde cada número se construye mediante la fracción $\frac{a}{b}$, siendo a, b números enteros aleatorios generados en intervalos crecientes:

  $a, b \in [1..i]$, para $i = \{16, 32, 64, 128, 256\}$
- Realización de 10^6 productos independientes en cada intervalo.

11,0010010000111111011010100010000100001011010001100001000110100110001001100011001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

Los resultados obtenidos se muestran en la siguiente tabla:

Prueba a/b	16	32	64	128	256
Simplificación de la mantisa periódica	34,85%	44,87%	49,55%	53,25%	71,95%
Existencia de submantisas periódicas en la mantisa fija	23,14%	17,69%	10,12%	7,98%	0,05%

Tabla 5-1: Frecuencia de normalización de las mantisas fijas y periódicas

De los datos que muestra la tabla 5-1 se deduce que la tendencia de la frecuencia para la normalización de la mantisa periódica es creciente conforme aumenta el rango de generación de los factores. Esto es debido a la entrada en operación de un conjunto cada vez mayor de divisores comunes de los denominadores. Sin embargo, la tendencia de frecuencia de la normalización de la mantisa fija es decreciente y próxima a cero para intervalos de generación elevados. En consecuencia se considera oportuno realizar la normalización de la mantisa para ajustar el resultado a su configuración correcta, aunque en aquellas situaciones en las que el coste temporal es un aspecto a considerar, se puede eximir la normalización de la mantisa fija debido al pequeño porcentaje de veces en las que actúa, o bien, realizarla sólo tras un conjunto de operaciones.

Experimentos II

El objetivo de estos experimentos es el de establecer una relación entre la naturaleza de los operandos y la complejidad espacial en operaciones sucesivas encadenadas mediante el estudio del crecimiento en el número de cifras de la mantisa fija y periódica de los resultados. Para ilustrar la relación con el rango de generación de los números se consideran intervalos crecientes según una progresión aritmética de diferencia diez. El perfil de las pruebas que se realizan es el siguiente:

- Producto de números racionales periódicos representados en el formato propuesto.

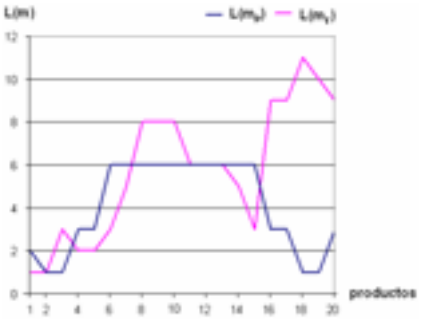
11,00100100001111110110101010001000100001011010001100001000110100110001001100011001100101000001100111010001001010010000010010011100000100010001010011001111110011000111010 0

- Los valores pertenecen al intervalo (0, 80], donde cada número se construye mediante la fracción $\frac{a}{b}$, siendo a, b números enteros aleatorios generados en intervalos crecientes:

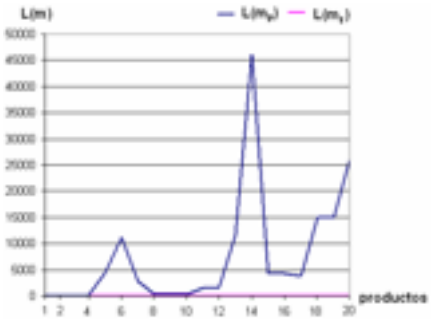
  a, b ∈ [1..i], para i = {10, 20, …, 80}

- Realización de 20 multiplicaciones sucesivas en cada intervalo.

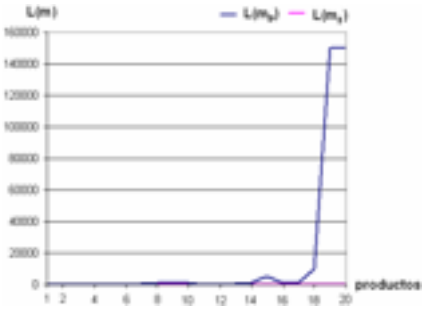
Los resultados obtenidos se muestran en las gráficas de la figura 5-26. En cada gráfica se muestra conjuntamente el crecimiento de la mantisa fija y periódica para el intervalo de generación indicado.



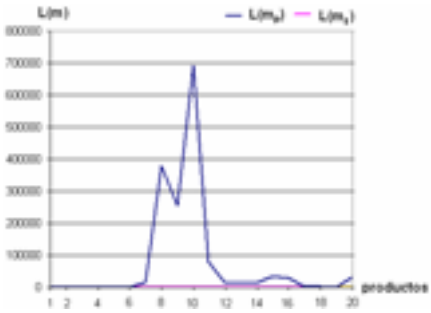
a, b ∈ [1..10]



a, b ∈ [1..20]



a, b ∈ [1..30]



a, b ∈ [1..40]

11,0010010000111111011010100010001000010110100011000010001101001100011001100110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101100

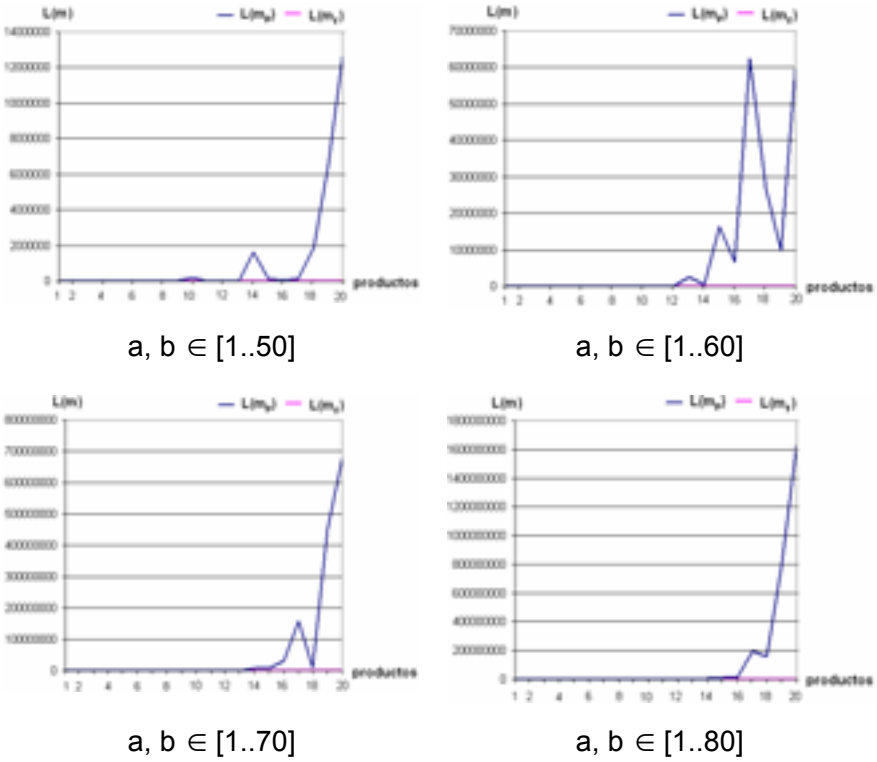


Figura 5-26: Crecimiento de la longitud de las mantisas del resultado de operaciones encadenadas

Se observa que el crecimiento de las mantisas fijas es significativamente menor que el de las mantisas periódicas. Éstas presentan un crecimiento que está en relación directa con el tamaño del intervalo de generación de los números y de cantidad de multiplicaciones que se realicen. Se producen notables disminuciones y aumentos bruscos de tamaño que se deben a las frecuentes simplificaciones en la etapa de normalización que reducen la longitud de la mantisa periódica.

En todos los casos, para un pequeño número de multiplicaciones sucesivas, el tamaño de la mantisa periódica se mantiene en unos límites reducidos independientemente del rango de generación de los factores.

11,00100100001111110110101000100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

Experimentos III

El objetivo de estas pruebas es el de comprobar la calidad de los resultados que proporciona el método de multiplicar convencional sobre operandos representados en el formato estándar IEEE-754 para secuencias de multiplicaciones encadenadas. Las pruebas comparan la desviación que produce el método convencional con el resultado exacto. Para contemplar el mayor número de casos se consideran múltiples escenarios de generación de los factores que combinen los aspectos de redondeo, precisión y cantidad de operaciones sucesivas.

El perfil de los experimentos es el siguiente:

- Realización de multiplicaciones sucesivas mediante el método de multiplicar propuesto para operandos racionales y mediante la multiplicación de acuerdo con la representación IEEE-754 en simple y doble precisión. La relación entre la cantidad de multiplicaciones de cada secuencia corresponde con una progresión geométrica de razón 10 según la siguiente expresión:

  $$R_{\mathbb{Q}} = \prod_{i=1}^{10^t} q_i \text{ , donde } t \in \{0..7\} \qquad [5.57]$$

- Los números pertenecen al intervalo $(0, 50]$ y se construyen mediante una fracción $\frac{a}{b}$, donde a, b son valores enteros en el rango $[1..50]$.

- Realización de 10^3 pruebas de cada caso.

El crecimiento de la complejidad espacial de los sucesivos resultados parciales de las multiplicaciones sugiere orientar la generación aleatoria de los factores para mantener el tamaño de los números dentro de unos límites manejables. La sucesión de los factores de cada secuencia de operaciones se genera con el siguiente criterio:

$q_1 = \frac{a_1}{b_1}$, donde a_1 y b_1 son aleatorios en el rango $[1..50]$.

$q_2 = \frac{a_2}{b_2}$, donde $a_2 = b_1$ y b_2 es aleatorio en el rango $[1..50]$.

…

11,00100100001111110110101000100010000101101000110000100011010011000100111000101000101110000000011011100000111001101000100101001000000100100111000001000100101001111100110001110100

$q_i = \frac{a_i}{b_i}$, donde $a_i = b_{i-1}$ y b_i es aleatorio en el rango [1..50].

…

$q_t = \frac{a_t}{b_t}$, donde $a_t = b_{t-1}$ y $b_t = a_1$.

Con este método de generación de los operandos, el resultado de la expresión [5.58], independientemente del número de factores que contenga, es siempre igual a 1. Se ha comprobado experimentalmente que en todos los casos el método propuesto alcanza este número exactamente. Para medir la calidad de los métodos convencionales se comparan sus resultados con el valor esperado en cada caso. La siguiente tabla muestra el promedio de la posición de la primera cifra incorrecta en los resultados.

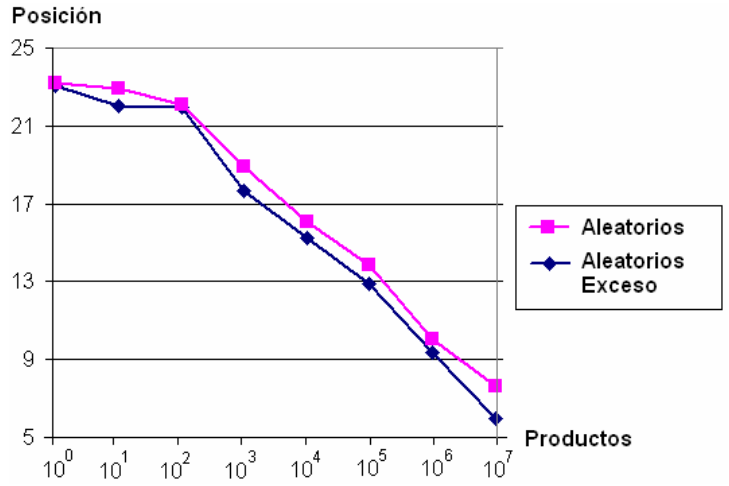
11,00100100001111110110101000100010000101101000110000100011010011000100110001100110001010001011100000001101110000011100110100010010010000001001001110000010001
000101001100111111001100011110100

Simple precisión:

IEEE-754 simple precisión				
Número de operaciones	Números aleatorios		Números aleatorios con redondeo por exceso	
	Posición	σ	Posición	σ
10^0	23,23	10,43	23,09	10,35
10^1	22,91	8,8	22,01	8,74
10^2	22,11	8,83	21,98	8,56
10^3	18,89	7,95	17,63	7,83
10^4	16,09	7,04	15,21	8,046
10^5	13,87	7,38	12,92	6,65
10^6	10,07	6,48	9,34	6,00
10^7	7,62	5,23	5,99	5,78

Tabla 5-2: Primera posición incorrecta en operaciones sucesivas con el formato IEEE-754 en simple precisión

El la figura siguiente se muestran gráficamente los resultados de la tabla anterior.



11,00100100001111110110101001000100001011010001100001000110101100010011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011110100

Figura 5-27: Primera posición incorrecta en operaciones sucesivas con el formato IEEE-754 en simple precisión

Doble precisión:

IEEE-754 doble precisión				
Número de operaciones	Números aleatorios		Números aleatorios con redondeo por exceso	
	Posición	σ	Posición	σ
10^0	51,87	16,32	51,74	16,10
10^1	51,23	15,49	51,05	16,45
10^2	50,16	15,36	49,78	15,37
10^3	48,34	14,93	47,34	15,28
10^4	45,93	14,13	45,11	14,35
10^5	43,13	14,05	42,34	13,48
10^6	39,81	13,53	38,67	13,25
10^7	36,21	13,07	34,85	12,58

Tabla 5-3: Primera posición incorrecta en operaciones sucesivas con el formato IEEE-754 en doble precisión

El la figura siguiente se muestran gráficamente los resultados de la tabla anterior.

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001100101000001110011010001001010010000001001001110000010001 0001010011001111110011000111010 0

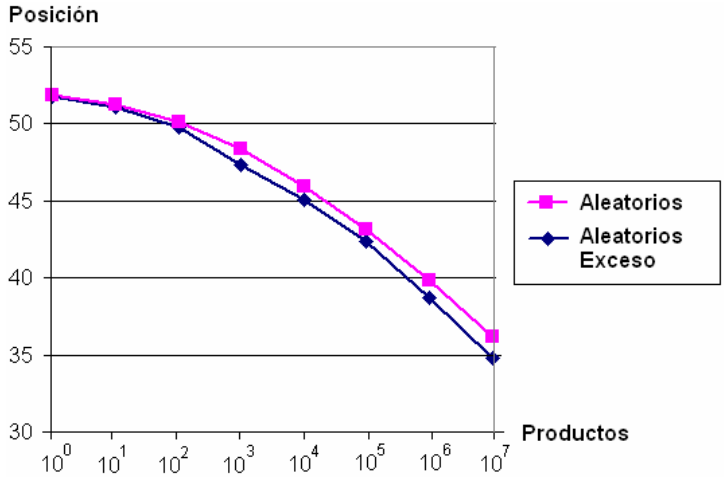
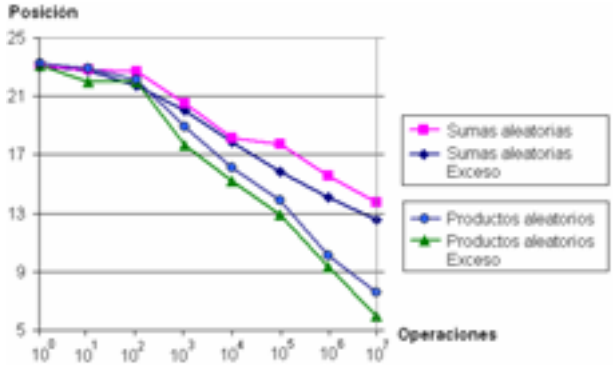


Figura 5-28: Primera posición incorrecta en operaciones sucesivas con el formato IEEE-754 en doble precisión

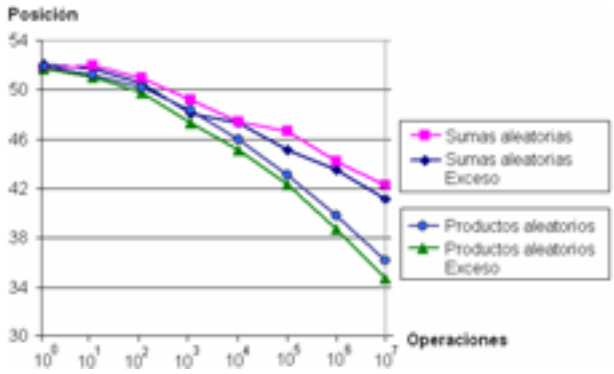
Los resultados que muestran las tablas 5-2 y 5-3 y que ilustran las figuras 5-27 y 5-28 verifican que la posición de la primera cifra diferente se aleja de la precisión establecida por el formato conforme aumenta la cantidad de operaciones y cada vez adquiere mayor pendiente. En todos los casos, las multiplicaciones de números representados con un redondeo en exceso producen un resultado de peor calidad que el producto de valores con un redondeo arbitrario.

En la comparación de los datos de la multiplicación con los de la operación de suma se observa que con la misma cantidad de operaciones la desviación que se produce en una secuencia de multiplicaciones es mucho mayor que en una secuencia de sumas. En la siguiente gráfica se muestra la tendencia comparada para números en simple y doble precisión.

11,00100100001111110110101000100010000101101000110000100011010011000100011000110011000101000101110000000110111000001110011010001001010010000001001001110000010001 0001010011001111110011000111010 0



(a)



(b)

Figura 5-29: Comparación de la primera posición incorrecta en operaciones sucesivas con el formato IEEE-754.(a) simple precisión; (b) doble precisión

Las siguientes tablas y figuras muestran la dimensión del error cometido en relación con el valor exacto del resultado.

Para datos en *simple precisión*:

IEEE-754 simple precisión				
Número de operaciones	Números aleatorios		Números aleatorios con redondeo por exceso	
	Posición	σ	Posición	σ
10^0	$2{,}991 \cdot 10^{-8}$	$1{,}429 \cdot 10^{-13}$	$3{,}001 \cdot 10^{-8}$	$8{,}284 \cdot 10^{-12}$

11,00100100001111110110101001000100001011010001100001000110100110001001100011001100101000001001001110000010001000010011001111110011000111010 0

10^1	$5{,}960 \cdot 10^{-8}$	$2{,}530 \cdot 10^{-10}$	$9{,}239 \cdot 10^{-8}$	$7{,}903 \cdot 10^{-10}$
10^2	$3{,}213 \cdot 10^{-7}$	$6{,}629 \cdot 10^{-8}$	$3{,}823 \cdot 10^{-7}$	$1{,}266 \cdot 10^{-8}$
10^3	$2{,}071 \cdot 10^{-6}$	$3{,}618 \cdot 10^{-6}$	$5{,}866 \cdot 10^{-6}$	$3{,}894 \cdot 10^{-6}$
10^4	$2{,}019 \cdot 10^{-5}$	$7{,}310 \cdot 10^{-5}$	$6{,}112 \cdot 10^{-5}$	$1{,}277 \cdot 10^{-5}$
10^5	$3{,}042 \cdot 10^{-4}$	$4{,}313 \cdot 10^{-4}$	$7{,}257 \cdot 10^{-4}$	$7{,}194 \cdot 10^{-4}$
10^6	$5{,}038 \cdot 10^{-3}$	$3{,}724 \cdot 10^{-3}$	$8{,}233 \cdot 10^{-3}$	$2{,}874 \cdot 10^{-4}$
10^7	$1{,}027 \cdot 10^{-2}$	$8{,}228 \cdot 10^{-2}$	$5{,}657 \cdot 10^{-2}$	$1{,}280 \cdot 10^{-2}$

Tabla 5-4: Error promedio en operaciones sucesivas con el formato IEEE-754 en simple precisión

La figura siguiente ilustra gráficamente los datos de la tabla anterior:

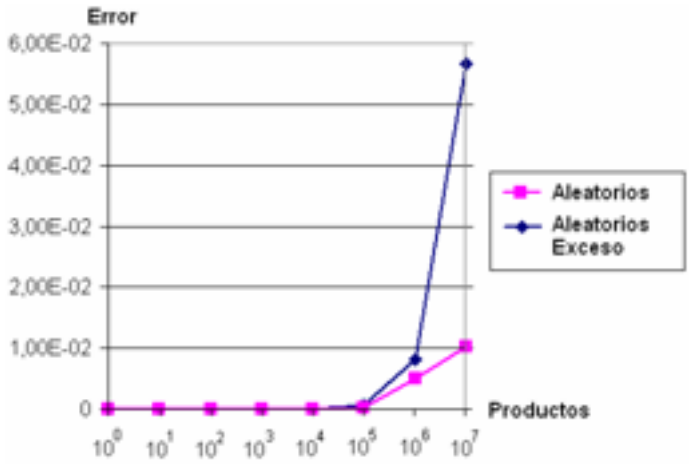


Figura 5-30: Error promedio en operaciones sucesivas con el formato IEEE-754 en simple precisión

Para datos en *doble precisión*:

IEEE-754 doble precisión				
Número de operaciones	Números aleatorios		Números aleatorios con redondeo por exceso	
	Posición	σ	Posición	σ

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001100101000011100000011011010001001010010000001001001110000010001000101001100111111001100011101000

10^0	$5{,}589 \cdot 10^{-17}$	$8{,}582 \cdot 10^{-28}$	$5{,}891 \cdot 10^{-17}$	$1{,}627 \cdot 10^{-32}$
10^1	$2{,}661 \cdot 10^{-16}$	$9{,}187 \cdot 10^{-21}$	$3{,}477 \cdot 10^{-16}$	$3{,}192 \cdot 10^{-20}$
10^2	$7{,}940 \cdot 10^{-16}$	$6{,}425 \cdot 10^{-16}$	$9{,}029 \cdot 10^{-16}$	$2{,}664 \cdot 10^{-16}$
10^3	$1{,}645 \cdot 10^{-15}$	$6{,}151 \cdot 10^{-15}$	$5{,}465 \cdot 10^{-15}$	$4{,}813 \cdot 10^{-15}$
10^4	$2{,}826 \cdot 10^{-14}$	$1{,}917 \cdot 10^{-14}$	$7{,}768 \cdot 10^{-14}$	$6{,}510 \cdot 10^{-14}$
10^5	$3{,}235 \cdot 10^{-13}$	$5{,}433 \cdot 10^{-13}$	$8{,}459 \cdot 10^{-13}$	$1{,}854 \cdot 10^{-13}$
10^6	$7{,}005 \cdot 10^{-12}$	$1{,}087 \cdot 10^{-12}$	$1{,}231 \cdot 10^{-11}$	$8{,}909 \cdot 10^{-11}$
10^7	$6{,}617 \cdot 10^{-11}$	$5{,}983 \cdot 10^{-11}$	$2{,}127 \cdot 10^{-10}$	$2{,}336 \cdot 10^{-10}$

Tabla 5-5: Error promedio en operaciones sucesivas con el formato IEEE-754 en doble precisión

El crecimiento del error se ilustra en la siguiente figura:

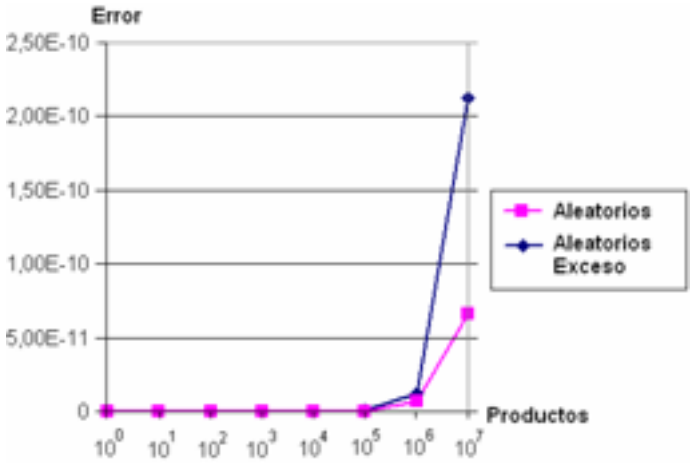


Figura 5-31: Error promedio en operaciones sucesivas con el formato IEEE-754 en doble precisión

Los resultados que se muestran referentes al tamaño del error corroboran las observaciones realizadas sobre la primera posición incorrecta. Con una cantidad reducida de operaciones el error se mantiene en una cota cercana al límite de precisión del formato. Se observa que la desviación que comete el procesamiento de los números con redondeo en exceso está un orden de magnitud por encima de los

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001100101000010001101100001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

otros casos de prueba. Al igual que pasaba en la operación de suma, la no compensación de los restos en el redondeo provoca una mayor desviación respecto al valor exacto.

El error crece a un ritmo proporcional a la cantidad de operaciones encadenadas que se ejecuten. Para una mayor claridad en los resultados la figura 5-32 muestra gráficamente la evolución del error en escala logarítmica.

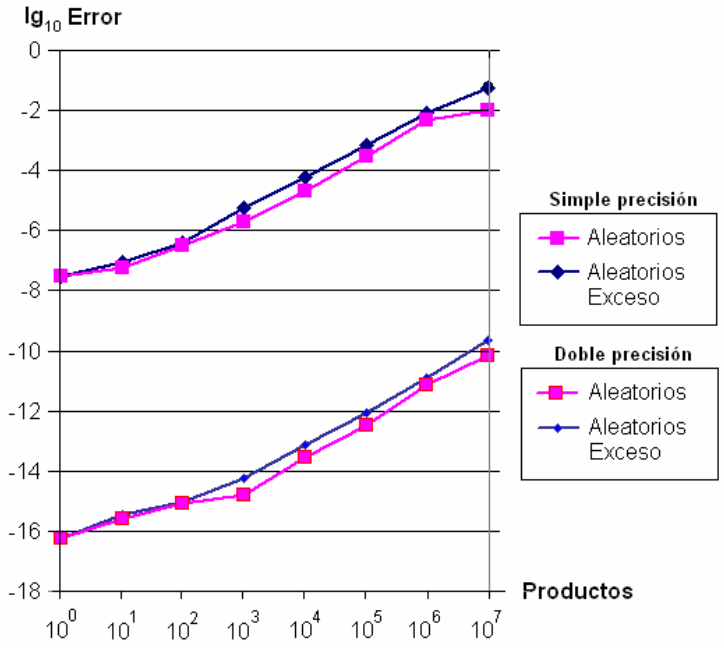
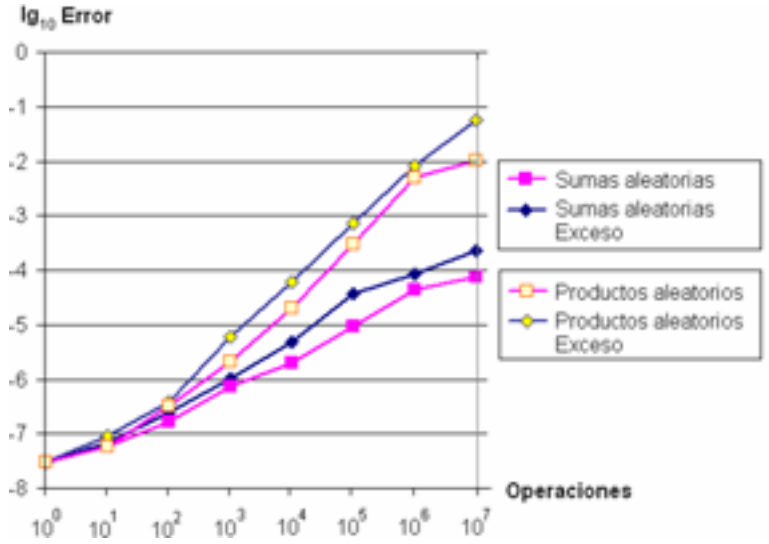


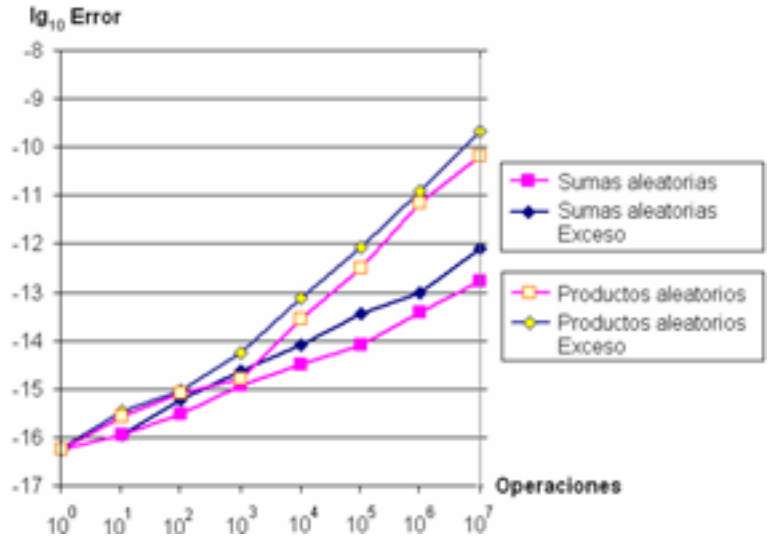
Figura 5-32: Comparación del logaritmo del error promedio de las multiplicaciones en operaciones sucesivas con el formato IEEE-754 en simple y doble precisión

La comparación con la operación de suma pone de manifiesto que el error que se comete en el caso de sucesiones de multiplicaciones es mayor que en el caso de sucesiones de sumas. Las siguientes figuras muestran esta circunstancia en escala logarítmica.

11,00100100001111111011010100010001000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100



(a)



(b)

Figura 5-33: Comparación del logaritmo del error promedio en secuencias de operaciones de suma y producto con el formato IEEE-754. (a) simple precisión; (b) doble precisión

11.00100100001111110110101010001000100001011010001100001000110100110001001100011001100101010001001010010000001001001110000010001
0001010011001111110011000111010 0

Los resultados anteriores muestran los errores que comete el cálculo aritmético mediante el formato de representación IEEE-754. En secuencias de operaciones sucesivas, las desviaciones van adquiriendo progresivamente una dimensión considerable que terminará por afectar a toda la parte significativa del número. Ante esta situación se debe estudiar la conveniencia de utilizar las técnicas convencionales de representación y cálculo en aquellos problemas con exigencias de precisión.

Conclusiones

En este capítulo se ha presentado un procedimiento de cálculo para efectuar la operación de multiplicación sobre operandos racionales del que se extraen las siguientes conclusiones:

- Se realiza una evaluación exacta de la función de multiplicación para números racionales expresados en el formato propuesto.
- La expresión de la complejidad temporal de la operación mantiene una relación cuadrática con el tamaño de los operandos al igual que el algoritmo clásico.
- La operatoria considera la estructura de los factores descrita en la instrumentación de la función identidad. En su procesamiento se hace uso de la metodología de operación desarrollada en este trabajo basada en lógica almacenada y estructuras iterativas.
- El procesamiento contempla elementos de control de la longitud de los resultados. El desarrollo del cálculo para un resultado aproximado es similar al clásico de coma flotante.
- La comparativa con los métodos de multiplicar convencionales pone de manifiesto la cuantía de los errores que éstos producen.

Capítulo VI

Conclusiones

1. Aportaciones
2. Líneas futuras

VI - 1

Aportaciones

La investigación que recoge esta memoria constituye un avance en el desarrollo de modelos de computación sobre números racionales. Se presenta una arquitectura que engloba los elementos necesarios para el procesamiento con control de la precisión que permiten su adecuación a los requerimientos concretos de cada problema. La operatoria contempla tanto el formato de codificación de los números como los procedimientos para operar con ellos.

El trabajo realizado abarca tanto el nivel formal como el procedimental de la solución. Las aportaciones se centran en la formalización de la unidad aritmética y de los métodos de cálculo ajustable, la definición de un esquema de representación numérica posicional con capacidad de expresar de forma exacta los números racionales, el desarrollo de algoritmos de cálculo de operaciones elementales y la concepción de estrategias de construcción de los operadores a bajo nivel para la mejora del rendimiento del procesador especializado.

Se han revisado exhaustivamente las propuestas más novedosas sobre el tratamiento de los problemas con necesidades variables de precisión

11,0010010000111111010101000100010000101101000110000100011010011000100111000101000101110000000011011100000111001101000100101001000000100100111000001000100010010011001111100110001110100

desde la perspectiva de la arquitectura de los computadores. La investigación en ese terreno ha sido prolífica en los últimos años, sin embargo, en lo que se refiere a la representación de números racionales y reales mediante la expresión directa de su valor, se enfrenta frecuentemente con la dificultad para almacenar en un espacio material finito una cantidad infinita de cifras fraccionarias. Como consecuencia de ello la computación aritmética exacta se orienta hacia modelos de cálculo simbólico, expresiones que acotan la precisión o soluciones de alto nivel.

El modelo computacional que se propone se circunscribe en el conjunto de los números racionales. La idea consiste en aprovechar las características matemáticas de su representación fraccionaria y codificar por separado los dígitos fijos y periódicos. Los números racionales disponen de una representación posicional de su valor con una cantidad finita de cifras fraccionarias significativas, lo que permite su codificación en un espacio material de representación finito.

Las funciones aritméticas que engloba el modelo realizan el cálculo sobre valores expresados en el formato de representación propuesto. Su instrumentación se fundamenta en estrategias de procesamiento flexible con elementos de gestión sobre la calidad del resultado que consisten en esquemas iterativos que realizan un procesamiento progresivo de los datos. La estructura de los operadores juega un papel determinante en la aplicación de estos algoritmos, donde la metodología de operación que se propone favorece la aplicación de técnicas de segmentación y reutilización del hardware a la vez que simplifica la lógica de interconexión entre operadores. El diseño basado en memorias con resultados precalculados es una alternativa muy versátil que se ha tenido en cuenta en su construcción.

Se han realizado una serie de experimentos que analizan las características de las funciones que proporciona el modelo, destacándose la capacidad de procesamiento exacto para operandos representados en el formato propuesto. Esta característica se utiliza como referencia para comprobar la capacidad de expresión de formatos y métodos de operación convencionales, en concreto con el estándar de representación IEEE-754.

VI - 2

Líneas futuras

En lo que se refiere a los aspectos pendientes de ser investigados en el contexto de los procesadores de cálculo especializado, se destacan los que se enumeran a lo largo de este apartado.

La representación de la información es un aspecto crucial en el procesamiento de cualquier problema. En este sentido se propone acentuar el esfuerzo investigador en las siguientes cuestiones:

- Estudiar otros métodos de representación de los números racionales que contemplen las ventajas del formato propuesto e incorporen mayor flexibilidad de procesamiento en tiempo y precisión. Los trabajos se pueden dirigir hacia la concepción de esquemas de representación híbridos entre los métodos posicionales y simbólicos que mantengan el equilibrio entre exactitud y complejidad espacial.
- Investigar las posibilidades de extender el espacio de representación a otros conjuntos numéricos sin perder las características de exactitud y gestión de la precisión. La combinación del esquema propuesto con el método de fracciones continuas

11,0010010000111111011010100010001000101100011000010001101001100011000100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

promete ser una interesante técnica para representar valores reales irracionales, en concreto aquellos para los que la codificación mediante fracciones continuas genera cadenas de dígitos periódicas.

En relación con la operatoria de bajo nivel, el procedimiento de cálculo de las funciones aritméticas que se desarrolla en este trabajo se fundamenta en operadores que procesan datos de k bits de forma atómica, vislumbrándose un amplio campo de estudio:

- Análisis del tamaño crítico de los operandos que optimice el rendimiento de los operadores según las características de cada función y las necesidades de cada problema.
- Estudio de implementaciones alternativas de los k-operadores y sus posibilidades de interconexión para constituir operadores de orden superior.
- Concepción de estrategias para reducir la complejidad espacial de los k-operadores. Merecen especial atención las memorias multipuerto, las técnicas de compresión de datos o los mecanismos de direccionamiento reducido.

En cuanto al modelo de procesamiento numérico, la investigación se puede extender a los siguientes ámbitos:

- Incorporar nuevas funcionalidades al modelo aritmético exacto mediante la investigación de métodos de cálculo de otras operaciones en $\mathbb{Q}$ (división, potencia con exponente entero, …).
- Ampliar el modelo a operaciones con resultado irracional (potencia con exponente racional, funciones trigonométricas, logarítmicas, …).
- Dotar al modelo de la capacidad de gestión del tiempo de procesamiento. La posibilidad de producir una cantidad variable de cifras del resultado sugiere la existencia de correspondencias entre el tiempo de cálculo y la precisión obtenida.

Por último, tomar en consideración los múltiples aspectos de la unidad aritmética de coma flotante que quedan pendientes, tanto aquellos de

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

bajo nivel como los de nivel medio-alto que son decisivos a la hora de sacar el máximo partido al procesador.

- Optimizar de las estructuras de memoria y vías de comunicación con ésta para manipular los datos de longitud variable e incorporar estrategias que minimicen la complejidad de los diseños.
- Concebir un módulo de gestión de la precisión que determine la cantidad de cifras del resultado en función de las características del problema. Este módulo deberá relacionar la cantidad de cifras significativas con el grado de aproximación al resultado exacto deseado para determinar el valor que toman los parámetros de ajuste de la precisión de las operaciones aritméticas del modelo.
- Desarrollar compiladores que optimicen el uso de las instrucciones de precisión variable y aritmética exacta.
- Construir un prototipo funcional que contemple todas las cuestiones relevantes y posea la capacidad suficiente para abordar problemas en entornos realistas.

Referencias

A continuación se detallan las reseñas bibliográficas del material que ha servido para la elaboración de este trabajo:

[Aberth y Schaefer, 1992]
O. Aberth, M.J. Schaefer, *Precise Computation Using Range Arithmetic, via C++*, ACM Transactions on Mathematical Software, Vol. 18, no. 5, pp. 481-491, 1992.

[Alefeld y Herzberger, 1983]
G. Alefeld, J. Herzberger, *Introduction to Interval Computations. Computer Science and Applied Mathematics*. Academic Press, 1983.

[Altwaijry, 1997]
H. Altwaijry. *Area and performance Optimiced CMOS Multipliers*. PhD Thesis, Stanford University, 1997.

[Antelo et al, 2002]

E. Antelo, T. Lang, P. Montuschi, A. Nannarelli, *Fast Radix-4 Retimed Division with Selection by Comparisons*, IEEE International Conference on Application-Specific Systems, Architectures, and Processors, pp. 185-196, 2002.

[Arnold et al, 2003]

M.G. Arnold, J.García, M.J. Schulte, *The Interval Logaritmic Number System*, Proceedings of the 16th IEEE Symposium on Computer Arithmetic, 2003.

[Bailey, 1993]

D. H. Bailey. *A Portable High Performance Multiprecision Package*. TR-RNR-90-022, 1993.

[Beaumont-Smith et al, 1998]

A. Beaumont-Smith, N. Burgess, S. Lefrere, C.C. Lim, *Reduced Latency IEEE Floating-Point Standard Adder Architectures*, Proceedings of the 14th IEEE Symposium on Computer Arithmetic, 1998.

[Belski y Kaluzhnin, 1980]

A.A. Belski, L.A. Kaluzhnin, *División inexacta*. Lecciones populares de matemáticas, Ed. Mir, 1980.

[Bennett y Melski, 1995]

S. Bennett, D. Melski. *A Reason to Add Registers*. Technical Report, 1995.

[Benowitz et al, 2002]

E.G. Benowitz, M.D. Ercegovac, F. Fallah. *Reducing the Latency of Division Operations with Partial Caching*. Proc. 36th Asilomar Conference on Signals, Systems and Computers, 2002.

[Bertoni et al, 2003]

G. Bertoni, L. Breveglieri, I. Koren, P. Maistri and V. Piuri, *Error Analysis and Detection Procedures for a Hardware Implementation of the Advanced Encryption Standard*, IEEE Trans. on Computers, pp. 492-505, 2003.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010001001100111111001100011110100

[Bewick, 1994]
G.W. Bewick. *Fast multiplication: Algorithms and implementation*. PhD Thesis, Dept. of Electrical Engineering, Standford University, 1994.

[Bewick y Flynn, 1992]
G. Bewick, M. Flynn. *Bynary multiplication using partially redundant multipliers*. TR CSL-TR-92-528. Computer System Laboratory, Stanford University, 1992.

11,0010010000111111010101000100001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101100

[Bohlender, 1990]
G. Bohlender, *What Do We Need Beyond IEEE Arithmetic?*, Computer Arithmetic and Self-Validating Numerical Methods, C. Ullrich, ed. pp 1-32, Boston: Academic Press, 1990.

[Bohlender, 1991]
Gerd Bohlender, *Decimal Floating-Point Arithmetic in Binary Representation*, Computer arithmetic: Scientific Computation and Mathematical Modelling, 1991.

[Boldo y Daumas, 2003]
S. Boldo and M. Daumas, *Representable correcting terms for possibly underflowing floating point operations*, Proceedings of the 16th IEEE Symposium on Computer Arithmetic, 2003.

[Brent y Kung, 1982]
R.P. Brent, H.T. Kung, *A regular layout for parallel adders*, IEEE Transactions on Computers, C-31(3): 260-264, 1982.

[Brezinski, 1980]
C. Brezinski, *History of Continued Fractions and Pade Approximants*. Springer-Verlag, 1980.

[Bruguera y Lang, 1999]
J.D. Bruguera, T. Lang. *Leading-One Prediction with Concurrent Position Correction*, IEEE Transactions on Computers, Vol 48, no. 10, 1999.

[Bruguera y Lang, 2000]
J.D. Bruguera, T. Lang, *Rounding in Floating-Point Addition using a Compound Adder*, Internal Report, University of Santiago de Compostela, Spain, 2000.

[Bruguera y Lang, 2001]
J. D. Bruguera, T. Lang. *Multilevel Reverse Most--Significant--Carry Computation*. IEEE Transactions on Very Large Scale of Integration (VLSI) Systems. Vol. 9. no. 6. 2001.

11,00100100001111110110101000100010000101101000110000100011010011000100110001100011001010001011100000001101110000011100110100010010010000001001001110000010001
000101001100111111001100011101000

[Buchberger, 1991]
B. Buchberger, *Groebner Bases in MATHEMATICA: Enthusiasm and Frustation*, Programming Environments for High-level Scientific Problem Solving, pp. 80-91, 1991.

[Carr, 1993]
S. Carr. *Memory Hierarchy Management*. PhD Thesis, Rice University, 1993.

[Chen et al, 2000]
F. Cheng, S.H. Unger, M. Theobald. *Self-Timed Carry-Lookahead Adders*. IEEE Transactions on Computers, Vol. 48, no. 7, pp. 659-672, 2000.

[Chen-Ying, 1996]
H. Chen-Ying. *Variable Precision Arithmetic Processor in FPGAs.* Master's Thesis, University of Toronto, 1996.

[Cilio y Corporal, 1999]
A. G. M. Cilio, H. Corporal, *Floating Point to Fixed Point Conversion of C Code*, Computational Complexity, pp 229-243, 1999

[Clark, 1998]
D. Clark, *Supercomputing: The Next Generation*, IEEE Computational Science and Engineering, Vol. 5, no. 5, pp. 79-81, 1998.

[Cohen et al, 1983]
M.S. Cohen, T.E. Hull, V.C. Hamacher, *CACAD: A Controlled-Precision Decimal Aritmetic Unit*, IEEE Transactions on Computers, Vol C-32, pp 370-377, 1983

[Cowlishaw, 2003]
M. F. Cowlishaw, *Decimal Floating-Point: Algorism for Computers,* Proceedings of the 16th IEEE Symposium on Computer Arithmetic, pp. 104-111, 2003.

[Conte et Al, 1997]

T.M. Conte, P.K. Dubey, M.D. Jennings, R.B. Lee, A. Peleg, S. Rathnam, M. Schlansker, P.Song, A. Wolfe, *Challenges to Combining General-Purpose and Multimedia Processors*, IEEE Computer, pp 33-37, 1997.

[Dadda, 1965]
L. Dadda. *Some Schemes for Parallel Multipliers*, Alta Frequenza, Vol. 34, 1965.

[Dekker, 1971]
T. Dekker., *A Floating Point Technique for Extending the Avaliable Precision*, Numerische Mathematik. Vol. 18 pp. 224-242, 1971.

[Ercegovac y Trivedi, 1977]
M.D. Ercegovac, K.S. Trivedi, *On-line Arithmetic for Division and Multiplication*, IEEE Transactions on Computers, Vol. 26, no.7, pp. 681-687, 1977.

[Ercegovac y Trivedi, 1987]
M.D. Ercegovac, K. Trivedi, *On-line Operations*, IEEE Transactions on Computers, Vol 36, no. 7, pp. 895-897, 1987.

[Ercegovac et al, 2000a]
M.D. Ercegovac, T. Lang, J.-M. Muller, A. Tisserand. *Reciprocation, Square Root, Inverse Square Root, and Some Elementary Functions Using Small Multipliers*. IEEE Trans. Computers, Vol. 49(7), pp. 628-637, 2000.

[Ercegovac et al, 2000b]
M.D. Ercegovac, L. Imbert, D.W. Matula, J. Muller, G. Wei. *Improving Goldschmidt Division, Square Root, and Square Root Reciprocal*. IEEE Transaction on Computers, Vol. 49, nº 7, 2000.

[EU, 1999]
European Commission Directorate General II, *The introduction of the Euro and the Rounding of Currency Amounts*, II/28/99-EN Euro Papers n. 22, DGII/C-4-SP(99), 1999.

[Even et al, 2003]

11,00100100001111110110101000100010000101101000110000100011010011000100110001100011001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010100

G. Even, P-M. Seidel, W.E. Ferguson, *A Parametric Error Analysis of Goldschmidt's Division Algorithm*, Proceedings of the 16th IEEE Symposium on Computer Arithmetic, 2003.

[Even y Müller, 2000]
G. Even, S.M. Müller, P-M. Seidel. *A Dual Precision IEEE Floating-Point Multiplier*. Integration, The VLSI Journal, Vol. 29, Issue 2, 2000.

[Even y Seidel, 2000]
G. Even, P. Seidel. *A comparison of Three Rounding Algorithms for IEEE Floating-Point Multiplication*. IEEE Transactions on Computers, Vol. 49, no 7, pp. 638-650, 2000.

[Fahmy y Flynn, 2003]
A.H. Fahmy, M. J. Flynn. *The Case for a Redundant Format in Floating Point Arithmetic*. Proceedings of the 16th IEEE Symposium on Computer Arithmetic, 2003.

[Ferreira, 1998]
A. Ferreira Tenca, *Variable Long-Precision Arithmetic (VLPA) for Reconfigurable Coprocessor Architectures*, PhD Thesis, University of Los Angeles, 1998.

[García et al, 2003a]
Juan Manuel García Chamizo, Jerónimo Mora Pascual, Higinio Mora Mora, Mª Teresa Signes Pont, *Time-Precision Flexible Arithmetic Unit*, XVIII Conference on Design of Circuits and Integrated Systems (DCIS), 2003.

[García et al, 2003b]
Juan Manuel García Chamizo, Jerónimo Mora Pascual, Higinio Mora Mora, Mª Teresa Signes Pont, *Calculation Methodology for Flexible Arithmetic Processing*, IFIP International Conference on Very Large Scale Integration (VLSI-SoC), 2003.

[García et al, 2003c]

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101000

Juan Manuel García Chamizo, Jerónimo Mora Pascual, Higinio Mora Mora, *Time-Precision Flexible Adder*, IEEE Internacional Conference on Electronics, Circuits and Systems (ICECS), 2003.

[Gianantonio, 1993]
P.D. Gianantonio, *A Functional Approach to Computability on Real Numbers*, Ph. D. Thesis, Università di Pisa-Genova-Udine. 1993.

[Goldberg, 1967]
I.B. Goldberg, *27 Bits Are Not Enough for 8-Digit Accuracy*. Communications of the ACM. 10(2), pp 105-106, 1967.

[Goldberg, 1990]
D. Goldberg. *Computer Architecture A Quantitative Approach*, Chapter Appendix A. Morgan Kaufmann, 1990.

[Goldberg, 1991]
D. Goldberg, *What every Computer Scientist Should Know About Floating-Point Arithmetic*, Computer Surveys, Vol. 23, no. 1, pp. 5-48, 1991.

[Guyot et al, 1989]
A. Guyot, Y. Herreros, J.M. Muller, *JANUS, an On-line Multiplier/divider for manipulating large numbers*, Proceedings of the 9th IEEE Symposium on Computer Arithmetic, pp. 106-111, 1989.

[Guelfond, 1979]
A.O. Guelfond, *Resolución de ecuaciones en números enteros*, Lecciones populares de matemáticas, Ed. Mir, 1980.

[Hehner y Horspool, 1979]
E.C.R. Hehner, R.N.S. Horspool, *A New Representation of the Rational Numbers for fast Easy Arithmetic*, SIAM J. Computing, Vol. 8, no. 2, 1979.

[Herring, 2000]
C. Herring, *Microprocessors, Microcontrollers, and Systems in the New Millenium*, IEEE Micro, Vol. 20, no. 6, pp. 45-51, 2000

11,00100100001111110110101000100010000101101000110000100011010011000100110001100010100010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

[Hoffmann, 1989]
C.M. Hoffmann. *The Problems of Accuracy and Robustness in Geometric Computation*, IEEE Computer, Vol. 22, no. 3, pp. 31-40, 1989.

[Hokenek y Montoye, 1990]
E. Hokenek, G. R. Montoye. *Leading-zero anticipator (lza) in the IBM risc system/6000 floating point execution unit*, IBM J. Res. Develop., pp 71-77, 1990.

[Hormigo et al, 1999]
J. Hormigo, J. Villalba, E.L. Zapata, *Interval Sine and Cosine Functions Computation Based on Variable-Precision CORDIC Algorithm*, 14th IEEE Symposium on Computer Arithmetic, 1999.

[Hormigo el al, 2000]
J. Hormigo , J. Villalba, M.J. Schulte, *Hardware Algorithm for Variable-Precision Division*, 4th Conf. on Real Numbers and Computers, 2000.

[Hsu, 1996]
C.-Y. Hsu, *Variable Precision Arithmetic Processor in FPGAs*, Master's Thesis, University of Toronto, 1996.

[Hull et al, 1991]
T.E. Hull, M.S. Cohen, C.B. Hull, *Specification for a Variable-Precision Arithmetic Coprocessor*, Proceedings of the 10th Symposium on Computer Arithmetic, pp. 127-131, 1991.

11,0010010000111111010101000100001000010110100011000010001101001100010011000110100010111000000011011100000111001101000100101001000000100100111000001000100010100111001111100110001110100

[IEEE, 1985]
American National Standars Institute and Institute of Electrical and Electronic Engineers. *IEEE Standard for Binary Floating-Point Arithmetic. ANSI/IEEE Standard 754,* 1985.

[IEEE, 1987]
American National Standars Institute and Institute of Electrical and Electronic Engineers. *IEEE Standar for Radix-Independent Floating-Point Arithmetic. ANSI/IEEE Standard 854*, 1987.

[IEEETC, ISSN: 0018-9340]
IEEE Transactions on Computers, ISSN: 0018-9340, IEEE Computer Society, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA, 90720-1314, Periodicidad mensual.

[IEEEVLSI, ISSN:1063-8210]
IEEE Transactions on Very Large Scale Integration Systems, ISSN: 1063-8210, IEEE Computer Society, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA, 90720-1314, Periodicidad trimestral.

[Inacio y Ombres, 1996]
C. Inacio, D. Ombres, *The DSP decision: fixed point or floating point?*, IEEE Spectrum, Vol 33, 1996.

[Ing-Jer y Tzu-Chin, 1998]
H. Ing-Jer, P. Tzu-Chin. *Analysis of x86 Instruction Set Usage for DOS/Windows Applications and Its Implication on Superscalar Design*. Proc. of the International Conference on Computer Design. IEEE, 1998.

[Ito et al, 1997]
M. Ito, N. Takagi, S. Yagima. *Efficient Initial Approximation for Multiplicative Division and Square Root by a Multiplication with Operand Modification.* IEEE Transactions on Computers, Vol. 46, nº 4. 1997.

[JACM, ISSN: 0004-5411]

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101100

Journal of the ACM, ISSN: 0004-5411, Association Computing Machinery, 1515 Broadway, New York, NY, 10036. Periodicidad trimestral.

[JCSC, ISSN: 0218-1266]
Journal of Circuit Systems and Computers, ISSN: 0218-1266, World Scientific Publication. Journal Dept. PO Box 128 Farrer Roadsingapore. Singapore, 912805. Periodicidad trimestral.

[JCSS, ISSN: 0022-0000]
Journal of Computer and System Sciences, ISSN: 0022-0000, Academic Press Inc. 525 B ST, STE 1900, San Diego, CA, 92101-4495. Periodicidad bimensual.

[JSC, ISSN: 1383-7621]
Journal of Systems Architecture, ISSN: 1383-7621, Elsevier Science BV, PO Box 211, Amsterdam, Netherlands, 1000 AE. Periodicidad mensual.

[Kaihara y Takagi, 2003]
M.E. Kaihara, N. Takagi, *A VLSI Algorithm for Modular Multiplication/Division*, Proceedings of the 16th IEEE Symposium on Computer Arithmetic, 2003.

[Karatsuba y Ofman, 1963]
A. Karatsuba, Y. Ofman. *Multiplication on Multidigit Numbers on Automata*, Soviet Phys. Doklady, Vol. 7, no. 7, 1963.

[Kim et al, 1998]
S. Kim, KI. Kum y W. SPNG, *Fixed-Point Optimization Utility for C and C++ Based Digital Processing Programs*, IEEE Transactions on Circuits and Systems-II: Analog and digital signal processing, Vol. 45, no. 11, 1998.

[Klatte et al, 1991]
R. Klatte, U. Kulisch , M. Neaga, D. Ratz, Ch. Ullrich, *PASCAL – XSC: Language Reference with Examples,* 1991.

[Kloos et al, 2002]

11,0010010000111111010101000100001011010001100001000110100110001001100011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111010 0

H. Kloos, J.P. Wittenburg, W. Hinrichs, H. Lieske, L. Friebe, C. Klar, P. Pirsch, *HIPAR-DSP 16, a scalable highly parallel DSP core for system on chip video and image processing applicantions,* IEEE International Conference on Acoustics, Speech, and Signal Processing, 2002

[Kneip et al 1994]
J. Kneip, K. Rönner, P. Pirsch, *A Data Path Array with Shared Memory as Core of a High Performance DSP*, International Conference on Application Specific Array Processors, 1994.

[Knuppel, 1994]
O. Knuppel, *PROFIL/BIAS – A Fast Interval Library*, Computing, Vol. 53, ns. 3-4, pp. 277-288, 1994.

[Kolla et al, 1999]
R. Kolla, A. Vodopivec, J. Wolf, V. Gudenberg, *The IAX Architecture: Interval Arithmetic Extension*, Tech. Report, University of Würzburg,1999.

[Koren et al, 2003]
I. Koren, Y. Koren, B. Oomman, *Saturating Counters: Application and Design Alternatives*, Proceedings of the 16th IEEE International Symposium on Computer Arithmetic, 2003.

[Kornerup, 1994]
P. Kornerup, *Digit-Set Conversions: Generalizations and Applications*, IEEE Transactions on Computers, Vol. 43, no. 5, pp. 622-629, 1994.

[Kornerup, 2003]
P. Kornerup. *Revisiting SRT Quotient Digit Selection*, Proceedings of the 16th IEEE Symposium on Computer Arithmetic, 2003.

[Kornerup y Matula, 1983a]
D. Matula, P. Kornerup. *Finite Precision Rational Arithmetic: Slash Number Systems*, IEEE Transactions on Computers, Vol. C-32 pp. 3-18, 1983.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010100110011111001100011101000

[Kornerup y Matula, 1983b]
D. Matula, P. Kornerup, *Finite Precision Rational Arithmetic: An Arithmetic Unit*, IEEE Transactions on Computers, Vol. C-32 pp. 378-387, 1983.

[Kornerup y Matula, 1991]
P. Kornerup, D. Matula, *Algorithms for Arbitrary Precision Floating Point Arithmetic*, Proceedings of the 10th IEEE Symposium on Computer Arithmetic, 1991.

[Kum et al, 1997]
KI Kum, J. Kang y W. Sung, *A Floating-point to Fixed-point C Converter for Fixed-point Digital Signal Processors*, Proceeding of the 2nd SUIF Compiler Workshop, 1997.

[Lakamraju et al, 2002]
V. Lakamraju, I. Koren and C.M. Krishna, *Filtering Random Networks to Synthesize Interconnection Networks with Multiple Objectives*, IEEE Transactions on Parallel and Distributed Systems, pp. 1139-1149, 2002.

11,00100100001111110110101000100001000010110100011000010001101001100010011000110100101000101110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101 00

[Lang y Antelo, 2001]
T. Lang and E. Antelo, *Correctly Rounded Reciprocal Square-root by Digit Recurrence and Radix-4 Implementation*, Proceeding of the 15th IEEE Symposium on Computer Arithmetic, 2001.

[Lefevre y Muller, 2003]
V. Lefevre and J.M. Muller, *On-the-fly Range Reduction*, Journal of VLSI Signal Processing, pp. 31-35, 2003.

[Liddicoat y Flynn, 2001]
A. A. Liddicoat, M. J. Flynn. *High-Performance Floating Point Divide*, Euromicro Symposium on Digital System Design, September 2001.

[Loeckx y Sieber, 1984]
J. Loeckx y K. Sieber, *The Foundations of Programs Verification*, Ed. John Wiley & Sons Ltd. Londres, 1984.

[McIlhenny y Ercegovac, 1999]
R. McIlhenny, M.D. Ercegovac, *On the Design of an On-line FFT Network for FPGA's*. 33rd Asilomar Conference on Signals, Systems and Computers, 1999.

[Mencer et al, 1999]
O. Mencer, M. Morf, M. J. Flynn, *Precision of Semi-Exact Redundant Continued Fraction Arithmetic for VLSI*, SPIE '99 ,Arithmetic session, 1999.

[Mencer, 2000]
O. Mencer. *Rational Arithmetic Units in Computer Systems*, PhD Thesis, Stanford University, 2000.

[Michelucci y Moreau, 1997]
D. Michelucci, J-M. Moreau, *Lazy Arithmetic*, IEEE Transactions on Computers, Vol. 46, no. 9, pp. 961-975, 1997.

[Montuschi y Lang, 1999]

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110100010110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

P. Montuschi, T. Lang, *Very High Radix Square Root with Prescaling and Rounding and a Combined Division/Square Root Unit*. IEEE Transactions on Computers, Vol. 48, no. 8, pp. 827-841, 1999.

11,0010010000111111011010100010000101101000110000100011010011000100111000110011000101000101110000000110111000001110011010001001010010000001001001110000010001
0001010011001111110011000111101 00

[Montuschi y Lang, 2001]
P. Montuschi, T. Lang, *Boosting Very-High Radix Division with Prescaling and Selection by Rounding*. IEEE Transactions on Computers Vol. 50, no. 1, pp. 13-27, 2001

[Moore, 1964]
C. D. Moore, *An Introduction to Continued Fractions*. The National Council of Teachers of Mathematics, 1964

[Moore, 1979]
R.E. Moore. *Methods and Applications of Interval Analysis*, Studies an Applied Mathematics, 1979.

[Mora, 2001]
J. M. Mora Pascual, *Unidades Aritméticas en Coma Flotante para Tiempo Real*, PhD Thesis, Universidad de Alicante, 2001.

[Muller, 1991]
J.M. Muller, *On-line Computations: a survey and some new results*, IFIP Workshop on Algorithms and Parallel VLSI Architectures, 1991.

[Muller, 2003]
J-M. Muller, *Partially rounded Small-Order Approximations for Accurate, Hardware-Oriented, Table-Based Methods*, Proceedings of the 16th IEEE Symposium on Computer Arithmetic, 2003.

[Nambu et al, 1998]
H. Nambu, K. Kanetu, K.Higeta, M. Usami, T. Kusonoki, K. Yamaguchi, N. Homma. *A 1,8ns Access, 550 Mhz 4,5Mb CMOS SRAM*. IEEE International Solid-State Circuit Conference, 1998.

[Nedovic et al, 2002]
N. Nedovic, W. W. Walker, V. G. Oklobdzija, M. Aleksic, *A Low Power Symmetrically Pulsed Dual Edge-Triggered Flip-Flop*, Proceedings of the 28th European Solid-State Circuits Conference, 2002.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010100100000010010011100000100010001001100111110011000111010 0

[Neumaier, 1990]
A. Neumaier, *Interval Methods for Systems of Equations*. Cambridge University Press. 1990.

11,0010010000111111011010100010001000010110100011000010001101001100010011000110100010111000000011011100000111001101000100101001000000100100111000001000100010100111001111100110001110100

[Neumann, 1999]
P. G. Neumann, *Practical Architectures for Survivable Systems and Networks*, Army Research Lab., 1999.

[Nielsen, 1997]
A. M. Nielsen, *Number systems and Digit Serial Arithmetic*, PhD Thesis, Odense University, 1997.

[Nielsen et al, 2000]
A.M. Nielsen, D.W.Matula, C.N. Lyu, G. Even, *An IEEE Compliant Floating-Point Adder that Conforms with the Pipelined Packet-Forwarding Paradigm*, IEEE Transactions on Computers, Vol 49, no.1, pp. 33-47, 2000.

[Nielsen y Kornerup, 1999]
A. M. Nielsen, P. Kornerup, *Redundant Radix Representations of Rings*, IEEE Transactions on Computers, Vol 48, no. 11, pp. 1153 – 1165, 1999.

[Oberman, 1996]
S.F. Oberman. *Design Issues in High Performance floating Point Arithmetic Units*. TR CSL-TR-96-711. Computer System Laboratory, Stanford University, 1996.

[Oberman et al, 1998]
S. Oberman, F. Weber, N.Juffa, G. Favor, *AMD 3DNow! Technology and the K6-2 Microprocessor*, Proceedings of Hot Chips 10, pp. 245-254, 1998.

[Oklobdzija et al, 1996]
V.G. Oklobdzija, D. Villeger, S.S. Liu. *A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach*. IEEE Transactions on Computers. Vol. 45, pp. 294-306, 1996.

[Oklobdzija et al, 2003]
V. G. Oklobdzija, B. R. Zeydel, H. Dao, S. Mathew, R. Krishnamurthy, *Energy-Delay Estimation Technique for High-Performance Microprocessor*

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

VLSI Adders, Proceedings of the 16th IEEE Symposium on Computer Arithmetic, 2003.

[Omondi, 1994]
A. Omondi. *Computer Arithmetic Systems, Algorithms, Architecture and Implementations*. Prentice-Hall, 1994.

[Parhami, 2000]
B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*, Oxford University Press, 2000.

[Parhi, 1997]
K.K. Parhi, *Fast Low-energy VLSI Binary Addition*, Proceedings of International Conference on Computer Design ICCD, pp. 676-684, 1997.

[Park et al, 1999]
W.C. Park, T.D. Han, S.D. Kim, S.B. Yang. *A Floating point Multiplier Performing IEEE Rounding and Addition in Parallel*. Journal of Systems Architecture, Vol. 45. no. 14, 1999.

[Patterson y Hennessy, 2002]
D.A. Patterson, J.L. Hennessy, *Computer Architecture a quantitative approach*, Morgan Kaufmann Publishers, 2002.

[Paul y Seidel, 2003]
W. J. Paul, P-M. Seidel, *To Booth or Not To Booth?*, Integration, the VLSI journal, 2003.

[Paulin et al, 2001]
P. Paulin, F. Karim and P. Bromley, *Network Processors: A Perspective on Market Requirements, Processor Architectures and Embedded S/W Tools*, Proceedings of the Design, Automation and Test in Europe, 2001.

[Peleg y Weiser, 1996]
A. Peleg y U. Weiser, *MMX Technology Extension to the Intel Architecture*, IEEE Micro July/August, pp 42-50, 1996.

[Piñeiro y Bruguera, 2002]
J.-A. Piñeiro, J.D. Bruguera. *High-Speed Double-Precision Computation of Reciprocal, Division, Square Root and Inverse Square Root*. IEEE Transactions on Computers, Vol. 51, No. 12. 2002.

[Piso et al, 2002]
D. Piso, J.-A. Piñeiro, J.D. Bruguera. *Analysis of the Impact of Different Methods for Division/Square Root Computation in the Performance of a Superscalar Microprocessor*. Proceedings of EUROMICRO Symposium on Digital System Design. 2002.

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110100010110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

[Press et al, 1994]
W. H. Press et Al, *Numerical Recipes in C: the art of Scientific Computing*, Cambridge University Press, 2nd edition, 1994.

[Priest, 1991]
D.M. Priest, *Algorithms for Arbitrary Precision Floating Point Arithmetic*, Proceedings of the IEEE 10th Symposium of Computer Arithmetic, pp. 132-143, 1991.

[Quach et al, 1991]
N. Quach, N. Takagi and M. Flynn, *On Fast IEEE Rounding*, Technical Report CSL-TR-91-459, Stanford University, 1991.

[Quach y Flynn, 1990]
N. Quach, M.J. Flynn, *An improved Algorithm for High-Speed Floating-Point Addition*, Technical Report CSL-TR-90-442, Stanford University, 1990.

[Ratz, 1990]
D. Ratz, *The Effects of the Arithmetic of Vector Computers on Basis Numerical Methods*, Computer Arithmetic and Self-Validating Numerical Methods, C. Ullrich, pp 499-514, Academic Press, 1990.

[Robertson y Trivedi, 1977]
J.E. Robertson, K.S. Trivedi, *On the Use of Continued Fractions for Digital Computer Arithmetic*. IEEE Transactions on Computers, 1977.

[Sáez et al, 1998]
E. Sáez, J. Villalba, J. Hormigo, F.J. Quiles, J.I. Benavides, E.L. Zapata, *FPGA Implementation of a Variable Precision CORDIC processor*,13th Conf. on Design of Circuits and Integrated Systems (DCIS'98), 1998.

[Sanna et al, 1998]
A. Sanna, P. Montuschi, M. Rossi, *A Flexible Algorithm for Multiprocessor Ray Tracing*, The computer journal, Vol. 41, No. 7, pp. 503-516, 1998.

11,0010010000111111010101000100010000101101000110000100011010011000100111000110011001010001011100000001101110000011100110100010010100100000010010011100000100010001001100111110011000111010 0

[Schneider et al, 2000]

A. Schneider, R. McIlhenny, M.D. Ercegovac, *BigSky - An On-Line Arithmetic Design Tool for FPGAs*, IEEE Symposium on Field-Programmable Custom Computing Machines, 2000.

11,00100100001111110110101000100010000101101000110000100011010011000100110001100010100010111000000011011100000111001101000100101001000000100100111000001000100010011001111110011000111010 0

[Schönhage y Strassen, 1971]
A. Schönhage, V. Strassen. *Schnelle Multiplikation grosser Zahlen*, Computing, Vol. 7, pp. 281-292, 1971.

[Schulte, 1994]
M.J. Schulte. *Optimal initial approximations for the Newton-Raphson division algorithm*. Computing, Vol. 53. 1994.

[Schulte, 1996]
M.J. Schulte, *A Variable-Precision, Interval Arithmetic Processor*, PhD Thesis, University of Texas, 1996.

[Schulte, 2000]
M.J. Schulte, *A Family of Variable-Precision Interval Arithmetic Processors*, IEEE Transactions on Computers, Vol. 49, no. 5, pp. 1-11, 2000.

[Schulte et al, 1999]
M.J. Schulte, A. Akkas, V. Zelov, J.C. Burley, *The Interval Enhanced GNU Fortran Compiler*, Reliable Computing, Vol. 5, no. 3, pp. 311-322, 1999.

[Schulte et al, 2000]
M. J. Schulte, P. I. Balzola, A. Akkas, R. W. Brocato, *Integer Multiplication with Overflow Detection or Saturation*, IEEE Transactions on Computers, Vol. 49, 2000.

[Schulte y Swartzlander, 1995]
M.J. Schulte, E.E. Swartzlander, Jr. *A Processor for Staggered Interval Arithmetic*, Proceedings of the 1995 International Conference on Application Specific Array Processors, pp. 104-112, 1995.

[Schulte y Swartzlander, 2000]
M. J. Schulte, E. E. Swartzlander, *A Family of Variable-Precision, Interval Arithmetic Processors*, IEEE Transactions on Computers, Vol. 49, 2000.

[Seidel et al, 2001]

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110100010110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101 00

P-M. Seidel, L. McFearin, D.W. Matula, *Binary Multiplication Radix-32 and Radix-256*, Proceedings of the 15th IEEE Symposium on Computer Arithmetic, 2001.

11,00100100001111110110101010001000100001011010001100001000110100110001001100011001010001011100000001101110000011100110100010010100100000010010011100000100010001010011001111110011000111010 0

[Seidensticker, 1983]
R.B. Seidensticker, *Continued fractions for high-speed and high-accuracy computer arithmetic*. Proceeding of the 6th IEEE Symposium on Computer Arithmetic, 1983.

[Slater, 1996]
M. Slater, *The Microprocessor Today*, IEEE Micro, Vol. 16, no. 6, pp. 32-44, 1996.

[Smith, 1991]
D.M. Smith, *A Fortan Package for Floating-Point Multiple-Precision Arithmetic*, ACM Transactions on Mathematical Software, Vol. 17, no. 2, pp. 273-283, 1991.

[Smith y Weingart, 1999]
W. S. Smith, S. Weingart, *Building a High-Performance, Programmable Secure Coprocessor*, Computer Networks, Vol. 31, No. 9, 1999.

[Srinivas y Parhi, 1992]
H.R. Srinivas, K.K. Parhi, *A Fast VLSI Adder Architecture*, IEEE Journal of Solid-State Circuits, SC-19 (5), pp. 761-767, 1992.

[Sterbenz, 1974]
P. Sterbenz, *Floating-Point Computation*, Prentice-Hall, Englewood Cliffs, New Jersey, 1974.

[Sun Microsystems, 2000]
Numerical Computation Guide, Sun Microsystems, 2000.

[Suzuki et al, 1996]
H. Suzuki, H. Morinaka, H.Makino, Y. Nakase, K. Mashiko, T. Sumi, *Leading-Zero Anticipatory Logic for High Speed Floating Point Addition*, IEEE J. Solid-State Circuits, Vol. 31, no. 8, pp. 1157-1164, 1996.

[Takagi, 1998]

N. Takagi, *A VLSI algorithm for Modular Division based on the Binary GCD algorithm*, IEICE Trans. Fundamentals, Vol. E81-A, no.5, 1998.

[Takagi y Horiyama, 1999]
N. Takagi, T. Horiyama. *A High-Speed Reduced-Size Adder Under Left-ro-Right Arrival*. IEEE Transactions on Computers, Vol. 48, no. 1, pp. 76-80, 1999.

[Tang, 1991]
P.T.P. Tang, *Table-lookup Algorithms for Elementary Functions and Their Error Analysis*, Proc. 10th Symposium on Computer Arithmetic, pp. 232-236, 1991.

[Tenca y Ercegovac, 1998]
A. Tenca, M.D. Ercegovac. *A variable long-precision arithmetic unit design for reconfigurable coprocessor architectures*. Proceeding of IEEE Symposium on FPGAs for Custom Computing Machines, 1998.

[Thakkar y Huff, 1999]
S. Thakkar, T. Huff, *The Internet Streaming SIMD Extensions*, Intel Technology Journal, Vol. Q2, 1999.

[Toom, 1963]
A. L. Toom. *The complexity of a scheme of functional elements realizing the multiplication of integers*. Soviet Math, Vol. 3, 1963.

[Turing, 1937]
A.M. Turing, *On computable Numbers, with an Application to the Entscheidungs Problem*, Proc. London Math. Soc. 42, pp. 230-265, 1937.

[Varga, 1990]
R.S. Varga, *Scientific Computation on Mathematical Problems and Conjectures*, *SIAM*, Philadelphia, 1990.

[Villalba et al, 2002]
J. Villalba Moreno, G. Bandera Burgueno, M. A. González Peñalber, J. Hormigo Aguilar. *Polynomial Evaluation on Multimedia Processors,*

11,0010010000111111011010100010001000010110100011000010001101001100010011000110001010001011100000001101110000011100110100010010010000001001001110000010001
0001010011001111110011000111010 0

International Conference on Application-specific Systems, Architectures and Processors, 2002.

[Vuillemin, 1990]
J.E. Vuillemin, *Exact Real Computer Arithmetic with Continued Fractions*, IEEE Transactions on Computers, Vol 39, pp. 1087-1105, 1990.

[Wada et al, 1992]
T. Wada, S. Rajan, S.A. Przybylski, *An Analytical Access Time Model for On-Chip Cache Memories*. IEEE Journal of Solid-State Circuits, Vol. 27, no. 8, 1992.

[Wallace, 1964]
C.S. Wallace. *A Suggestion for a Fast Multiplier*, IEEE Transactions on Computers, Vol. 13, no. 2, 1964.

[Wei y Thompson, 1990]
B.W.Y. Wei, C.D. Thompson. *Area-time Optimal Adder Design*. IEEE Transactions on Computers, Vol. 39, no. 5, 1990.

[Wen-Chang y Chein-Wei, 2000]
Y. Wen-Chang, J. Chein-Wei. *High-Speed Booth Encoded Parallel Multiplier Design*. IEEE Transactions on Computers, Vol. 49, no. 7, 2000.

[Wilkinson, 1964]
J. Wilkinson., *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, New Jersey, 1964.

[Wilton y Jouppi, 1994]
S.J.E. Wilton, N.P. Jouppi. *An Enhanced Access and Cycle Time Model for On-Chip Caches*. Digital Western Research Laboratory, 2000.

[Wires, 2000]
K. E. Wires, *Arithmetic Units for Digital Signal Processing and Multimedia*, PhD Thesis, Lehigh University, 2000.

[Wolf y Franklin, 2002]
T. Wolf, M. Franklin, *Design Tradeoffs for Embedded Network Processors*, International Conference on Architecture of Computing Systems, 2002.

[Wong y Goto, 1995]
W.F. Wong and E. Goto, *Fast Evaluation of the Elementary Functions in Simple Precision*, IEEE Trans. Computers, Vol. 44, no 3, pp 453-457, 1995.

[Xiaoning et al, 1999]

11,00100100001111110110101000100010000101101000110000100011010011000100110001100110100010110000000110111000001110011010001001010010000001001001110000010001
000101001100111111001100011101100

N. Xiaoning, L. Gazsi, F. Engel, G. Fettweis, *A new network processor architecture for high-speed communications*, IEEE Workshop on Signal Processing Systems, 1999.

[Yee y Tygar, 1995]
B.S. Yee, J.D. Tygar, *Secure Coprocessors in electronic Commerce Applications*, The First USENIX Workshop on Electronic Commerce, 1995.

[Yu et al, 2003]
X. Y. Yu, V. G. Oklobdzija, W. W. Walker, *An Efficient Transistor Optimizer for Custom Circuits*, International Symposium on Circuits and Systems, 2003.

[Zimmermann, 1997]
H. Zimmermann. *Binary Adder Architectures for Cell-Based VLSI and their Syntesis*. PhD Thesis, Swiss Federal Institute of Technology, 1997.

[Zuras, 1994]
D. Zuras. *More on Squaring and Multiplying Large Integres*. IEEE Transactions on Computers, Vol. 43, no. 8, 1994.

Resumen

Este trabajo presenta la concepción y el desarrollo de un modelo de procesador aritmético especializado para el cálculo numérico exacto con posibilidades de ajuste de la precisión. Se describen los métodos de expresión y procesamiento flexible así como una propuesta de implementación.

El procedimiento de representación de los números aprovecha la circunstancia de que la expresión fraccionaria posicional de todo número racional está formada por una cantidad finita de cifras significativas. A partir del esquema de representación en coma flotante, la incorporación de una mantisa que codifique la parte periódica junto con la flexibilidad en la longitud de los campos del formato da lugar a una función de representación de elementos de $\mathbb{Q}$. Los operadores sobre este sistema de representación numérica mantienen la capacidad operativa y añaden características de ajuste de la precisión atendiendo a los requerimientos de cada problema. El procesamiento de precisión variable que se aborda es el de las operaciones primitivas de identidad, suma y producto.

La operatoria se apoya en definiciones y teoremas sobre el cálculo de números racionales expresados en notación fraccionaria posicional. El desarrollo de los métodos se sustenta en dos pilares básicos: el diseño basado en esquemas iterativos, el cual permite que el procesamiento alcance a todas las cifras de los operandos y la utilización de memorias con resultados precalculados, lo que garantiza estructuras robustas y posibilidades de reutilización y paralelización. La integración de estos métodos de cálculo se materializa en la concepción de un procesador aritmético especializado que incorpora los elementos adicionales para la correcta gestión de la precisión de los operandos y de los resultados. Una unidad de control de la precisión determinará el grado de aproximación al resultado exacto para cada operación y una unidad flexible de memoria dispondrá la estructura necesaria de almacenamiento para alojar los datos de longitud variable.

Se realizan experimentos que muestran los aspectos relevantes tanto de los métodos como de los resultados. Se toma como base de comparación la capacidad de estos operadores para obtener un resultado exacto y, así, medir la desviación que produce el procesamiento con otros métodos de cálculo convencionales para valorar la oportunidad de su utilización en problemas críticos.

Resum

Aquest treball presenta la concepció i el desenvolupament d'un model de processador aritmètic especialitzat per al càlcul numèric exacte amb possibilitats d'ajustament de la precisió. Descriu els mètodes d'expressió i processament flexible, com també una proposta d'implementació.

El procediment de representació dels nombres aprofita la circumstància que l'expressió fraccionària posicional de tot nombre racional està formada per una quantitat finita de xifres significatives. A partir de l'esquema de representació en coma flotant, la incorporació d'una mantissa que codifique la part periòdica juntament amb la flexibilitat en la longitud dels camps del format dóna lloc a una funció de representació d'elements de $\mathbb{Q}$. Els operadors sobre aquest sistema de representació numèrica mantenen la capacitat operativa i afegeixen característiques d'ajustament de la precisió atenent als requeriments de cada problema. El processament de precisió variable que s'hi aborda és el de les operacions primitives d'identitat, suma i producte.

L'operatòria es fonamenta en definicions i teoremes sobre el càlcul de nombres racionals expressats en notació fraccionària posicional. El

desenvolupament dels mètodes està sustentat en dos pilars bàsics: el disseny basat en esquemes iteratius, que permet que el processament abrace totes les xifres dels operands i la utilització de memòries amb resultats precalculats, cosa que garanteix estructures robustes i possibilitats de reutilització i paral·lelisme. La integració d'aquests mètodes de càlcul es materialitza en la concepció d'un processador aritmètic especialitzat que incorpora els elements addicionals per a la correcta gestió de la precisió dels operands i dels resultats. Una unitat de control de la precisió determinarà el grau d'aproximació al resultat exacte per a cada operació, i una unitat flexible de memòria disposarà l'estructura necessària d'emmagatzematge per a allotjar les dades de longitud variable.

Es fan experiments que mostren els aspectes rellevants tant dels mètodes com dels resultats. Es pren com a base de comparació la capacitat d'aquests operadors per a obtenir un resultat exacte i, així, mesurar la desviació que produeix el processament amb altres mètodes de càlcul convencionals per a valorar l'oportunitat de la seua utilització en problemes crítics.

Abstract

This doctoral dissertation presents a model of specialized arithmetic processor from the scratch to precisely calculate numbers and which also allows further precision adjustment to be made on it. It describes not only expression but flexible processing methods together with an implementation proposal. It is hoped that its outcome will provide a deeper insight when building high performance processors.

The number representation format takes advantage of the fact that the positional fractional expression of any rational number is made up of a finite amount of significant digits. Taking the floating point representation as a starting point, a representation function of any $\mathbb{Q}$ element can be obtained through the addition of one mantissa that codifies the periodic part, given the flexibility the format shows in length. The operators on this numerical representation system maintain the operational capacity and incorporate characteristics of precision adjustment required by each problem. The variable precision processing discussed is the primitive identity, addition and multiplication operations.

The operating process is supported by definitions and theorems on the rational number calculation expressed in positional fractional notation. The development of the methods rests on two basic pillars: iterative design frames, which enable processing to account for all the operand digits and the use of memories containing pre-calculated results, ensuring robust structures parallel designs and further re-use. The integration of these calculation methods accomplishes a specialized arithmetic processor that incorporates the additional elements for the correct management of the precision of operands and results. A precision control unit will determine the degree of proximity the exact result for each operation and, a flexible memory unit will get adequate structure to lodge the data variable in length.

The experiments validate diverse aspects of the methods used and the results obtained. The capacity of these operators to deliver an exact result is taken as base for comparison to measure the deviation shown by conventional calculation method processing and to value its use in solving critical problems.