(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 546 915**

(21) Número de solicitud: 201430456

(51) Int. CI.:

G06F 7/38 (2006.01)

(12) SOLICITUD DE PATENTE A1

(22) Fecha de presentación:

28.03.2014

(43) Fecha de publicación de la solicitud:

29.09.2015

(71) Solicitantes:

UNIVERSIDAD DE MÁLAGA (100.0%)
Plaza de El Ejido, s/n
29071 Málaga ES

(72) Inventor/es:

HORMIGO AGUILAR, Francisco Javier y
VILLALBA MORENO, Julio

(74) Agente/Representante:

ZEA CHECA, Bernabé

(54) Título: **Unidades aritméticas en coma fija y conversores asociados**

(57) Resumen:

Dispositivos para realizar una operación deseada de al menos un primer número en coma fija pre-procesado con N+1 dígitos, para generar al menos un segundo número en coma fija pre-procesado con Z+1 dígitos son propuestos. Un formato en coma fija pre-procesado es un formato en coma fija en el que el LSD de todos los números representados exactamente en dicho formato es igual a B/2 (es decir, 1 para base binaria), y el resto son redondeados a uno de estos números. Los dispositivos comprenden al menos una unidad aritmética con una primera entrada para recibir los N MSDs de dicho al menos primer número en coma fija pre-procesado. La al menos una unidad aritmética está configurada para generar los Z MSDs del al menos segundo número en coma fija pre-procesado.

ES 2 546 915 A1

DESCRIPCIÓN

Unidades aritméticas en coma fija y conversores asociados.

La presente invención se refiere al procesamiento de datos y más concretamente a dispositivos para realizar operaciones de números en coma fija y los conversores asociados a los mismos.

ESTADO DE LA TÉCNICA

En los sistemas de procesado de información, la representación de los números se realiza mediante cadenas binarias. Los bits se pueden organizar en dígitos dependiendo del radix o base.

Los números pueden representarse en varios formatos. Los formatos más utilizados son el formato en coma flotante (FP) y el formato de coma fija (FF). En formato de coma fija, el cual incluye los números enteros, el número de dígitos fraccionarios y dígitos enteros es fijo. En esta representación, los números negativos se representan típicamente en formato de complemento, respecto de la base. Por ejemplo para números binarios se utiliza un formato de complemento a dos.

En coma flotante, el número se compone de la mantisa (Ma), la base (B) y el exponente (Ex). Por lo tanto, el valor (Va) representado sería Va = B * Ma ^ Ex. Entonces, solamente los números Ma y Ex necesitan almacenarse. El formato estándar IEEE-754 es el más extendido. El estándar define cinco formatos básicos que llevan el nombre de su base numérica y el número de bits usados en su codificación de intercambio. La precisión típica de los formatos binarios básicos es un bit más que la anchura de su mantisa (o mantisa). El bit de precisión extra proviene de un bit a uno implícito (oculto) en la parte más significativa. El número en coma flotante típico estará normalizado tal que el bit más significativo será un uno. Si conocemos que el bit más significativo es uno, entonces no se necesita codificarlo en el formato de intercambio.

Los sistemas para realizar operaciones entre estos números pueden usar una pluralidad de unidades funcionales. Estas unidades pueden realizar transformaciones

numéricas como operaciones aritméticas, conversiones de formato, evaluación de funciones, etc. El formato utilizado para representar los números con los que estos circuitos operan define completamente el diseño de estos circuitos y, por tanto, sus parámetros fundamentales de eficiencia tales como precisión, rango, velocidad, área y consumo. En consecuencia, el formato utilizado en estos sistemas influye enormemente en su eficiencia.

Dos circuitos básicos que se requieren en la mayoría de tales unidades funcionales son los circuitos de redondeo y los circuitos para complemento a dos.

Los circuitos de redondeo se utilizan cuando es necesario reducir el número de dígitos significativos, tanto en números en formato de coma fija como en la mantisa de números en formato de coma flotante. El circuito que realiza la función de complemento a dos se utiliza para cambiar el signo del número. Cualquier mejora en la eficiencia de estos dos circuitos afecta directamente a la eficiencia de la mayoría de las unidades funcionales que los incluyan.

Para realizar el complemento a la base de un número, primero se realiza el complemento a la base menos uno, una operación que se realiza sobre todos los dígitos en paralelo. Posteriormente se le suma al número una unidad-en-el-último lugar (ULP). En el caso binario, para que un circuito que lleva a cabo el complemento a dos de un número de N bits serían necesarios N inversores y un sumador de N bits. En el caso de una operación de resta (X-Y = X+(-Y)), que en realidad consiste en una suma con el complemento a dos del sustraendo, el bit de entrada de acarreo del sumador se suele utilizar para añadir el ULP. Sin embargo, esto no significa que cada vez que se requiere llevar a cabo el complemento a dos el motivo es una resta. Tales casos son la operación de valor absoluto o la suma/resta de números en representación signo-magnitud, una representación típicamente usada en coma flotante.

Con respecto a los circuitos de redondeo, se utilizan varias formas de redondeo. Una que demuestra importantes propiedades y es la más utilizada es el "redondeo al par más cercano". En este modo, el valor que se utiliza como valor final es el valor que está más cerca del valor real y, en caso de empate, el valor par. Usando este tipo de

redondeo, se obtiene un error inferior a +-0.5ULP y no presenta ningún sesgo en los errores.

Dado un número de D1 dígitos, para realizar una operación de redondeo a D2 dígitos, asumiendo D1 > D2, D1-D2 dígitos deben desecharse. Para que el redondeo sea al número más cercano, es importante examinar el valor del dígito más significativo de los que necesitan ser desechados (MD) y el dígito menos significativo de los que quedan (LD):

- Si MD < (B/2) entonces simplemente dichos dígitos son descartados.
- Si MD > (B/2) entonces dichos dígitos se descartan y se añade el valor uno al dígito menos significativo que permanece.
- Si MD = (B/2) entonces se debe verificar si alguno de los dígitos a descartarse no es cero (sticky bit). Si es así, entonces el redondeo se realiza según el segundo caso. Si todos son cero, entonces si el dígito LD es par entonces el redondeo se realiza según el primer caso y si es impar según el segundo caso.

Por lo tanto, el circuito básico para implementar este tipo de redondeo requiere un sumador para sumar uno si es necesario y un circuito para calcular el sticky bit.

Los circuitos de complemento a la base y redondeo son necesarios en las unidades funcionales tales como sumadores, multiplicadores, divisores, unidades FMAD, operadores de valor absoluto, conversores de formato o conversores de precisión etc. El coste adicional, por ejemplo en el área o retardo, que plantean dichos circuitos en las mencionadas unidades funcionales es generalmente substancial, sobre todo porque están típicamente en la vía crítica.

En el estado de la técnica anterior se han hecho varios intentos para reducir los efectos de estos cálculos, es decir el complemento a dos, el cálculo del sticky bit y redondeo. En ciertos documentos del estado de la técnica se ha propuesto precalcular el sticky bit o quitar estas operaciones de la vía crítica o reducir el número total de operaciones de redondeo necesarias o combinar redondeo y complemento a dos.

Sería deseable tener circuitos y métodos que reduzcan el coste en área, retardo y

consumo de los circuitos de redondeo al más cercano y/o de complemento a la base.

La presente invención se refiere a varios métodos y dispositivos para evitar o al menos reducir parcialmente este problema.

RESUMEN

La presente descripción se refiere a configuraciones y circuitos para operaciones en coma fija que implementan técnicas para codificar números con objeto de realizar funciones de redondeo al más cercano y complemento a la base sin la necesidad de realizar una suma. Por tanto, los sistemas que usen el tipo de codificación propuesto y que requieran estas operaciones podrían simultáneamente reducir área, retardo y consumo de potencia.

Con este fin, la presente descripción se centra en el diseño de sistemas digitales de procesamiento de información más eficientes (más rápidos, menor coste, menor consumo de energía) mediante el uso de una nueva familia de formatos o una modificación de los formatos de codificación numérica, aplicable a la mayoría de los formatos actuales, lo que implica cambios en los circuitos que procesan dichos formatos. Estos formatos simplifican drásticamente los circuitos para el redondeo al más cercano y complemento a la base, sin afectar negativamente al resto del circuito.

En un primer aspecto, se propone un dispositivo para realizar una operación deseada de al menos un primer número en coma fija pre-procesado con N+1 dígitos, para generar al menos un segundo número en coma fija pre-procesado con Z+1 dígitos. El dispositivo comprende al menos una unidad aritmética con una primera entrada para recibir los N MSDs de dicho al menos primer número en coma fija pre-procesado. La al menos una unidad aritmética está configurada para generar los Z MSDs del al menos segundo número en coma fija pre-procesado. El Digito Menos Significativo (LSD) de todos los números en coma fija pre-procesados es igual a B/2, siendo B la base del sistema numérico.

Una ventaja del dispositivo es la capacidad de realizar las operaciones mencionadas

sin usar explícitamente el LSD de los números en coma fija. Para lograr esto, los números en coma fija necesitan estar en un formato pre-procesado. El formato propuesto puede derivarse de cualquier formato no procesado, ya sea formato de coma fija o de coma flotante. En el caso de números en coma fija el formato pre-procesado puede obtenerse mediante la adición de un nuevo dígito como el dígito menos significativo (LSD). El valor de dicho dígito (KD) es igual a la base de representación dividida entre dos. En el caso de números de coma flotante, se lleva a cabo el mismo proceso para la mantisa del número FP.

Por lo tanto, en principio, los números pre-procesados necesitan un dígito más que los no procesados con la misma precisión. Sin embargo, como este dígito KD (o LSD) es una constante, no tiene que ser almacenado ni transmitido de forma explícita. Solamente puede ser requerido representar este dígito en una forma explícita, cuando existe la necesidad de realizar operaciones (aritmética, conversiones, o de otro tipo) con esos números. Por lo tanto, el almacenamiento y transmisión de números en formato pre-procesado (implícito) es equivalente al convencional.

Además, el número de valores representados en los dos formatos correspondientes (pre-procesado y no procesado) será el mismo. Sin embargo, los valores representados exactamente en cada formato, será diferente. Por ejemplo, en un formato binario de coma fija con sólo dos bits fraccionarios, cuatro valores son exactamente representables (0, 0.25, 0.5, 0.75), y en el formato pre-procesado correspondiente (es decir, tres bits fraccionarios), también cuatro valores son exactamente representables, pero unos diferentes (0.125, 0.375, 0.625, 0.875). Más específicamente, los valores exactamente representables en formato pre-procesado aparecerán exactamente en el punto intermedio entre la representación numérica exacta de los valores no procesados exactamente representables en el formato no procesado original. Esto significa que la precisión será equivalente en ambos formatos, pero la conversión entre ellos no puede ser exacta.

Un sistema digital que use el formato pre-procesado puede implementarse más eficientemente si el dígito KD está implícito. Dicho dígito KD puede añadirse a la entrada de un circuito de procesamiento, o introducirse cuando una operación requiere

su presencia. Por otro lado, si el número tiene que incluir explícitamente el dígito KD, por ejemplo para una operación posterior, entonces el dígito KD puede añadirse a la salida de una operación anterior.

Resumiendo, un formato en coma fija pre-procesado es un formato en coma fija en el que el LSD de todos los números representados exactamente en dicho formato es igual a B/2 (es decir, 1 para base binaria), y el resto son redondeados a uno de estos números. Por tanto, dicho LSB podría ser almacenado, transmitido o incluso operado, implícitamente. Un formato en coma flotante pre-procesado es un formato en coma flotante en el que la mantisa es un número en coma fija pre-procesado.

El uso números en formato pre-procesado simplifica enormemente la operación de redondeo “al más cercano” o “al par más cercano”. Esta es la principal ventaja del uso de este formato. Dado un número en coma fija, o la mantisa de un número en coma flotante, de D1-dígitos, la operación de redondeo “al más cercano” a un formato pre-procesado de D2+1 dígitos, siendo D1 y D2 números naturales tal que D1>D2, se realiza descartando los D1-D2 dígitos menos significativos (truncado). En el caso del redondeo “al par más cercano”, antes de operar es necesario comprobar si los D1-D2 dígitos menos significativos son todos cero (lo cual suele realizarse, calculando el sticky bit). Si es así, mientras se eliminan los D1-D2 dígitos menos significativos, se realizaría el siguiente proceso sobre el siguiente digito:

- Si el siguiente dígito es par, entonces se quedaría igual.
- Si el siguiente dígito es impar, entonces se le restaría uno a dicho dígito (lo que en ningún caso provocaría acarreo).

El uso de números en formato pre-procesado también simplifica la operación de complemento a la base. Debido al valor específico del LSD, la suma de 1 ULP después de complementar el número a la base menos uno, simplemente devuelve el valor del LSD a B/2, y no se produce acarreo hacia el resto de los dígitos. Por ejemplo, en formato binario, después de complementar a uno un número binario pre-procesado, el LSB es igual a cero y la suma de un ULP no produce ningún acarreo sino simplemente establece el LSB a uno de nuevo. Por lo tanto, la implementación del complemento de la base de un número pre-procesado sólo requiere complementar a la base menos uno

todos los dígitos menos el LSD, que permanece igual.

Las implementaciones según dicho aspecto tienen la ventaja de que no se necesita lógica para redondear por exceso (o hacia arriba). La eliminación de la lógica para redondear por exceso, que generalmente es un sumador independiente (o incrementador) o un sumador compuesto (sumador que devuelve X + Y y X + Y + 1) junto con otra lógica de control se hace posible porque el redondeo "al más cercano" para obtener un número pre-procesado se realiza, como se ha explicado antes, simplemente mediante truncado. Además, en muchos casos, no hay ninguna necesidad de tener una lógica para calcular el sticky bit. La eliminación de la lógica para el cálculo del sticky bit es posible porque, en muchos casos, el sticky bit es siempre uno, puesto que el último dígito oculto siempre es necesariamente B/2 (dígito KD). Por último, otra ventaja es que no puede ocurrir desbordamiento después del redondeo.

En las siguientes realizaciones de unidades aritméticas, se considera, en general, que los números en coma fija con signo, tanto los no procesado como los pre-procesados, son representados en representación complemento a dos, siendo el MSB equivalente al bit de signo. Sin embargo, un experto en la técnica podría apreciar que otros formatos que tienen una representación diferente, tal como números en signo-magnitud, podrían ser utilizados con modificaciones menores en los circuitos descritos. Además se considera que los números en coma fija, entradas y salidas del mismo módulo de suma o resta, están alineados por la izquierda. Esto significa que el MSB de estos números tiene el mismo peso en todos estos números y, por tanto, tienen el mismo número de bits enteros, aunque el número de bits fraccionarios podría ser distinto. Sin embargo, un experto en la técnica podría apreciar que los números que no estén alineados podrían ser configurados para estar alineados mediante extensión de ceros o extensión de signo, para números sin signo y con signo, respectivamente.

En algunas realizaciones, la al menos una unidad aritmética podría comprender además al menos una segunda entrada para recibir los L MSDs de un tercer número en coma fija pre-procesado con L+1 dígitos, y en el que L≥ N, y el LSD es igual a B/2. Alguien experto en la materia podría apreciar que si L<N, ambos números, es decir, el primer y tercer número, podrían ser intercambiados para cumplir dicha condición. Dicha

unidad aritmética podría comprender además un módulo de suma para generar un valor, correspondiente al segundo número en coma fija pre-procesado. Dicho segundo número en coma fija pre-procesado podría ser el resultado, redondeado al más cercano, de la suma del primer y el tercer número en coma fija pre-procesado. En implementaciones alternativas, dicho tercer número en coma fija pre-procesado podría ser una constante, y podría no recibirse explícitamente. En estas implementaciones el módulo de suma podría ser optimizado aún más, para realizar la suma de dicho número constante.

En algunas realizaciones, el módulo de suma podría comprender un sumador configurado para recibir los N MSBs del primer y tercer número en coma fija pre-procesado, en una primera y segunda entrada, respectivamente. En las siguientes realizaciones, el LSB del primer número en coma fija pre-procesado es considerado implícitamente para realizar la suma. En implementaciones alternativas, el sumador podría estar configurado para incorporar explícitamente el LSB de dicho número, es cual es siempre uno, aumentando en un bit el tamaño de sumador.

Cuando Z≤N, dicho sumador podría estar configurado para generar los Z MSBs del valor equivalente a sumar dichas dos entradas, más un acarreo de entrada. Dicho acarreo de entrada podría ser igual al (N+1)-ésimo bit del tercer número en coma fija pre-procesado, ya que el LSB del primer número en coma fija pre-procesado es uno. La principal ventaja de esta configuración es que no se requiere ningún circuito adicional para realizar el redondeo al más cercano del resultado, e incluso la generación de los N-Z LSBs no se requiere. Por tanto, alguien experto en la materia podría apreciar que una parte significativa de dicho sumador podría ser optimizado internamente, ya que solamente la última señal de acarreo correspondiente a la suma de los N-Z LSBs es requerida.

Por otro lado, cuando Z=N=L, el LSB del resultado exacto de la suma es cero, y por lo tanto, un redondeo por exceso se realiza siempre, lo que produce cierto sesgo. En este caso, el módulo de suma podría estar configurado además para fijar a cero el segundo LSB del segundo número en coma fija pre-procesado. Esta configuración adicional evita dicho sesgo. Además, el sumador podría ser simplificado ya que dicho segundo LSB podría no ser generado. En implementaciones alternativas, para evitar dicho

redondeo hacia arriba, la unidad aritmética o el dispositivo podrían configurarse para devolver el resultado exacto de la suma, el cual es un número no procesado (ya que el LSB es cero).

Cuando Z>N, dicho sumador podría estar configurado para generar los N MSBs del segundo número en coma fija pre-procesado produciendo un valor equivalente sumar dichas dos entradas más un acarreo de entrada. Dicho acarreo de entrada podría ser igual al (N+1)-ésimo bit del tercer número en coma fija pre-procesado, ya que el LSB del primer número en coma fija pre-procesado es uno. El módulo de suma podría estar configurado además para fijar el (N+1)-ésimo bit del segundo número en coma fija pre-procesado, igual al inverso del (N+1)-ésimo bit del tercer número en coma fija pre-procesado, que es equivalente a sumarle uno. Dicho módulo de suma podría estar configurado además para fijar los restantes Z-N-1 LSBs de los Z MSBs del segundo número en coma fija pre-procesado, igual a los Z-N-1 LSBs de los Z MSBs del tercer número en coma fija pre-procesado. El LSB del segundo número en coma fija pre-procesado es igual a uno y podría estar implícito. De nuevo no se requiere ningún circuito adicional para realizar el redondeo al más cercano del resultado.

En algunas realizaciones el módulo de suma podría estar configurado además para negar uno de los números de entrada. Como se ha declarado antes, dicha negación se realiza invirtiendo todos los bits excepto el LSB.

En algunas realizaciones dicha operación de negación se realiza selectivamente de acuerdo a una señal de control.

En otras implementaciones, el módulo de suma podría comprender más de dos entradas, para recibir más de dos números pre-procesados, respectivamente, para ser sumados. En este caso, el LSB de todos los números pre-procesados de entrada podría ser sumado al resultado de la suma de los restantes bits, como un valor constante, siendo éste el resultado de la suma del LSB de todos los números pre-procesados de entrada. Por ejemplo, si el módulo de suma está configurado para recibir NN operandos pre-procesados de entrada, todos con MM+1 bits, el resultado del módulo de suma podría ser obtenido sumando el valor NN (el cual es la suma del LSB de todas las entradas), correctamente alineado, al resultado de la suma de los MM MSBs de todos los números de entrada. Si los tamaños de los números de entrada no

son iguales, el peso de cada LSB tiene que ser tenido en cuenta para generar dicho valor constante. Por otro lado, si el valor constante es impar, entonces el resultado de la suma es un número pre-procesado. En otro caso, el segundo LSB del resultado podría ser fijado a cero para evitar el sesgo debido al redondeo.

Aunque el módulo de suma de las realizaciones propuestas aquí tiene el resultado de salida en un formato no redundante, alguien experto en la materia podría apreciar que la extensión de estas realizaciones a implementaciones con la salida en un formato redundante, tal como formato de acarreo almacenado o de dígitos con signo, podría realizarse de una forma directa.

En algunas realizaciones la al menos una unidad aritmética podría comprender un módulo de multiplicación, para generar un valor correspondiente al segundo número en coma fija pre-procesado.

En algunas realizaciones el módulo de multiplicación podría ser un elevador al cuadrado. Dicho de otra forma, el módulo de multiplicación podría estar configurado para generar dicho valor, correspondiente al segundo número en coma fija pre-procesado, el cual podría ser el resultado, redondeado al más cercano, de elevar al cuadrado el primer número en coma fija pre-procesado, teniendo el LSD igual a B/2.

Cuando el primer número en coma fija pre-procesado es con signo, el elevador al cuadrado podría comprender un módulo configurado para generar los N+1 MSBs de la magnitud (es decir, el valor sin signo) del primer número en coma fija pre-procesado. En este caso, un elevador al cuadrado para números sin signo podría ser usado para calcular la magnitud del segundo número en coma fija pre-procesado, mientras que el signo, el cual es siempre positivo, podría añadirse más tarde. En implementaciones alternativas, un elevador al cuadrado para números con signo podría ser usado, en lugar del calculador de la magnitud y del elevador al cuadrado para números sin signo. En otras implementaciones, el primer enfoque podría ser usado para diseñar un elevador al cuadrado combinado para números con y sin signo.

En algunas realizaciones el módulo de multiplicación podría estar configurado para generar dicho valor, correspondiente al segundo número en coma fija pre-procesado, el

cual podría ser el resultado, redondeado al más cercano, de la multiplicación del primer y un cuarto número en coma fija pre-procesado de T+1 dígitos, teniendo el LSD igual a B/2.

Cuando el cuarto número en coma fija pre-procesado es un número constante, el módulo de multiplicación podría ser un multiplicador por constante. En este caso, dicho número constante podría no ser recibido explícitamente. Alguien experto en la materia podría apreciar que cualquier técnica de optimización para implementar multiplicadores por constante podría ser aplicada a la invención revelada, de una forma directa.

En algunas realizaciones la al menos una unidad aritmética podría comprender además al menos una segunda entrada para recibir los T MSDs del cuarto número en coma fija pre-procesado.

En algunas realizaciones el módulo de multiplicación podría comprender un multiplicador. El multiplicador podría estar configurado para generar los N+T+1 MSBs del resultado de la multiplicación, ya que el LSB de dicho resultado es siempre uno, para números pre-procesados de entrada. Si el módulo de multiplicación es un elevador al cuadrado, solamente se requiere la generación de los 2*N MSBs, ya que, también el segundo LSB es siempre cero. El módulo de multiplicación podría comprender además un módulo de truncado, conectado a la salida del multiplicador, para recibir la salida del multiplicador, y generar los Z MSBs del segundo número, truncando dicha salida. El LSB del segundo número en coma fija pre-procesado está implícito y es igual a uno. De nuevo, no se requiere ningún circuito adicional para realizar el redondeo al más cercano del resultado, tal como un sumador para redondear hacia arriba, o un calculador del sticky bit.

Como los N+T-Z+2 LSBs del resultado exacto de la multiplicación no se requieren para obtener el segundo número en coma fija pre-procesado correctamente redondeado, el módulo de multiplicación podría optimizarse evitando la generación explicita de dichos N+T-Z+2 LSBs. Por lo tanto, en algunas realizaciones el módulo de multiplicación podría comprender un módulo multiplicación redundante configurado para recibir, en una primera entrada, los N MSBs del primer número en coma fija pre-procesado y generar, en un formato de representación redundante, como mucho los N+T+1 MSDs

del valor correspondiente a la operación de multiplicación entre dicho primer número en coma fija pre-procesado y el cuarto número en coma fija pre-procesado. El LSD del resultado de dicha multiplicación es implícito e igual a uno. El módulo de multiplicación podría comprender además un módulo de conversión, conectado a la salida de dicho módulo de multiplicación, configurado para recibir los Z MSDs de la salida de dicho multiplicador redundante, y un bit de acarreo, y generar una salida de Z bits correspondiente a la conversión del valor redundante recibido a formato de representación no redundante. El módulo de multiplicación podría comprender además un módulo de red de acarreo configurado para recibir los N+T+1-Z LSDs de la salida de dicho módulo multiplicador redundante, y generar dicho bit de acarreo correspondiente al acarreo de salida de la conversión de los N+T+1-Z LSDs de la salida de dicho módulo de multiplicación redundante a representación no redundante.

Alguien experto en la materia podría apreciar que la longitud de palabra de los valores intermedios en las realizaciones divulgadas aquí, garantiza el menor error de redondeo. Sin embargo, si un mayor error es permisible, esos tamaños podrían reducirse para simplificar el hardware de una forma directa. Por ejemplo, el tamaño de la salida del multiplicador redundante podría ser menor de N+T+1 dígitos, tal que la entrada el módulo de conversión podría mantenerse igual, mientras que la entrada del módulo de red de acarreo podría ser reducido en consonancia.

Alguien experto en la materia podría apreciar que, además de la propuesta descrita arriba, diferentes técnicas de optimización, las cuales podrían aprovecharse del hecho de que los N+T-Z+2 LSBs no se requieren explícitamente, tal como multiplicadores truncados, podrían ser aplicados a la invención revelada, de una forma directa.

En algunas realizaciones el módulo de multiplicación redundante podría comprender un generador de productos parciales configurado para recibir, en una primera entrada, los N MSBs del primer número en coma fija pre-procesado y generar, en una salida, los productos parciales correspondientes a la multiplicación de dicha entrada y los T MSBs del cuarto número en coma fija pre-procesado. Si dicho cuarto número en coma fija pre-procesado es una constante, dicho generador de productos parciales podría estar optimizado para generar un conjunto reducido de productos parciales correspondiente

al producto de dicha primera entrada por dicho número constante sin recibir este último explícitamente. Si éste no es una constante, dicho generador de productos parciales podría estar configurado para recibir dichos T MSBs. El módulo de multiplicación redundante podría comprender además un árbol de compresores, con una primera entrada conectada a la salida del generador de productos parciales y una segunda entrada configurada para recibir los N MSBs, y los T MSBs, del primer, y cuarto, número pre-procesado, respectivamente. En implementaciones alternativas, cuando el cuarto número en coma fija pre-procesado es una constante, dichos T MSBs podrían ser tenidos en cuenta internamente al árbol de compresores, para generar un circuito más optimizado. Dicho árbol de compresores podría estar configurado para generar, en una representación redundante, como mucho los N+T+1 MSDs de un valor correspondiente a la operación de multiplicación entre dichos números pre-procesados en una salida. Como el LSB de los números pre-procesados es igual a uno, el generador de productos parciales no requiere generar productos parciales para los dichos LSBs y podría considerarse que ya están generados. Ellos pueden ser introducidos directamente en el árbol de compresores (externamente o internamente) lo que resulta en menos operaciones y lógica para el generador de productos parciales. En una implementación alternativa, dichos LSBs podrían ser considerados dentro del propio generador de productos parciales, y dichos valores podrían no ser introducidos en dicha segunda entrada del árbol de compresores.

En algunas realizaciones, la unidad aritmética podría comprender un módulo de desplazamiento a la izquierda para generar un valor, correspondiente al segundo número en coma fija pre-procesado. Dicho segundo número en coma fija pre-procesado podría ser el resultado, redondeado al más cercano, del desplazamiento a la izquierda del primer número en coma fija pre-procesado. Aunque la operación de desplazamiento a la izquierda (es decir, la multiplicación por una potencia de la base), para números no procesados, es una operación exacta, es decir, el resultado no necesita ningún redondeo, esto no es cierto para formatos en coma fija pre-procesados. El resultado exacto de desplazar a la izquierda un número en coma fija pre-procesado no es un número pre-procesado, ya que su LSD no es igual a B/2. Por lo tanto se requiere una operación de redondeo, la cual, en principio, no implica ninguna operación adicional. Sin embargo, este redondeo podría producir cierto sesgo

introducido por el hecho de que siempre se realiza un redondeo por exceso. En implementaciones alternativas, para evitar dicho redondeo por exceso, la unidad aritmética, o el dispositivo, podrían configurarse para devolver el resultado exacto del desplazamiento, el cual es un número no procesado.

En algunas realizaciones, el módulo de desplazamiento a la izquierda podría estar configurado además para completar las posiciones vacantes, debidas al desplazamiento a la izquierda, fijando el MSB de la posiciones vacantes a cero y el resto a uno, o fijando el MSB de las posiciones vacantes uno y el resto a cero. Esta configuración produce un redondeo por defecto para el primero, y un redondeo por exceso para el segundo.

En algunas realizaciones el módulo de desplazamiento a la izquierda podría estar configurado para, selectivamente, completar dichas posiciones vacantes, aleatoriamente, basándose en el valor de un bit seleccionado, o de una combinación de bits seleccionados. Esta configuración permite evitar el sesgo en el redondeo. En algunas implementaciones, dicho bit (o bits) seleccionado podría pertenecer al número de entrada, mientras que en otras una nueva entrada podría configurarse.

En algunas realizaciones, el módulo de desplazamiento a la izquierda podría estar configurado además para recibir la cantidad de desplazamiento para seleccionar el número de bits a desplazar.

En algunas realizaciones, el módulo de desplazamiento a la izquierda podría comprender un desplazador variable configurado para recibir un bit para completar las posiciones vacantes.

En algunas realizaciones, dicho desplazador variable podría comprender un número de sucesivos multiplexores que podría ser igual al primer entero mayor o igual que el logaritmo en base 2 de la máxima cantidad de desplazamiento [log2(máxima cantidad de desplazamiento)], con cada multiplexor configurado para efectuar una operación de desplazamiento a la izquierda de 2^i posiciones, iϵ[0, número de multiplexores-1], y cada multiplexor configurado para completar las posiciones vacantes usando el valor de dicho bit recibido.

En algunas realizaciones, al menos una unidad aritmética podría comprender un módulo de valor absoluto, para generar un valor correspondiente al segundo número en coma fija pre-procesado. Dicho segundo número en coma fija pre-procesado podría ser el resultado del valor absoluto del primer número en coma fija pre-procesado. Esta operación implica la negación del número de entrada, si éste es negativo. Como el número de entrada es un número pre-procesado, esta negación podría ser implementada simplemente invirtiendo todos los bits menos el LSB, y no se requiere ninguna suma. Por tanto, el módulo de valor absoluto podría comprender un inversor de bit condicional configurado para recibir, en una primera entrada, los N MSBs del primer número en coma fija pre-procesado. Dicho inversor de bit condicional podría generar un valor correspondiente al complemento a uno de la primera entrada, si su MSB es igual a uno.

En algunas implementaciones al menos una unidad aritmética podría comprender un módulo calculador de funciones elementales, para generar un valor correspondiente al segundo número en coma fija pre-procesado. Dicho segundo número en coma fija pre-procesado podría ser el resultado, redondeado al más cercano, de aplicar una función elemental al primer número en coma fija pre-procesado. Dicha función elemental podría ser cualquier función matemática de una variable, tal como funciones trigonométricas, logaritmo, exponencial, etc. Pero alguien experto en la materia podría apreciar que una extensión a funciones multivariables es directa. El módulo calculador de funciones elementales podría comprender una tabla de búsqueda, configurada para recibir, en una primera entrada, los N MSDs del primer número en coma fija pre-procesado. Dicha tabla de búsqueda podría estar configurada además para almacenar y devolver los Z MSDs de dicho segundo número en coma fija pre-procesado correspondiente a cada posible entrada. El LSD de dicho segundo número en coma fija pre-procesado es igual a B/2 y podría estar implícito. Una ventaja de esta propuesta es que el LSD del número de salida no necesita almacenarse o devolverse explícitamente. Otra ventaja es que el valor almacenado en la tabla de búsqueda es redondeado exactamente a cualquier precisión por debajo de Z+1 dígitos, simplemente mediante truncado.

En algunas realizaciones, el dispositivo podría comprender además un conversor de números no procesados a números pre-procesados en coma fija, conectado a una entrada de la unidad aritmética, y configurado para recibir un número en coma fija no

procesado de E+1 bits, y generar un número en coma fija pre-procesado. Introduciendo tal conversor delante de las unidades aritméticas, de acuerdo a las realizaciones reveladas aquí, permite que un número en un formato coma fija no procesado sea operado por dichas unidades aritméticas que funcionan en formato en coma fija pre-procesado.

Cuando el número en coma fija pre-procesado tiene E+1-K1 bits, con K1<E entonces el conversor podría comprender una unidad de redondeo configurada para eliminar los K1+1 LSBs del número en coma fija no procesado, para generar los E-K1 MSBs del número coma fija pre-procesado. El LSB de dicho número en coma fija pre-procesado es igual a B/2 y está implícito.

En algunas realizaciones la unidad de redondeo podría estar configurada además para, selectivamente, poner a cero el segundo LSB del número en coma fija pre-procesado si todos los K1+1 LSBs del número en coma fija no procesado son iguales a cero. Esta configuración permite evitar el sesgo en el redondeo.

Cuando el número en coma fija pre-procesado tiene E+1+K2 bits entonces el conversor podría comprender un módulo de rellenado, configurado para recibir el número en coma fija no procesado y generar los E+K2 MSBs del número en coma fija pre-procesado fijando los E+1 MSBs del número en coma fija pre-procesado al mismo valor que los E+1 bits del número en coma fija no procesado y los restantes bits a cero. El LSB del número en coma fija pre-procesado es igual a uno y está implícito.

En algunas realizaciones el módulo de rellenado podría estar configurado además para generar selectivamente el valor correspondiente a restar uno del segundo LSB del mencionado número en coma fija pre-procesado cuando un bit seleccionado, o una combinación de bit seleccionados, del número no procesado de entrada es igual a uno. Esta configuración permite evitar el sesgo en el redondeo.

En algunas realizaciones el dispositivo podría comprender además un conversor de números coma fija pre-procesados a números coma fija pre-procesados, conectado a una entrada y/o una salida de la unidad aritmética, y configurado para recibir un

número inicial coma fija pre-procesado de J+1 bits, y generar un subsecuente número coma fija pre-procesado de diferente tamaño. Este podría ser útil a la entrada, por ejemplo, cuando un operando es proporcionado a la unidad aritmética con más precisión (o con menos precisión) de lo necesario. De la misma forma, si el resultado de la operación necesita ser convertido a un número coma fija de diferente tamaño, de forma que éste pueda ser utilizado por un circuito posterior, dicho conversor puede ser utilizado a la salida. Por lo tanto, el conversor podría colocarse antes o después de la unidad aritmética, de acuerdo con esto.

Cuando el subsecuente número en coma fija pre-procesado tiene J+1-P1 bits, P1<J, entonces el conversor podría comprender una unidad de redondeo para eliminar los P1+1 LSBs de los J+1 bits del número inicial pre-procesado, para generar los J-P1 MSBs del subsecuente número en coma fija pre-procesado. El LSB del subsecuente número coma fija pre-procesado es igual a B/2 y está implícito.

Cuando el subsecuente número en coma fija pre-procesado tiene J+1+P2 bits, entonces el conversor podría comprender un módulo de rellenado, configurado para recibir los J MSBs del número en coma fija pre-procesado inicial, y generar los J+P2 MSBs del subsecuente número en coma fija pre-procesado, fijando el MSB de los P2 LSBs a uno, o a cero, y los restante P2-1 bits de dichos P2 LSBs, al inverso del mencionado MSB. Dependiendo del valor de dicho MSB, un redondeo efectivo por exceso, o por defecto, es producido. Los J MSBs del subsecuente número en coma fija pre-procesado podrían ser los mismos que los J MSBs del número en coma fija pre-procesado inicial. El LSB del subsecuente número coma fija pre-procesado es igual a B/2 y está implícito.

En algunas realizaciones el módulo de rellenado podría estar configurado además para fijar aleatoriamente dicho MSB, basándose en el valor de un bit seleccionado, o de una combinación de bits seleccionados. En algunas implementaciones dicho bit (o bits) podrían seleccionarse del número en coma fija pre-procesado inicial.

En algunas realizaciones el dispositivo podría comprender además un conversor de números coma fija pre-procesados a números coma fija no procesados, conectado a la salida de una unidad aritmética y configurado para, recibir un número en coma fija pre-

procesado de W+1 bits y generar un número en coma fija no procesado. Introduciendo tal conversor detras de las unidades aritméticas, de acuerdo a las realizaciones reveladas aquí, permite que un número en coma fija pre-procesado generado por dicha unidad aritmética, sea operado por circuitos en coma fija comunes.

Cuando el número en coma fija no procesado tiene W+1-V1 bits, V1<W, entonces el conversor podría comprender un módulo de redondeo, configurado para recibir los W+2-V1 MSBs del número en coma fija pre-procesado y generar los W+1-V1 bits del número en coma fija no procesado.

En algunas realizaciones el módulo de redondeo podría comprender un sumador. Dicho sumador podría estar configurado para recibir, en una entrada, los W+1-V1 MSBs del número en coma fija pre-procesado e incrementar dicho valor de entrada, si el (W+2-V1)-ésimo MSB de dicho número pre-procesado es igual a 1. La computación del sticky bit no es requerida ya que la entrada es un número pre-procesado y su LSB es igual a uno.

Cuando el número en coma fija no procesado tiene W+1+V2 bits, entonces el conversor podría comprender un módulo de rellenado, configurado para recibir los W MSBs del número en coma fija pre-procesado y generar los W+V2+1 bits del número en coma fija no procesado poniendo el MSB de los V2+1 LSBs a uno, y los restantes bits a cero.

En los siguientes realizaciones de conversores, se considera que los números en coma flotante, tanto los no procesados como los pre-procesados, son representados por un bit de signo, un exponente y una mantisa normalizada sin signo, de tal forma que el MSB es igual a uno y está explícitamente incluido en la representación de la mantisa. Sin embargo, un experto en la técnica podría apreciar que otros formatos que tienen una representación diferente podrían ser utilizados con modificaciones menores en los circuitos descritos.

En algunas realizaciones, el dispositivo podría comprender además un conversor de números coma flotante pre-procesados a números coma fija pre-procesados,

conectado a una entrada de una unidad aritmética, y configurado para recibir un número en coma flotante con una mantisa de F+2 bits, y para generar un número en coma fija pre-procesado. Introduciendo un conversor de este tipo antes de una unidad aritmética, de acuerdo a las realizaciones descritas aquí, permite que un número en formato coma flotante pre-procesado sea operado por dichas unidades aritméticas funcionando con formato coma fija pre-procesado.

Cuando el número en coma fija pre-procesado comprende G bits, con G<F+4, el conversor de números coma flotante pre-procesados a números coma fija pre-procesados podría comprender un calculador de la cantidad de desplazamiento, que recibe el exponente del número en coma flotante pre-procesado, en una entrada, y genera una cantidad de desplazamiento, en una salida. El conversor podría comprender además un módulo de desplazamiento, con una primera entrada para recibir los G-1 MSBs de la mantisa del número en coma flotante pre-procesado, una segunda entrada, conectada a la salida del calculador de cantidad de desplazamiento, y una tercera entrada, para recibir el signo del mencionado número en coma flotante, para generar los G-1 MSBs del número en coma fija pre-procesado, en una salida. El LSB de dicho número en coma fija pre-procesado es igual a B/2 y podría estar implícito.

En algunas realizaciones, el módulo de desplazamiento del conversor de números coma flotante pre-procesados a números coma fija pre-procesados podría comprender un desplazador aritmético a la derecha conectado a un inversor de bits condicional. En algunas implementaciones el inversor precede al desplazador, mientras que en otras podría ser al contrario.

Cuando el número en coma fija pre-procesado comprende F+C+3 bits, C>0, el conversor de números coma flotante pre-procesados a números coma fija pre-procesados podría comprender un calculador de cantidad de desplazamiento, que recibe el exponente del número pre-procesado en una entrada, y que genera una cantidad de desplazamiento en una salida, y un módulo de desplazamiento aritmético a la derecha, con una primera entrada conectada a la salida del calculador de desplazamiento, y configurado para generar los F+C+2 MSBs del número en coma fija pre-procesado, mediante el desplazamiento aritmético a la derecha de un valor

intermedio de F+C+2 bits. Dicho valor intermedio podría estar formado, de izquierda a derecha, por el bit de signo, los F+1 MSBs de la mantisa del número en coma flotante pre-procesado, y el MSB de los C LSBs puesto a cero, y el resto a uno, o el MSB de los C LSBs puesto a uno, y el resto a cero.

En algunas realizaciones, el módulo de desplazamiento aritmético a la derecha podría estar configurado para poner aleatoriamente dicho MSB de los C LSBs del mencionado valor de F+C+2 bits, en base al valor de un bit seleccionado, o de una combinación de bits seleccionados. En algunas implementaciones dicho bit (o bits) podrían seleccionarse del número en coma flotante pre-procesado.

En algunas realizaciones, el módulo de desplazamiento aritmético a la derecha podría estar configurado además para generar selectivamente el complemento a uno del resultado de la mencionada operación de desplazamiento.

En algunas realizaciones, el dispositivo podría comprender además un conversor de números coma fija pre-procesados a números coma flotante pre-procesados, conectado a una salida de una unidad aritmética, y configurado para convertir un número coma fija de Q+2 bits a un número coma flotante con una mantisa de M+2 bits. El conversor de números coma fija pre-procesados a números coma flotante pre-procesados podría comprender un calculador de cantidad de desplazamiento, un módulo para calcular el exponente, con una primera entrada para recibir la cantidad de desplazamiento del calculador de cantidad de desplazamiento, y una salida para generar el exponente del número coma flotante pre-procesado, y un calculador de la mantisa. El calculador de la mantisa podría comprender un módulo de normalización, con una primera entrada para recibir los Q MSBs de los Q+1 LSBs del número en coma fija, y una segunda para recibir la tercera cantidad de desplazamiento. El módulo de normalización podría estar configurado para desplazar a la izquierda dichos Q MSBs de acuerdo con dicha cantidad de desplazamiento, completando el MSB de las posiciones vacantes con cero y el resto con unos, o el MSB con uno y el resto con ceros, para generar como mucho los M+1 MSBs de la mantisa. El signo del número coma flotante pre-procesado podría corresponder con el MSB del número coma fija pre-procesado. Introduciendo un conversor de este tipo después de una unidad aritmética de acuerdo a

las realizaciones descritas aquí permite que un número en formato de coma fija pre-procesado generado por ésta, sea procesado por dispositivos pre-procesados en coma flotante.

En algunas realizaciones, el módulo de normalización del calculador de la mantisa podría estar configurado para completar dichas posiciones vacantes, aleatoriamente, basándose en un bit seleccionado, o en una combinación de bits seleccionados. En algunas implementaciones dicho bit (o bits) podrían seleccionarse del número coma fija pre-procesado. En otras implementaciones, una nueva entrada podría configurarse.

En algunas realizaciones, el módulo de normalización del calculador de la mantisa podría estar configurado además para generar selectivamente el complemento a uno del resultado de dicho desplazamiento.

En algunas realizaciones, el dispositivo podría comprender además un conversor de números coma fija pre-procesados a números coma flotante no procesados, conectado a una salida de una unidad aritmética, y configurado para para convertir un número coma fija de H+2 bits a un número coma flotante con una mantisa de R+1 bits.

En algunas realizaciones, dicho conversor de números coma fija pre-procesados a números coma flotante no procesados podría comprender un calculador de cantidad de desplazamiento, un módulo para calcular el exponente y un módulo calculador de mantisa. Dicho módulo para calcular el exponente podría tener una primera entrada, para recibir la cantidad de desplazamiento del calculador de cantidad de desplazamiento, y una salida, para generar el exponente del número coma flotante no procesado. El módulo calculador de mantisa podría comprender un módulo de normalización, con una primera entrada para recibir los H MSBs de los H+1 LSBs del número coma fija pre-procesado, y una segunda, para recibir la cantidad de desplazamiento. Dicho módulo de normalización podría estar configurado para generar un valor correspondiente a como mucho los R+2 MSBs de los H+1 LSBs del número en coma fija pre-procesado desplazado a la izquierda, de acuerdo con dicha cantidad de desplazamiento. Dicho módulo calculador de mantisa podría comprender además un módulo de redondeo configurado para recibir la salida del módulo de normalización y

generar como mucho los R+1 bits de la mantisa del número coma flotante no procesado. El signo del número coma flotante no procesado podría corresponder al MSB del número coma fija pre-procesado.

En algunas realizaciones, dicho módulo de normalización podría estar configurado además para generar selectivamente la negación de dicho valor de como mucho R+2 bits.

En algunas realizaciones, el módulo de redondeo podría comprender un sumador. Dicho sumador podría estar configurado para recibir, en una entrada, los como mucho R+1 MSBs de la salida del módulo de normalización e incrementar dicha valor de entrada, si el LSB de dicha salida es igual a 1.

En algunas realizaciones, el dispositivo podría comprender además un conversor de números coma flotante no procesados a números coma fija pre-procesados, conectado a una entrada de una unidad aritmética, y configurado para convertir un número en coma flotante no procesado con una mantisa de S bits en un número en coma fija pre-procesado de A+2 bits. Introduciendo tal conversor delante de las unidades aritméticas, de acuerdo a las realizaciones reveladas aquí, permite que un número en formato coma fija no procesado sea operado por dichas unidades aritméticas funcionando en formato coma fija pre-procesado.

En algunas realizaciones, dicho conversor de números en coma flotante no procesados a números en coma fija pre-procesados podría comprender un calculador de la cantidad de desplazamiento, que recibe el exponente del número en coma flotante no procesado, en una entrada, y que genera una cantidad de desplazamiento en una salida, un conversor de números en coma fija no procesados a números en coma fija pre-procesados, de acuerdo a las realizaciones descritas aquí, y un módulo de desplazamiento. Dicho conversor de números en coma fija no procesados a números en coma fija pre-procesados podría estar configurado para recibir como mucho los S bits de la mantisa del número en coma flotante no procesado y generar los A MSBs de

un número en coma fija pre-procesado. El módulo de desplazamiento podría tener una primera entrada, para recibir la salida de A bits de dicho conversor, una segunda entrada, conectada a la salida del calculador de cantidad de desplazamiento, y una tercera entrada, para recibir el signo del mencionado número en coma flotante. Dicho módulo de desplazamiento podría estar configurado para generar los A+1 MSBs del número en coma fija pre-procesado de salida, desplazando a la derecha, de acuerdo a la segunda entrada, la primera entrada aumentada por la izquierda con el bit de signo. El LSB de dicho número en coma fija pre-procesado es igual a B/2 y podría estar implícito. En algunas implementaciones, el MSB de la mantisa del número en coma flotante podría estar implícito, ya que siempre es igual a uno, y podría no ser recibido explícitamente por el conversor.

En algunas realizaciones, dicho módulo de desplazamiento podría estar configurado además para generar selectivamente un valor igual al complemento a uno del resultado de dicho desplazamiento.

En algunas realizaciones, el módulo de desplazamiento podría comprender un desplazador aritmético a la derecha acoplado a un inversor de bits condicional. En algunas implementaciones el inversor precede al desplazador, mientras que en otras podría ser al contrario.

En algunas realizaciones, el dispositivo podría comprender además una tercera entrada y/o salida para recibir y/o devolver el LSD de dichos primero y/o segundo número en coma fija pre-procesado. Alternativamente, dicha tercera entrada y/o salida podrían tener un valor de B/2, ya que el LSD de los números en coma fija pre-procesados es igual a B/2. Por lo tanto, el número pre-procesado completo podría ser usado en las operaciones siguientes, aunque no sería necesario transmitir el número completo hasta la entrada del dispositivo y/o la salida.

En algunas realizaciones, el dispositivo podría comprender una pluralidad de unidades aritméticas y una entrada de selección de operación, para recibir una señal sobre la operación deseada. Dicho dispositivo podría estar configurado para seleccionar la salida de una unidad aritmética de la pluralidad de unidades aritméticas, basándose en

dicha señal sobre la operación deseada recibida.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

A continuación se describirán realizaciones particulares de la presente invención por medio de ejemplos no limitativos, con referencia a los dibujos adjuntos, en los que:

Fig. 1a y 1b ilustran ejemplos de implementación de un módulo de suma en coma fija para números pre-procesados de distinto tamaño;

Fig. 2a, 2b y 2c ilustran ejemplos de implementación de un módulo de suma en coma fija para números pre-procesados del mismo tamaño;

Fig. 3 ilustra la implementación de un circuito restador en coma fija para números pre-procesados de acuerdo a un ejemplo;

Fig. 4 ilustra la implementación de un circuito sumador/restador en coma fija para números pre-procesados de acuerdo a un ejemplo;

Fig. 5a y 5b ilustran ejemplos de implementación de un módulo de multiplicación en coma fija para números pre-procesados;

Fig. 6a y 6b ilustran ejemplos de implementación de un multiplicador redundante para números pre-procesados;

Fig. 7a, 7b y 7c ilustran ejemplos de implementación de un módulo de elevar al cuadrado en coma fija para números pre-procesados;

Fig. 8 ilustra la implementación de un módulo de elevar al cuadrado redundante para números pre-procesados de acuerdo a un ejemplo;

Fig. 9 ilustra un ejemplo de implementación de un módulo de elevar al cuadrado para números con signo pre-procesados;

Fig. 10a, 10b y 10c ilustran ejemplos de implementación de un módulo de multiplicación por constante en coma fija para números pre-procesados;

Fig. 11 ilustra la implementación de un módulo de multiplicación por constante redundante para números pre-procesados de acuerdo a un ejemplo;

Fig. 12 ilustra un ejemplo de implementación de un desplazador a la izquierda para números pre-procesados;

Fig. 12a ilustra un ejemplo de implementación de un desplazador a la izquierda especial;

Fig. 13a, 13b y 13c ilustran ejemplos de implementación de conversores para convertir números en coma fija no procesados a números en coma fija pre-procesados;

Fig. 14a, 14b y 14c ilustran ejemplos de implementación de conversores para convertir números en coma fija pre-procesados a números en coma fija pre-procesados;

Fig. 15 ilustra un ejemplo de implementación de un conversor para convertir números en coma fija pre-procesados a números en coma fija no procesados;

Fig. 15a ilustra un ejemplo de implementación de un conversor para convertir números en coma fija pre-procesados a números en coma fija no procesados mediante redondeo al más cercano;

Fig. 16a, 16b y 16c ilustran ejemplos de implementación de conversores para convertir números en coma flotante pre-procesados a números en coma fija pre-procesados;

Fig. 17 ilustra un ejemplo de implementación de un conversor para convertir números en coma fija pre-procesados a números en coma flotante pre-procesados;

Fig. 18 ilustra un ejemplo de implementación de un conversor para convertir números en coma fija pre-procesados a números en coma flotante no procesados;

Fig. 19 ilustra un ejemplo de implementación de un conversor para convertir números en coma flotante no procesados a números en coma fija pre-procesados;

DESCRIPCION DETALLADA DE LAS REALIZACIONES

En los siguientes ejemplos de unidades aritméticas, se considera, en general, que los números en coma fija con signo, tanto los no procesado como los pre-procesados, son representados en representación complemento a dos, siendo el MSB equivalente al bit de signo. Sin embargo, un experto en la técnica podría apreciar que otros formatos que tienen una representación diferente, tal como números en signo/magnitud, podrían ser utilizados con modificaciones menores en los circuitos descritos. Además se considera que los números en coma fija, entradas y salidas del mismo módulo de suma o resta, están alineados por la izquierda. Esto significa que el MSB de estos números tiene el mismo peso en todos estos números y, por tanto, tienen el mismo número de bits enteros, aunque el número de bits fraccionarios podría ser distinto. Sin embargo, un experto en la técnica podría apreciar que los números que no estén alineados podrían ser configurados para estar alineados mediante extensión de ceros o extensión de signo, para números sin signo y con signo, respectivamente.

Fig. 1a y 1b ilustran las implementaciones de un módulo de suma en coma fija de acuerdo a diferentes ejemplos. Un módulo de suma en coma fija 300SFJ, o 400SFJ, recibe los N MSBs de un primer número en coma fija pre-procesado de N+1 bits, y los N+M+1 MSBs de un segundo número en coma fija pre-procesado de N+M+2 bits, en una primera y una segunda entrada, respectivamente, siendo M≥0. El módulo de suma en coma fija 300SFJ, o 400SFJ, genera los Z MSBs, de un tercer número en coma fija pre-procesado de Z+1 bits, correspondientes a la suma de ambos números de entrada. El LSB de los números en coma fija pre-procesados es igual a uno y no necesita introducirse, o generarse, explícitamente en el módulo de suma. El módulo de suma en coma fija 300SFJ, o 400SFJ, comprende un sumador de N bits 320SFJ, o 420SFJ, teniendo la primera y segunda entrada de N bits conectada a los N MSBs del primer y segundo número en coma fija pre-procesado, respectivamente, y el acarreo de entrada conectado al (N+1)-ésimo MSB de dicho segundo número en coma fija pre-procesado.

El sumador 320SFJ, o 420SFJ, genera los N MSBs del tercer número en coma fija pre-procesado. Fig. 1b muestra el caso extremo en el que Z=N. En el caso de que Z>N, el (N+1)-ésimo MSB del tercer número en coma fija pre-procesado es fijado al inverso del (N+1)-ésimo MSB del segundo número en coma fija pre-procesado, mientras que los Z-N-1 LSBs, de dicho tercer número en coma fija pre-procesado, son fijados igual a los Z-N-1 LSBs de dicho segundo número en coma fija pre-procesado. Fig. 1a muestra el caso extremo en el cual Z=N+M+1. Por otro lado, si Z<N, el sumador de N bits 320SFJ, de Fig. 1a, podría ser sustituido por un sumador de Z bits, para sumar los Z MSBs del primer y segundo número de entrada pre-procesado, y un módulo de red de acarreo, para generar el acarreo de entrada de dicho sumador de Z bits, teniendo en cuenta los N+1-Z LSBs de los N+1 MSBs de dicho primer y segundo número de entrada. El LSB del tercer número pre-procesado es igual a uno, no necesita generarse y está implícito en estos ejemplos.

Fig. 2a y 2b ilustran un módulo de suma en coma fija de acuerdo a otros ejemplos, en los cuales los números de entrada tienen el mismo tamaño, lo que provoca que el resultado exacto de la suma podría no ser un número pre-procesado. Un módulo de suma en coma fija 100SFJ,o 200SFJ, recibe los N MSBs de un primer, y un segundo, número en coma fija pre-procesado, en una primera, y una segunda entrada, respectivamente, teniendo cada número en coma fija pre-procesado N+1 bits. El LSB de los números en coma fija pre-procesados es igual a uno. El módulo de suma en coma fija 100SFJ, o 200SFJ, genera un tercer número en coma fija pre-procesado correspondiente a la suma redondeada de ambos números de entrada sin sesgo. Un módulo de suma en coma fija 100SFJ, o 200SFJ, comprende un sumador 120SFJ (o 220SFJ), el cual genera los N-1 MSBs del tercer número en coma fija pre-procesado. El enésimo MSB se fija a cero, mientras que el LSB es de nuevo igual a uno y no necesita ser generado, ni devuelto. En la Fig. 2a el sumador 120SFJ podría producir N bits, pero solamente los N-1 MSBs de su salida son usados, mientras el acarreo de entrada Cin se conecta a 1. En Fig. 2b el sumador 220SFJ tiene N-1 bits y el acarreo de entrada se conecta a una puerta OR 225SFJ con las dos entradas conectadas al enésimo MSBs del primer y segundo número en coma fija pre-procesado, respectivamente. En una implementación alternativa, si el sesgo no es un problema, el enésimo MSB del tercer número en coma fija pre-procesado podría ser generado por el sumador en lugar de

fijarse a cero. En otra implementación alternativa, mostrada en Fig. 2c, el módulo de suma podría estar configurado para producir el resultado exacto de la suma, el cual es un número no procesado, sacando explícitamente el LSB fijado a cero, junto con la salida del sumador 120SNFXFJ.

Por otro lado, existen dos casos diferentes cuando uno de los números de entrada es un número no procesado. Cuando el tamaño del número de entrada no procesado es igual o mayor que el tamaño del número pre-procesado, el resultado exacto de la suma podría ser un número no procesado. La implementación de un módulo de suma en coma fija configurado para recibir los N MSBs de un primer número en coma fija pre-procesado de N+1 bits y los N+M+1 bits de un segundo número en coma fija, éste no procesado, podría ser similar al circuito mostrado en Fig. 1a. Sin embargo, en este caso, no hay un LSB implícito en la salida, la cual, en este caso, es un número en coma fija no procesado de N+M+1 bits. Si se desea un número de salida pre-procesado, un conversor de números no procesados a números pre-procesados, similar a alguno de los descritos posteriormente aquí, podría ser usado. Por otro lado, cuando el tamaño del número de entrada no procesado es menor que el tamaño del número pre-procesado, el resultado exacto de la suma es un número pre-procesado. En este caso, el módulo de suma en coma fija está configurado para recibir los N bits de un primer número en coma fija no procesado y los N+M MSBs de un segundo número en coma fija, éste pre-procesado, de N+M+1 bits. Los N MSBs del resultado se obtienen sumando los N bits del primer número y los N MSBs del segundo número, mientras que los M+1 LSBs son los M+1 LSBs del segundo número. Este último incluye el LSB que es implícito e igual a uno. Como el resultado es un número pre-procesado, una salida con menos bits, redondeada al más cercano, podría obtenerse simplemente truncando dicho resultado.

Fig. 3 ilustra un restador en coma fija de acuerdo a un ejemplo. Un módulo de resta en coma fija 100SUBFJ recibe los m MSBs, y los n MSBs, de un primer, y un segundo, número en coma fija pre-procesado de m+1, y n+1 bits, en una primera, y una segunda entrada, respectivamente, y genera un tercer número en coma fija pre-procesado de z+1 bits correspondiente al primer número de entrada menos el segundo. El LSB de los números en coma fija pre-procesados es igual a uno, y no necesitan introducirse o

generarse. El módulo de resta en coma fija 100SUBFJ comprende un módulo de suma en coma fija pre-procesado 120SUBFJ, similar a los presentados antes, configurado para recibir dicha primera entrada y el inverso bit a bit de dicha segunda entrada, realizada con el inversor de bit 125SUBFJ, lo que en la práctica niega el segundo número pre-procesado. La salida de z bits de dicho módulo de suma en coma fija corresponde a los z MSBs del resultado de la resta, mientras que su LSB es implícito e igual a uno. Una implementación muy similar es mostrada en Fig. 4, la cual corresponde a módulo de suma/resta en coma fija pre-procesado 100ADDSUBFJ. El inversor bit a bit es sustituido por un inversor de bit condicional 105ADDSUBFJ, para invertir selectivamente la segunda entrada. Por lo tanto, dicho módulo produce la suma o resta deseada de los números de entrada de acuerdo a una señal de control c1.

En los siguientes ejemplos de multiplicadores (incluyendo elevadores al cuadrado y multiplicadores por constante), se considera, a menos que se afirme algo diferente, que los números en coma fija son sin signo. Sin embargo, alguien experto en el arte podría apreciar que números en complemento a dos podrían ser operados en su lugar, haciendo modificaciones conocidas a los circuitos descritos, tal como extensión de signo, en lugar de extensión con cero, para las sumas.

Fig. 5a ilustra una implementación de un módulo de multiplicación en coma fija para números pre-procesados de acuerdo a un ejemplo. Un módulo de multiplicación en coma fija para números pre-procesados 100MFJ recibe los m MSBs y los n MSBs de un primer y un segundo número en coma fija pre-procesado, de m+1 y n+1 bits, en una primera y segunda entrada, respectivamente, y genera un tercer número en coma fija pre-procesado de z+1 bits correspondiente a la multiplicación de ambos números de entrada. El LSB de los números en coma fija pre-procesados es igual a uno y no es necesario introducirlo a la entrada de dicho módulo. El módulo de multiplicación en coma fija para números pre-procesados 100MFJ comprende un multiplicador en coma fija 110MFJ configurado para recibir dichas primera y segunda entrada, aumentadas un bit por la derecha con el LSB de los números pre-procesados, y generar los m+n+1 MSBs de la multiplicación de ambos números. La introducción de este uno adicional podría realizarse internamente al multiplicador sin necesitar una entrada especial. Estos son meramente ilustrados para indicar que el multiplicador debe tenerlos en

cuenta cuando realice la operación de multiplicación. Los z MSBs de la salida del multiplicador 110MFJ corresponden a los z MSBs del tercer número en coma fija pre-procesado. El LSB es igual a uno y no necesita almacenarse o generarse. En implementaciones alternativas el multiplicador coma fija podría simplemente generar el producto de la primera y segunda entrada del módulo de multiplicación, y dicho producto podría ser sumado con dichas primera y segunda entrada desplazadas un bit a la derecha, para producir el resultado correcto, correspondiente al producto de los números de entrada (completos).

Como solamente los z MSBs de la multiplicación son devueltos, el circuito, multiplicador podría ser optimizado evitando el cálculo de los LSBs. Fig. 5b ilustra un ejemplo de implementación de un multiplicador coma fija para números pre-procesados, el cual evita la generación de dichos LSBs. El multiplicador coma fija 200MFJ comprende un multiplicador redundante 205MFJ, un módulo de red de acarreo 207MFJ y un módulo de conversión 209MFJ. El multiplicador redundante 205MFJ recibe, en una primera y segunda entrada, los m MSBs y los n MSBs del primer y el segundo número en coma fija pre-procesado, de m+1 y n+1 bits, respectivamente, y dos entradas adicionales conectadas a uno, tal que dichos bits de la primera y segunda entrada se aumentan un bit por la derecha. Sin embargo, en una implementación alternativa, la introducción del uno adicional podría realizarse internamente al módulo 205MFJ sin necesitar una entrada especial. Esto es meramente ilustrado en el ejemplo de Fig. 5b, y en otros ejemplos siguientes, para indicar la necesidad de la introducción funcional del LSB implícito. El multiplicador redundante 205MFJ genera, en un formato de representación redundante, los n+m+1 MSDs del valor correspondiente a la operación de multiplicación entre dichos números pre-procesados. El LSD de dicho resultado es siempre uno y no se requiere explícitamente. El multiplicador redundante 205MFJ mostrado en la Fig. 5b genera el resultado en formato de acarreo almacenado, y entonces dicho resultado se entrega en una primera y una segunda salida de n+m+1 bit cada una, correspondientes a la palabra de suma y a la palabra de acarreo, respectivamente. Sin embargo, un experto en la materia podría apreciar que, con modificaciones menores de los circuitos presentados, podrían usarse otros formatos de representación redundante, tal como representación de dígitos con signo.

El módulo de red de acarreo 207MFJ recibe los n+1 LSDs de la salida de dicho multiplicador redundante, los cuales no incluyen el LSD implícito del formato pre-procesado, y genera el bit de acarreo correspondiente a la conversión de dichos dígitos a una representación binaria no redundante. En este ejemplo particular, como se usa representación de acarreo almacenado, el módulo de red de acarreo 207MFJ recibe los n+1 LSBs de las palabras de suma y acarreo, en una primera y segunda entrada, respectivamente, y genera el último bit de acarreo correspondiente a la suma de ambas entradas.

El módulo de conversión 209MFJ recibe los m MSDs de la salida del multiplicador redundante 205MFJ y el bit de acarreo, desde el módulo de red de acarreo 207MFJ, y genera los m bits correspondientes a los m MSBs del valor de la multiplicación de las mantisas de entrada en una representación no redundante. En este ejemplo particular, como se usa representación de acarreo almacenado, el módulo de conversión 209MFJ recibe los m MSBs de las palabras de suma y acarreo, en una primera y una segunda entrada, respectivamente, y el bit de acarreo, en una tercera entrada, y genera un valor correspondiente a la suma de ambas palabras de entrada y el bit de acarreo. Además, en este ejemplo particular, el tamaño de la salida y de la primera entrada son iguales, pero en una implementación alternativa, el tamaño de la salida podría ser z+1 bits, siendo z<n+m+1. En este caso, módulo de red de acarreo 207MFJ podría recibir los n+m-z+1 LSDs de la salida del multiplicador redundante, y el módulo de conversión 209MFJ, los z MSDs.

Fig. 6a y 6b ilustran las implementaciones de un multiplicador redundante para números pre-procesados 300MFJ, y 400MFJ, respectivamente, en las cuales no se recibe el LSB de los números de entrada. El multiplicador redundante para números pre-procesados representado en Fig. 6a, y Fig. 6b, recibe solamente los m MSBs, y los n MSBs, de un primer, y un segundo, número coma fija pre-procesado (X e Y) de m+1, y n+1 bits, respectivamente, ya que el LSB es constante e igual a uno. Dicho multiplicador redundante genera, en una representación redundante, los m+n+1 MSDs del resultado de la multiplicación de ambos números de entrada, siendo el LSB de dicho resultado también implícito e igual a uno. Dicho de otra forma, si los m MSBs de X se representan por X', y los m MSBs de Y por Y', entonces el valor a la salida de

n+m+1 dígitos es igual a X'*Y'+1/2X'+1/2Y'.

El multiplicador redundante para números pre-procesados representado en Fig. 6a comprende un módulo generador de productos parciales 325MFJ y un árbol de compresores 330MFJ. El módulo generador de productos parciales 325MFJ recibe dichos m MSBs, y n MSBs, de los dos números coma fija pre-procesados, en una primera y una segunda entrada, respectivamente, y genera los productos parciales correspondientes al producto de la primera entrada por cada bit de la segunda entrada. En una implementación alternativa, la segunda entrada podría estar dividida en varios grupos de bits y los productos parciales generados podrían corresponder a los productos de la primera entrada por cada dicho grupo de bits.

El árbol de compresores 330MFJ recibe la salida del módulo generador de productos parciales 325MFJ y una copia de las dos entradas del módulo generador de productos parciales 325MFJ, y genera una salida de m+n+1 dígitos redundantes, correspondiente a la suma de todas sus entradas correctamente alineadas. Debemos notar que dichas copias están alineadas de tal forma que el segundo LSB está alineado con el LSB del producto parcial menos significativo (aquel correspondiente al LSB de la segunda entrada). En este ejemplo particular, como se usa representación de acarreo almacenado, se producen dos números de m+n+1 bits correspondientes a las palabras de suma y acarreo. En una implementación alternativa, un formato de representación redundante diferente podría ser usado. En otras implementaciones, si se desea una salida no redundante, un módulo de conversión podría ser usado para transformar la salida del árbol de compresores 330, a un número no redundante de m+n+1 bit correspondiente a los m+n+1 MSBs del producto de los números pre-procesados iniciales.

El multiplicador redundante para números pre-procesados representado en Fig. 6b es similar al anterior, pero la segunda entrada se recodifica (por ejemplo, mediante recodificación de Booth) antes de entrar en el generador de productos parciales 325bMFJ para producir menos productos parciales, mediante el uso del módulo de recodificación 320bMFJ. El valor uno se inserta también en la entrada del módulo de recodificación 320bMFJ, tal que los n bits de la segunda entrada son aumentados por

la derecha con un bit correspondiente al LSB implícito. Sin embargo, en otras implementaciones, la introducción del uno adicional podría realizarse internamente al módulo de recodificación 320bMFJ, sin necesidad de una entrada especial. Esto es meramente ilustrado en el ejemplo para indicar la necesidad de la introducción funcional del LSB implícito. De forma similar, el LSB de la otra entrada está también ilustrado en la primera entrada del generador de productos parciales 325bMFJ.

Las arquitecturas mostradas con referencia a Fig. 5a a 6b, podrían ser implementados para números o bien, sin signo, o bien, con signo, usando los módulos adecuados en consonancia, tal como multiplicadores en punto fijo para número sin signo, o para números con signo. Sin embargo un enfoque diferente podría ser utilizado para implementar módulos de multiplicación para número pre-procesados con signo. Éste está basado en usar la versión sin signo de cualquiera de los ejemplos mostrados anteriormente y la conversión de los números de entrada de complemento a dos al formato signo-magnitud. Esta conversión podría implementarse fácilmente, para números pre-procesados, usando un inversor de bit condicional para invertir los N-1 LSBs de los N MSBs de un número pre-procesado de N+1 bits, si este es negativo. Entonces, la magnitud podría ser operada por el módulo de multiplicación para números sin signo, mientras que el signo podría ser operado aparte. Finalmente, una conversión del resultado en signo-magnitud a un número en complemento a dos, la cual es similar a la anterior, es requerida. Además, un experto en la técnica podría apreciar que podría ser fácil modificar este diseño para soportar ambos formatos en la misma unidad.

Fig. 7a y 7b ilustran las implementaciones de un módulo de elevar al cuadrado en coma fija para números pre-procesados de acuerdo a dos ejemplos, considerando una entrada sin signo. Un módulo de elevar al cuadrado en coma fija para números pre-procesados 100SQFJ, ó 100bSQFJ, recibe los m MSBs de un primer número en coma fija pre-procesado de m+1 bits, en una primera entrada, y genera un segundo número en coma fija pre-procesado de z+1 bits correspondiente a elevar al cuadrado el número de entrada. El LSB de los números en coma fija pre-procesados es igual a uno y no es necesario introducirlo a la entrada de dicho módulo. El módulo de elevar al cuadrado en coma fija para números pre-procesados 100SQFJ de la Fig. 7a comprende un elevador

al cuadrado en coma fija 110SQFJ configurado para recibir dicha primera entrada, aumentada un bit por la derecha con el LSB del número pre-procesado, y generar los 2m MSBs del cuadrado de dicho número. La introducción de este uno adicional podría realizarse internamente al elevador al cuadrado sin necesitar una entrada especial. Este es ilustrados aparte simplemente para indicar que el elevador al cuadrado debe tenerlo en cuenta cuando realice la operación. La salida del elevador al cuadrado 110SQFJ es aumentada por la derecha con un bit fijado a cero, correspondiente al segundo LSB del resultado de la operación de elevar al cuadrado. Dicho bit a cero podría ser sacado por el elevador al cuadrado (o incluso evitado, si $z<2m+1$), aquí, es ilustrado aparte para indicar que su cálculo no es necesario. Los z MSBs de dicha salida del elevador a cuadrado aumentada corresponden a los z MSBs del segundo número en coma fija pre-procesado. El LSB es igual a uno y no necesita almacenarse o generarse.

Como alternativa, el módulo de elevar al cuadrado en coma fija para números pre-procesados 100bSQFJ de la Fig. 7b comprende un elevador al cuadrado en coma fija 110bSQFJ configurado para recibir tan solo dicha primera entrada y generar los 2m bits del cuadrado de dicho valor de entrada. Un sumador 120bSQFJ es usado para incorporar el efecto del LSB implícito del número de entrada, sumando los m MSBs de dicho número de entrada pre-procesado, alineados hacia la derecha, a la salida del elevador al cuadrado 110bSQFJ. En otras implementaciones, dicha suma podría realizarse internamente al elevador al cuadrado 110bSQFJ. De manera similar al ejemplo de la Fig. 7a, la salida del sumador 120bSQFJ podría ser aumentado hacia la derecha con un bit a cero si $z>2m$. Los z MSBs de la salida del sumador 120bSQFJ (aumentado si fuese necesario) corresponden a los z MSBs del segundo número en coma fija pre-procesado. El LSB es igual a uno y no necesita almacenarse o generarse.

Como solamente los z MSBs del cuadrado son devueltos, el circuito elevador al cuadrado podría ser optimizado evitando el cálculo de los LSBs. Fig. 7c ilustra un ejemplo de implementación de un elevador al cuadrado en coma fija para números pre-procesados, el cual evita la generación de dichos LSBs. El elevador al cuadrado en coma fija 300SQFJ comprende un módulo de elevar al cuadrado redundante 305SQFJ, un módulo de red de acarreo 307SQFJ y un módulo de conversión 309SQFJ. El

módulo de elevar al cuadrado redundante 305SQFJ recibe, en una primera entrada, los m MSBs de un primer número en coma fija pre-procesado de m+1 bits, y una entrada adicional conectada a 1, tal que los m bits en la entrada se aumentan un bit por la derecha. Sin embargo, en una implementación alternativa, la introducción del uno adicional podría realizarse internamente al módulo 205SQFJ sin necesitar una entrada especial. Esto es meramente ilustrado en el ejemplo de Fig. 7c, y en otros ejemplos siguientes, para indicar la necesidad de la introducción funcional del LSB implícito. El módulo de elevar al cuadrado redundante 305SQFJ genera, en un formato de representación redundante, los 2m MSDs del valor correspondiente al cuadrado del número de entrada. El segundo LSD, y el LSD, de dicho resultado son siempre cero, y uno, respectivamente, y no se requieren explícitamente. El módulo de elevar al cuadrado redundante 305SQFJ mostrado en la Fig. 7c genera el resultado en formato de acarreo almacenado y entonces dicho resultado se entrega en una primera y una segunda salida de 2m bit cada una, correspondientes a la palabra de suma y a la palabra de acarreo, respectivamente. Sin embargo, un experto en la materia podría apreciar que, con modificaciones menores de los circuitos presentados, podrían usarse otros formatos de representación redundante, tal como representación de dígitos con signo.

El módulo de red de acarreo 307SQFJ recibe los 2m-z LSDs de la salida de dicho módulo de elevar al cuadrado redundante 305SQFJ, y genera el bit de acarreo correspondiente a la conversión de dichos dígitos a una representación binaria no redundante. En este ejemplo particular, como se usa representación de acarreo almacenado, el módulo de red de acarreo 307SQFJ recibe los 2m-z LSBs de las palabras de suma y acarreo, en una primera y segunda entrada, respectivamente, y general el último bit de acarreo correspondiente a la suma de ambas entradas.

El módulo de conversión 309SQFJ recibe los z MSDs de la salida del módulo de elevar al cuadrado redundante 305SQFJ y el bit de acarreo desde el módulo de red de acarreo 307SQFJ, y genera los z bits correspondientes a los z MSBs del valor del número en coma fija de entrada al cuadrado, en una representación no redundante. En este ejemplo particular, como se usa representación de acarreo almacenado, el módulo de conversión 309SQFJ recibe los z MSBs de las palabras de suma y acarreo, en una

primera y una segunda entrada, respectivamente, y el bit de acarreo, en una tercera entrada, y genera un valor correspondiente a la suma de ambas palabras de entrada y el bit de acarreo.

Fig. 8 ilustra una implementación de un módulo de elevar al cuadrado redundante para números pre-procesados de acuerdo a un ejemplo, en el cual no se recibe el LSB del número de entrada. Por tanto, dicho módulo recibe solamente los m MSBs de un número coma fija pre-procesado (X), ya que el LSB es constante e igual a uno. Dicho módulo de elevar al cuadrado redundante 405SQFJ para números pre-procesados genera, en una representación redundante, los 2m MSDs del resultado de elevar al cuadrado el número de entrada pre-procesado, siendo el segundo LSB y el LSB de dicho resultado, implícitos e igual a cero y uno, respectivamente. Dicho de otra forma, si los m MSBs de X se representan por X', entonces el valor a la salida de 2m dígitos es igual a X'^2+X'. El módulo de elevar al cuadrado redundante para números pre-procesados 405SQFJ comprende un módulo generador de productos parciales 425SQFJ y un árbol de compresores 430SQFJ. El módulo generador de productos parciales 425SQFJ recibe dichos m MSBs del número coma fija pre-procesados, en una primera entrada, y genera un conjunto de productos parciales, los cuales permiten, sumándolos, obtener un valor correspondiente al cuadrado de dicha primera entrada (es decir, X'^2). Alguien experto en la materia podría apreciar que hay diferentes conjuntos de productos parciales que podrían utilizarse dependiendo del grado de optimización deseado.

El árbol de compresores 430SQFJ recibe la salida del módulo generador de productos parciales 425SQFJ y una copia de los m MSBs del número de entrada pre-procesado, y genera una salida de 2m dígitos redundantes correspondientes a la suma de todas sus entradas correctamente alineadas. Debemos notar que dichos m MSBs están alineados de tal forma que su LSB está alineado con el LSB del producto parcial menos significativo. En una implementación alternativa, dichos m MSBs podrían ser introducidos internamente en el árbol de compresores 430SQFJ, o en el módulo generador de productos parciales 425SQFJ. En este ejemplo particular, como se usa representación de acarreo almacenado, se producen dos números de 2m bits correspondientes a las palabras de suma y acarreo. En una implementación alternativa,

un formato de representación redundante diferente podría ser usado. En otras implementaciones, si se desea una salida no redundante, un módulo de conversión podría ser usado para transformar la salida del árbol de compresores 430SQFJ, a un número no redundante de 2m bits, correspondiente a los 2m MSBs del cuadrado del número pre-procesado inicial.

En los ejemplos mostrados en las Fig. 7a, 7b, 7c y 8, el número pre-procesado de entrada es considerado sin signo. Sin embargo, en implementaciones alternativas de esos ejemplos, el número pre-procesado de entrada, podría ser con signo. En ese caso, el elevador al cuadrado usado podría estar configurado específicamente para soportar el cálculo del cuadrado de números con signo, en lugar de para números sin signo. Además, las extensiones con ceros requeridas por las sumas, tales como las del ejemplo de la Fig. 7b, deberían sustituirse por una extensión de signo. Sin embargo, una solución diferente es presentada en el ejemplo de la Fig. 9. Dicha Fig. 9 ilustra la implementación de un módulo de elevar al cuadrado en coma fija 500SQFJ para números pre-procesados con signo de acuerdo a un ejemplo. El módulo de elevar al cuadrado en coma fija para números pre-procesados con signo 500SQFJ recibe los m MSBs de un primer número en coma fija pre-procesado de m+1 bits, y en complemento a dos, en una primera entrada, y genera un segundo número en coma fija pre-procesado de z+1 bits, y en complemento a dos, correspondiente a elevar al cuadrado el número de entrada. El LSB de los números en coma fija pre-procesados es igual a uno, y no es necesario introducirlo a la entrada, o generarlo a la salida, de dicho módulo. El módulo de elevar al cuadrado en coma fija para números pre-procesados con signo 500SQFJ de la Fig. 9 comprende un inversor de bits condicional 510SQFJ y un módulo de elevar al cuadrado en coma fija para números pre-procesados 520SQFJ para números sin signo de m-1 bits, similar a los presentados en los ejemplos anteriores. Los m-1 LSBs de la entrada se introducen en el inversor de bits condicional 510SQFJ. El MSB de dicha entrada, que es el signo del número de entrada pre-procesado, se utiliza para controlar el inversor de bit condicional 510SQFJ. El inversor de bit condicional 510SQFJ llevará a cabo una inversión bit a bit de dichos m-1 bits, si dicho bit de signo es igual a uno. Por tanto, la salida del inversor de bits condicional 510SQFJ, junto con el LSB implícito, corresponden con la magnitud del número pre-procesado de entrada, ya que dicho número es negado si es negativo. La salida del

inversor de bit condicional 510SQFJ de m-1 bits se conecta al módulo de elevar al cuadrado en coma fija para números pre-procesados 520SQFJ, el cual genera los z-1 MSBs del cuadrado de dicha magnitud. La salida del módulo de elevar al cuadrado en coma fija para números pre-procesados 520SQFJ, aumentada por la izquierda con el bit de signo, el cual es siempre cero, corresponde a los z MSBs del segundo número en coma fija pre-procesado, y en complemento a dos. El LSB es igual a uno y no necesita almacenarse o generarse.

Los ejemplos mostrados en las figuras Fig. 7a a 8 son para números sin signo, mientras que el de la Fig. 9 es exclusivamente para números con signo. Sin embargo, alguien experto en la técnica podría apreciar que es posible diseñar, con mínimas modificaciones, una nueva arquitectura, combinándolos, para soportar ambos formatos en la misma unidad.

Fig. 10a y 10b ilustran las implementaciones de un módulo de multiplicación por constante en coma fija para números pre-procesados de acuerdo a dos ejemplos. Un módulo de multiplicación por constante en coma fija para números pre-procesados 100MCFJ, ó 200MCFJ, recibe los m MSBs de un primer número en coma fija pre-procesado de m+1 bits, en una primera entrada, y genera un segundo número en coma fija pre-procesado de z+1 bits, correspondiente a la multiplicación del número de entrada por una constante en coma fija pre-procesada de n+1 bits. El LSB de los números en coma fija pre-procesados es igual a uno y no es necesario introducirlo a la entrada de dicho módulo. El módulo de multiplicación por constante en coma fija para números pre-procesados 100MCFJ de la Fig. 10a comprende un multiplicador por constante en coma fija 110MCFJ, configurado para recibir dicha primera entrada, aumentada un bit por la derecha con el LSB del número pre-procesado, y generar los m+n+1 MSBs de la multiplicación de dicho número por dicha constante. La introducción de este uno adicional podría realizarse internamente al multiplicador sin necesitar una entrada especial. Esto es meramente ilustrado para indicar que el multiplicador debe tenerlo en cuenta cuando realice la operación de multiplicación. Los z MSBs de la salida del multiplicador por constante 110MCFJ corresponden a los z MSBs del segundo número en coma fija pre-procesado. El LSB es igual a uno y no necesita almacenarse o generarse.

Como alternativa, el módulo de multiplicación por constante en coma fija para números pre-procesados 200MCFJ de la Fig. 10b comprende un multiplicador por constante en coma fija 110bMCFJ configurado para recibir, tan solo, dicha primera entrada y generar los m+n+1 bits de la multiplicación de dicho entrada y dicha constante. Un sumador 120bMCFJ es usado para incorporar el efecto del LSB implícito del número de entrada, sumando los n MSBs de la constante, alineados hacia la derecha, a la salida del multiplicador por constante 110bMCFJ. En el ejemplo de la Fig. 10b una constante sin signo es supuesta, pero extensión de signo, en lugar de extensión con ceros, podría utilizarse para constantes con signo. En otras implementaciones, un sumador de constante, optimizado para sumar el valor constate, a su único valor de entrada, podría ser usado, en lugar del sumador 120bMCFJ y la constante externa. En otras implementaciones, dicha suma podría realizarse internamente al multiplicador por constante 110bMCFJ. Los z MSBs de la salida del sumador de constante 120bMCFJ corresponden a los z MSBs del segundo número en coma fija pre-procesado. El LSB es igual a uno y no necesita almacenarse o generarse.

En implementaciones alternativas de los ejemplos de Fig. 10a y 10b la constante deseada podría no ser un número pre-procesado, porque su LSB podría no ser uno. Sin embargo, todos los LSBs antes del primer bit igual a uno podrían ser eliminados para generar una constante pre-procesada. En algunas implementaciones, esos LSBs igual a cero podrían ser añadidos a la derecha de la salida del multiplicador por constante 110MCFJ, o 110bMCFJ, si alguno de esos bits corresponden a la parte entera del número, para generar el resultado correcto. En este caso, el resultado podría ser un número no pre-procesado. En algunas implementaciones un conversor de números no procesados a números pre-procesados podría ser usado. En otros, el número no procesado podría ser el número de salida.

Como solamente los z MSBs de la multiplicación son devueltos, el circuito multiplicador podría ser optimizado, evitando el cálculo de los LSBs. Fig. 10c ilustra un ejemplo de implementación de un multiplicador por constante en coma fija para números pre-procesados, el cual evita la generación de dichos LSBs. El multiplicador por constante en coma fija 300MCFJ comprende un módulo de multiplicación por constante

redundante 305MCFJ, un módulo de red de acarreo 307MCFJ, y un módulo de conversión 309MCFJ. El módulo de multiplicación por constante redundante 305MCFJ recibe, en una primera entrada, los m MSBs del primer número en coma fija pre-procesado, y una entrada adicional conectada a 1, tal que los m bits en la entrada se aumentan un bit por la derecha. Sin embargo, en una implementación alternativa, la introducción del uno adicional podría realizarse internamente al módulo 305MCFJ sin necesitar una entrada especial. Esto es meramente ilustrado en el ejemplo de Fig. 10c, y en otros ejemplos siguientes, para indicar la necesidad de la introducción funcional del LSB implícito. El módulo de multiplicación por constante redundante 305MCFJ genera, en un formato de representación redundante, los n+m+1 MSDs del valor correspondiente a la operación de multiplicación entre el número de entrada pre-procesado y una constante en coma fija pre-procesada de n+1 bits. El LSD de dicho resultado es siempre uno y no se requiere explícitamente. El módulo de multiplicación por constante redundante 305MCFJ mostrado en la Fig. 10c genera el resultado en formato de acarreo almacenado y, entonces, dicho resultado se entrega en una primera y una segunda salida de n+m+1 bit cada una, correspondientes a la palabra de suma y a la palabra de acarreo, respectivamente. Sin embargo, un experto en la materia podría apreciar que, con modificaciones menores de los circuitos presentados, podrían usarse otros formatos de representación redundante, tal como representación de dígitos con signo.

El módulo de red de acarreo 307MCFJ recibe los n+1 LSDs de la salida de dicho módulo de multiplicación por constante redundante 305MCFJ, la cual no incluye el LSD implícito del formato pre-procesado, y genera el bit de acarreo correspondiente a la conversión de dichos dígitos a una representación binaria no redundante. En este ejemplo particular, como se usa representación de acarreo almacenado, el módulo de red de acarreo 307MCFJ recibe los n+1 LSBs de las palabras de suma y acarreo, en una primera y segunda entrada, respectivamente, y general el último bit de acarreo correspondiente a la suma de ambas entradas.

El módulo de conversión 309MCFJ recibe los m MSDs de la salida del módulo de multiplicación por constante redundante 305MCFJ y el bit de acarreo desde el módulo de red de acarreo 307MCFJ, y genera los m bits correspondientes a los m MSBs del

valor de la multiplicación del número en coma fija de entrada y la constante, en una representación no redundante. En este ejemplo particular, como se usa representación de acarreo almacenado, el módulo de conversión 309MCFJ recibe los m MSBs de las palabras de suma y acarreo, en una primera y una segunda entrada, respectivamente, y el bit de acarreo, en una tercera entrada, y genera un valor correspondiente a la suma de ambas palabras de entrada y el bit de acarreo. Además, en este ejemplo particular, el tamaño de la salida y de la primera entrada son iguales, pero en una implementación alternativa, el tamaño de la salida podría ser z+1 bits, siendo z<n+m+1. En este caso, módulo de red de acarreo 307MCFJ podría recibir los n+m-z+1 LSDs de la salida del multiplicador redundante, y el módulo de conversión 309MCFJ, los z MSDs.

Fig. 11 ilustra una implementación de un módulo de multiplicación por constante redundante para números pre-procesados 405MCFJ de acuerdo a un ejemplo, en el cual no se recibe el LSB del número de entrada. Por tanto, dicho módulo recibe solamente los m MSBs de un número coma fija pre-procesado (X), ya que el LSB es constante e igual a uno. Dicho módulo de multiplicación por constante redundante para números pre-procesados genera, en una representación redundante, los m+n+1 MSDs del resultado de la multiplicación entre el número de entrada pre-procesado y una constante en coma fija pre-procesada de n+1 bits (Y), siendo el LSB de dicho resultado también implícito e igual a uno. Dicho de otra forma, si los m MSBs de X se representan por X' y los m MSBs de Y por Y', entonces el valor a la salida de n+m+1 dígitos es igual a X'*Y'+1/2X'+1/2Y'. El módulo de multiplicación por constante redundante para números pre-procesados 405MCFJ comprende un módulo generador de productos parciales 425MCFJ y un árbol de compresores 430MCFJ. El módulo generador de productos parciales 425MCFJ recibe dichos m MSBs del número coma fija pre-procesados, en una primera entrada, y genera un conjunto de productos parciales, los cuales permiten, sumándolos, obtener un valor correspondiente al producto de dicha primera entrada por los n MSBs de la constante pre-procesada (es decir, X'*Y'). Alguien experto en la materia podría apreciar que hay diferentes conjuntos de productos parciales que podrían utilizarse dependiendo del grado de optimización deseado. Además, en una implementación alternativa, el módulo generador de productos parciales podría estar configurado para tener en cuenta, además, el LSB de la constante para producir dichos productos parciales (es decir, generar X'*Y'+1/2X').

El árbol de compresores 430MCFJ recibe la salida del módulo generador de productos parciales 425MCFJ, una copia de entrada de m bits, y los n MSBs de la constante pre-procesada, y genera una salida de m+n+1 dígitos redundantes correspondiente a la suma de todas sus entradas correctamente alineadas. Debemos notar que dicha copia y dichos n MSBs están alineados de tal forma que su segundo LSB está alineado con el LSB del producto parcial menos significativo. En una implementación alternativa dicha copia y dichos n MSBs de la constante pre-procesada podrían ser introducidos internamente en el árbol de compresores 430MCFJ, o en el módulo generador de productos parciales 425MCFJ. En este ejemplo particular, como se usa representación de acarreo almacenado, se producen dos números de m+n+1 bits, correspondientes a las palabras de suma y acarreo. En una implementación alternativa, un formato de representación redundante diferente podría ser usado. En otras implementaciones, si se desea una salida no redundante, un módulo de conversión podría ser usado para transformar la salida del árbol de compresores 430MCFJ, a un número no redundante de m+n+1 bit correspondiente a los m+n+1 MSBs del producto del número pre-procesado inicial y la constante.

Las arquitecturas mostradas con referencia desde Fig. 10a a 11, podrían ser implementados para números o bien, sin signo, o bien, con signo, usando los módulos adecuados en consonancia, tal como multiplicadores por constante en coma fija, para número sin signo, o para números con signo, y sustituyendo la extensión con ceros requerida por la suma, tal como la del ejemplo de la Fig. 10b, por extensión de signo. Sin embargo un enfoque diferente podría ser utilizado para implementar módulos de multiplicación por constante para número pre-procesados con signo. Este podría estar basado en el uso de la versión sin signo de cualquiera de los ejemplos mostrados anteriormente y la conversión de los números de entrada en complemento a dos al formato signo-magnitud. Esta conversión se implementa fácilmente para números pre-procesados usando un inversor de bit condicional para invertir los N-1 LSBs de los N MSBs de un número pre-procesado de N+1 bits, si éste es negativo. Entonces, la magnitud podría ser procesada por el módulo de multiplicación por constante para números sin signo, mientras que el signo es procesado aparte. Finalmente, una conversión del resultado en signo-magnitud a un número en complemento a dos, la cual es similar a la anterior, es requerida. Además, un experto en la técnica podría

apreciar que es fácil modificar este diseño para soportar ambos formatos en la misma unidad.

La implementación de un desplazador a la izquierda pre-procesado es descrita en Fig. 12, de acuerdo a un ejemplo. Como el desplazamiento a la izquierda de un número en coma fija pre-procesado produce un número no procesado, un redondeo al más cercano es requerido. El desplazador a la izquierda pre-procesado 100SHFJ realiza el desplazamiento a la izquierda de un número en coma fija pre-procesado, sin introducir sesgo debido al redondeo. El desplazador a la izquierda pre-procesado 100SHFJ recibe los n MSBs de un primer número en coma fija pre-procesado de n+1 bits, en una primera entrada, una cantidad de desplazamiento, en una segunda entrada, y genera un segundo número en coma fija pre-procesado de n+1 bits, correspondiente al desplazamiento a la izquierda del número pre-procesado de entrada de acuerdo a la cantidad de desplazamiento. El LSB de los números pre-procesados es igual a 1 y no necesita ser introducido ni generado. El desplazador a la izquierda pre-procesado 100SHFJ comprende un desplazador variable a la izquierda especial 160SHFJ con una nueva entrada de un bit que permite seleccionar el valor usado para rellenar las posiciones vacantes después del desplazamiento. El desplazador variable a la izquierda especial 160SHFJ está configurado para recibir los n MSBs del primer número en coma fija pre-procesado aumentados por la derecha con un bit con un valor aleatorio, en una primera entrada, la cantidad de desplazamiento, en una segunda entrada, y el inverso de dicho bit aleatorio, en dicha nueva entrada, la tercera. De esta forma, las posiciones vacantes después del desplazamiento son rellenadas aleatoriamente, o bien con un bit a uno y los restantes bits a cero, o lo contrario, y no se produce sesgo. El bit aleatorio podría ser cualquier bit seleccionado, o combinación de bits seleccionados, del primer número en coma fija pre-procesado, o cualquier otro bit con las adecuadas características estadísticas. En otras implementaciones la cantidad de desplazamiento podría ser un valor constante y el desplazamiento podría ser cableado en lugar de usar un desplazador variable especial. En implementaciones alternativas, el tamaño de la salida podría no ser igual al tamaño de la entrada

Un ejemplo del desplazador variable a la izquierda especial 160SFH, mostrado en Fig. 12a, se implementa usando varios multiplexores dos a uno (log2 de la máxima cantidad

de desplazamiento requerida) conectados en serie, de tal forma que la salida de un desplazador es usada en la entrada del siguiente. Las entradas de datos del primer multiplexor son conectadas a la primera entrada del desplazador a la izquierda, a la posición no desplazada y a la desplazada (2^0), respectivamente, mientras que el bit de control se acopla al LSB de la cantidad de desplazamiento (segunda entrada). Las entradas de datos del segundo multiplexor se acoplan a la salida de las posiciones primera, no desplazada y desplazada en 2 (2^1), respectivamente, mientras el bit de control se acopla a al segundo LSB de la cantidad de desplazamiento (segunda entrada). El resto del multiplexor es conectado en concordancia. En los desplazadores a la izquierda convencionales las posiciones vacantes son completadas con ceros. En esta propuesta las posiciones vacantes son completadas con la tercera entrada (nueva entrada L). En este ejemplo, la máxima cantidad de desplazamiento es m-1. La salida del desplazador variable a la izquierda especial 160SHFJ comprende los m MSBs del valor desplazado.

Las unidades aritmética descritas arriba requieren números en coma fija que hayan sido pre-procesados de acuerdo a la invención como se describió arriba. Estos números pre-procesados podrían ser generados por circuitos, tales como las mencionadas unidades aritméticas, que están diseñadas para funcionar con números pre-procesados, o podrían ser generados por conversores, diseñados para convertir número no procesados o números pre-procesados, en un formato diferente, en números en coma fija pre-procesados. Además, los números pre-procesados generados por los sumadores descritos arriba podrían, en concordancia, requerir conversores tales que los números generados podrían ser usados por circuitos que no estén diseñados para operar números en ese formato.

En los siguientes ejemplos, se considera que los números en coma fija, tanto los no procesado como los procesados, son representados en representación en complemento a dos, siendo el MSB equivalente al bit de signo. De la misma forma, los números en coma flotante, tanto los no procesados como los pre-procesados, son representados por un bit de signo, un exponente y una mantisa normalizada sin signo, de tal forma que el MSB es igual a uno y está explícitamente incluido en la representación de la mantisa. Sin embargo, un experto en la técnica podría apreciar

que otros formatos que tienen una representación diferente podrían ser utilizados con modificaciones menores en los circuitos descritos. Algunas de estas variaciones podrían ser:

a) en coma fija: representación signo-magnitud, o representación sin signo

b) en FP

- representación implícita del MSB de la mantisa, o
- representación fusionada del signo y la mantisa mediante representación en complemento a dos o cualquier otra representación.

Una categoría de conversores son los conversores para convertir números en coma fija no procesados a números en coma fija pre-procesados. En un primer caso, el número original en coma fija es mayor que el número en coma fija objetivo. El conversor que se discutirá con referencia a la Fig. 14a podría ser utilizado, pero introduce algo de sesgo. En caso de redondeo sin sesgo, el nuevo número se calcula con el circuito ilustrado en la Fig. 13a. Para un número de n+m+1 bits, los n-1 MSBs son los mismos en el original y en el número en coma fija objetivo. El enésimo MSB del nuevo número se pone a cero si los m+1 LSBs del número original son todos cero, e igual al enésimo MSB del número original, en otro caso. El LSB del nuevo número será 1, ya que el número en coma fija es un número pre-procesado, aunque, en la figura, está implícito.

Cuando el número en coma fija pre-procesado tenga más bits (n+m+1) que la mantisa del número en coma fija no procesado (n), entonces:

a) en el caso del redondeo con sesgo, el número no procesado se expande con tantos ceros como sea necesario. Esto se ilustra en la Fig. 13b. El LSB será implícito e igual a 1.

b) en el caso de redondeo sin sesgo, los n-1 MSBs son los mismos. El enésimo bit se fuerza a cero. Los m+1 LSBs se fijan igual al LSB del número no procesado. Esto se ilustra en la Fig. 13c. El LSB del número pre-procesado será implícito e igual a 1.

Otra categoría de conversores son los conversores para convertir números en coma fija pre-procesados a números en coma fija pre-procesados de diferente tamaño. La Fig. 14a es un ejemplo de un conversor de este tipo. El conversor 800a ilustra un conversor adaptado para convertir un número en coma fija pre-procesado que tiene n+m+1 bits a

un número de n+1 bits. El LSB de ambas números es igual a 1 y, por lo tanto, no se representa. Los n MSBs del número original serán los n bits más significativos del número pre-procesada objetivo. Es decir, tiene lugar una simple función de truncamiento.

La Fig. 14b es otro ejemplo de un conversor de números pre-procesados en coma fija a números pre-procesados en coma fija. El conversor 800b ilustra un conversor adaptado para convertir un número en coma fija pre-procesado de m+1 bits a uno de n+m+1 bits. El conversor 800b es una versión con sesgo de un conversor de este tipo. Una vez más, el LSB de ambos números es igual a 1 y por lo tanto no se representa. De acuerdo con el conversor 800b, un circuito para ampliar el tamaño del número original,añadiendo a la derecha un bit a uno y tantos ceros como sea necesario para completar el nuevo tamaño del número.

La Fig. 14c es otro ejemplo de un conversor de pre-procesados en coma fija a pre-procesados en coma fija. El conversor 800c ilustra un conversor adaptado para convertir un número en coma fija pre-procesado con n+1 bits a uno de n+m+1 bits. El conversor 800c es una versión sin sesgo de un conversor de este tipo Una vez más, el LSB de ambos números es igual a 1 y por lo tanto no se representa. De acuerdo con el conversor 800c, un circuito para ampliar el tamaño del número añadiéndole a la derecha un bit con un valor aleatorio y tantos bits, con el inverso de dicho valor, como se requieran para completar el nuevo tamaño del número. El bit aleatorio podría ser cualquier bit del número inicial, o una combinación de ellos, tal como el segundo LSB, que es lo que se muestra en la Fig. 14c.

Otra categoría de conversores son los conversores para convertir números en coma fija pre-procesados a números en coma fija no procesados. Fig. 15 ilustra un ejemplo de un conversor 100CFJ, para convertir un número pre-procesado de n+m+1 bits a un número no procesado de n bits. Los n+1 MSBs del número de entrada son introducidos en un módulo de redondeo 120CFJ, para producir un número, no procesado y redondeado, de n bits correspondiente al valor de salida. El cálculo del bit de sticky correspondiente a los restantes m bit, no es requerido, ya que el LSB es siempre 1 y, entonces, el bit de sticky también es uno.

La Fig. 15b muestra un ejemplo de implementación de dicho conversor cuando el módulo de redondeo realiza redondeo al más cercano. El conversor 100bCFJ comprende un sumador 1310aCFJ, que se usa para incrementar en uno, los n MSBs de la entrada pre-procesada, si el (n+1)-ésimo MSB de dicha entrada es uno. Cuando m=0, es decir, el número pre-procesado de entrada tiene n+1 bits, el valor de entrada de n bits se aumenta con el LSB de dicho número, el cual es uno, antes de introducirlo en el módulo de redondeo. En implementaciones alternativas diferentes unidades de redondeo, que realizan diferentes modos de redondeo, podrían ser usadas. Por otro lado, el conversor adaptado para convertir un número en coma fija pre-procesado de m+1 bits a un número en coma fija no procesados de n+m bits es similar al descrito con referencia a la Fig. 14b, salvo que la salida no tiene ningún LSB implícito.

Otra categoría de conversores son los conversores para convertir números FP pre-procesados a números en coma fija pre-procesados. La Fig. 16a ilustra un conversor 900FJ para la conversión de un número FP, que tiene una mantisa de n+m+1 bits y un exponente de d bits, en un número en coma fija de n+2 bits. Los n bits más significativos de la mantisa son de entrada al inversor de bits condicional 905FJ. El LSB de la mantisa es igual a 1 y no se introduce. El signo del número FP pre-procesado se utiliza para controlar el inversor de bits condicional 905FJ. La salida del inversor de bits condicional 905FJ, junto con el signo (sign_x), se introduce en desplazador a la derecha 910FJ. El desplazador a la derecha 910FJ tiene otra entrada para recibir la cantidad de desplazamiento del calculador de cantidad de desplazamiento 915FJ. El calculador de cantidad de desplazamiento 915FJ recibe el exponente del número FP pre-procesado y genera la cantidad de desplazamiento. La salida del desplazador a la derecha 910FJ son los n+1 MSBs del número en coma fija pre-procesado. El LSB es, de manera similar, igual a 1 y no es ni generado ni representado. En una implementación alternativa el inversor de bit condicional podría estar detrás del desplazador a la derecha.

La Fig. 16b ilustra un conversor con sesgo para la conversión de un número FP pre-procesado, que tiene n+1 bits de mantisa y un exponente de d bits, a un número en coma fija pre-procesado de n+m+2 bits. Los n bits más significativos de la mantisa se

introducen en el inversor de bits condicional 1005aFJ. El LSB de la mantisa es igual a 1 y no se introduce. El signo del número FP pre-procesado se utiliza para controlar el inversor de bit condicional 1005aFJ. La salida del inversor de bits condicional 1005aFJ, junto con el signo (sign_x), y expandida es introducida en el desplazador a la derecha 1010aFJ. La salida del inversor de bits condicional 1005aFJ se expandió mediante la adición por la derecha de un bit a uno y tantos bits a cero como sean necesarios para completar el nuevo tamaño. El desplazador a la derecha 1010aFJ tiene otra entrada para recibir la cantidad de desplazamiento del calculador de cantidad de desplazamiento 1015aFJ. El calculador de cantidad de desplazamiento 1015aFJ recibe el exponente del número FP pre-procesado y genera la cantidad de desplazamiento. La salida del desplazador a la derecha 1010aFJ son los n+m+1 MSBs del número en coma fija pre-procesado. El LSB es, similarmente, igual a 1 y no es ni generado ni representado. En una implementación alternativa el inversor de bit condicional podría estar detrás del desplazador a la derecha.

La Fig. 16c ilustra un conversor sin sesgo para la conversión de un número FP pre-procesado, que tiene n+1 bits de mantisa y un exponente de d bits, a un número en coma fija pre-procesado de n+m+2 bits. Los n bits más significativos de la mantisa se introducen en el inversor de bits condicional 1005bFJ. El LSB de la mantisa es igual a 1 y no se introduce. El signo del número FP pre-procesado se utiliza para controlar el inversor de bits condicional 1005bFJ. La salida del inversor de bits condicional 1005bFJ, junto con el signo (sign_x) y expandida, es introducida al desplazador a la derecha 1010bFJ. La salida del inversor de bits condicional se expandió mediante la adición por la derecha de un bit seleccionado al azar y tantos bits con el valor inverso de dicho bit al azar, según sean necesarios para completar el nuevo tamaño. El bit aleatorio podría ser cualquiera de la mantisa inicial, u otro con las adecuadas características estadísticas.. El desplazador a la derecha 1010bFJ tiene otra entrada para recibir la cantidad de desplazamiento del calculador de cantidad de desplazamiento 1015bFJ. El calculador de cantidad de desplazamiento 1015bFJ recibe el exponente del número FP pre-procesado y genera la cantidad de desplazamiento. La salida del desplazador a la derecha 1010bFJ son los n+m+1 MSBs del número en coma fija pre-procesado. El LSB es, similarmente, igual a 1 y no es ni generada ni representado. En una implementación alternativa el inversor de bit condicional podría

estar detrás del desplazador a la derecha.

Otra categoría de tales conversores es la de conversores para convertir números en coma fija pre-procesados a números FP pre-procesados. La Fig. 17 ilustra un ejemplo de tal conversor para un número en coma fija pre-procesado de m+2 bits y un número FP pre-procesado con una mantisa de n+1 bits. El conversor 600FJ comprende un módulo de normalización 630FJ que tiene un inversor de bits condicional 605FJ en serie con un desplazador a la izquierda pre-procesado 610FJ, el cual podría ser similar al descrito con referencia a la Fig. 12. El inversor de bits condicional tiene una primera entrada para recibir los m LSBs de los m+1 MSBs de un número en coma fija pre-procesado de m+2 bits. El MSB del número de m+2 bits es el signo, y será el signo del número FP pre-procesado, así como será usado para controlar el inversor de bits condicional 605FJ. La salida de m bits del inversor de bits condicional 605FJ es la entrada del desplazador a la izquierda pre-procesado 610FJ. En implementaciones alternativas el desplazador a la izquierda pre-procesado precede al inversor de bits condicional 605FJ. La función del desplazador a la izquierda pre-procesado 610FJ es normalizar el número de entrada, desplazándolo de acuerdo a la cantidad de desplazamiento recibida, y redondearlo sin sesgo. Una implementación de dicho desplazador a la izquierda pre-procesado es descrita con más detalle con referencia a la Fig. 12.

En este ejemplo de la Fig. 17, la máxima cantidad de desplazamiento es m+1. Si el número en coma fija es igual a cero y el bit aleatorio (R) en la Fig. 12 es también igual a cero, se requiere una máxima cantidad de desplazamiento que tiene un bit adicional (m+1) de manera que la mantisa pueda ser normalizada. Alternativamente, si cuando el número en coma fija es igual a cero, es tratado como un caso especial y convertido a cero en FP, entonces la máxima cantidad de desplazamiento podría ser igual a m.

El valor de entrada del desplazador a la izquierda pre-procesado 610FJ es aumentado con un LSB adicional, fijado a cualquier bit con un valor aleatorio (por ejemplo, el LSB del valor de entrada inicial) y ambas, las posiciones vacantes requeridas para completar el tamaño n, si n>m+1, y las posiciones vacantes producidas después del desplazamiento, se fijan al inverso de dicho bit aleatorio. La salida del desplazador a la

izquierda pre-procesado 610FJ comprende los n MSBs de la mantisa Mz del número FP pre-procesado. Dicha salida se corresponde sólo con los n MSBs del valor desplazado si n<m. El LSB de la mantisa Mz está implícito y es igual a 1.

En un camino paralelo, el conversor 600FJ comprende el módulo detector de uno de cabecera (LOD 615FJ), que tiene una entrada conectada a la salida del inversor de bits condicional 605FJ y una salida para la generación de la cantidad de desplazamiento del desplazador a la izquierda pre-procesado 610FJ que también se utiliza como entrada al módulo de cálculo de exponentes 620FJ para generar el exponente Ez del número FP pre-procesado. Alternativamente, la entrada del módulo LOD 615FJ podría estar conectada directamente a la entrada del conversor 600FJ, pero en este caso debería detectar el primer cero, en lugar del uno, cuando el número es negativo.

En comparación con los conversores convencionales de en coma fija a FP, cuando M>N, no hay redondeo hacia arriba después de la operación de desplazamiento y por lo tanto hay una reducción en los componentes y en el procesamiento. Cuando M<N, entonces no hay sesgo producido por el redondeo con la utilización del conversor propuesto.

Otra categoría de tales conversores es la de conversores para convertir números en coma fija pre-procesados a números FP no procesados. La Fig. 18 ilustra un ejemplo de tal conversor para números en coma fija pre-procesados de m+2 bits y un número FP no procesado con una mantisa de n bits. El conversor 1500FJ tiene una entrada para recibir los m+1 MSBs de un número en coma fija pre-procesado. El conversor 1500FJ comprende un módulo de normalización 1530FJ, que tiene un inversor de bits condicional 1505FJ en serie con un desplazador a la izquierda 1510FJ, y un módulo de redondeo 1540FJ. El inversor de bits condicional 1505FJ tiene una primera entrada para recibir los m LSBs de dicha entrada de m+1 bits. El MSB del número en coma fija pre-procesado es su signo y será el signo del número FP no procesado, y también será usado para controlar el inversor de bits condicional 1505FJ. La salida de m bits del inversor de bits condicional 1505FJ es la entrada al desplazador a la izquierda 1510FJ. El valor 1 se inserta también en la entrada del desplazador a la izquierda 1510FJ de forma que los m bits de la salida del inversor de bits condicional 1505FJ son

aumentados con un bit a la derecha correspondiéndose con el LSB implícito. Sin embargo, en otras implementaciones la introducción del uno adicional podría realizarse internamente en el desplazador a la izquierda 1510FJ sin la necesidad de una entrada especial. El desplazador a la izquierda 1510FJ produce una salida de n+1 bits correspondiente a la mantisa Mz del número FP no procesado antes del redondeo. Dicha salida se corresponde sólo con los n+1 MSBs del valor desplazado si n<m. Ambas, las posiciones vacantes para completar el tamaño n si n>m y las posiciones vacantes producidas después del desplazamiento se fijan a cero. La salida de n+1 bit del módulo de normalización 1530FJ se redondea a n bits, por el módulo de redondeo 1540FJ. El módulo de redondeo 1540FJ también genera una salida de desbordamiento que es usada por el calculador de exponente 1520FJ, para generar el exponente del número FP no procesado. El redondeador 1540FJ es similar al redondeador 100bCFJ explicado en la Fig. 15a. Un sumador se usa para incrementar en uno los n MSBs de la salida del módulo de normalización 1530FJ, si el LSB de dicha salida es uno. En una implementación alternativa diferentes unidades de redondeo, realizando diferentes modos de redondeo podrían ser usados. En otras implementaciones, el MSB de la mantisa normalizada Mz podría no incluir el uno de cabecera. Por lo tanto, la salida del desplazador podría tener un bit menos.

En un camino paralelo, el conversor 1500FJ comprende el módulo LOD 1515FJ, que tiene una entrada conectada a la salida del inversor de bits condicional 1505FJ y una salida para la generación de la cantidad de desplazamiento del desplazador a la izquierda 1510FJ, que también se utiliza, junto con la señal de desbordamiento, como entrada al módulo de cálculo de exponentes 1520FJ, para generar el exponente Ez del número FP no procesado. Alternativamente, la entrada del módulo LOD 1515FJ podría estar conectada directamente a la entrada del conversor 1500FJ. El conversor mostrado en este ejemplo podría producir cierto sesgo, cuando n<m y el número de entrada es tal que el LSB de la salida del desplazador a la izquierda 1510FJ coincide con el LSB de dicho número de entrada. Este sesgo podría ser evitado aplicando técnicas clásicas, cuando esta situación ocurre, tal como solo realizar el redondeo por exceso si el segundo LSB del número es también uno. En algunas implementaciones, dicha situación podría ser detectada mediante la comprobación de la cantidad de desplazamiento, mientras que en otras, podría detectarse calculando el bit de sticky

sobre los m-n LSBs del valor desplazado.

Otra categoría de conversores son los conversores para convertir números FP no procesados a números en coma fija pre-procesados. La Fig. 19 ilustra un conversor 1600FJ para la conversión de un número FP, que tiene una mantisa de m bits y un exponente de d bits, en un número coma fija pre-procesado de n+2 bits. La mantisa de m bits es introducida en un conversor en coma fija de números no procesado a procesados 1602FJ, similar a los descritos en Fig. 13a a 13c, de acuerdo a la relación entre n y m, configurado para generar los n MSBs de un número en coma fija pre-procesado de n+1 bit. En una implementación alternativa, como dicha mantisa está normalizada, su MSB podría estar implícito, y dicho MSB podría no ser introducido explícitamente en el conversor. Dichos n MSBs del número pre-procesado son la entrada al inversor de bits condicional 1605FJ, mientras que el LSB está implícito y es igual a uno. El signo del número FP no procesado se utiliza para controlar el inversor de bits condicional 1605FJ. La salida del inversor de bits condicional 1605FJ, junto con el signo (sign_x), se introducen en desplazador a la derecha 1610FJ. El desplazador a la derecha 1610FJ tiene otra entrada para recibir la cantidad de desplazamiento del calculador de cantidad de desplazamiento 1615FJ. El calculador de cantidad de desplazamiento 1615FJ recibe el exponente del número FP no procesado y genera la cantidad de desplazamiento. La salida del desplazador a la derecha 1610FJ se corresponden con los n+1 MSBs del número en coma fija pre-procesado. El LSB es, de manera similar, igual a 1 y no es ni generado ni representado. En una implementación alternativa, el inversor de bit condicional podría estar situado después del desplazador a la derecha.

A pesar de que se han descrito aquí sólo algunas realizaciones y ejemplos particulares de la invención, el experto en la materia comprenderá que son posibles otras realizaciones alternativas y/o usos de la invención, así como modificaciones obvias y elementos equivalentes. Además, la presente invención abarca todas las posibles combinaciones de las realizaciones concretas que se han descrito. El alcance de la presente invención no debe limitarse a realizaciones concretas, sino que debe ser determinado únicamente por una lectura apropiada de las reivindicaciones adjuntas.

Por otro lado, las realizaciones descritas de la invención con referencia a los dibujos comprenden sistemas informáticos y procesos realizados en sistemas informáticos, caracterizados a nivel funcional, e independientes del soporte o tecnología empleada para su implementación. Este medio de soporte podría ser, por ejemplo, un circuito integrado para aplicaciones específicas (ASIC, siglas en inglés), un circuito lógico programable (FPGA o CPLD, siglas en inglés) que incluyen una memoria, o cualquier otro dispositivo, estando dichos circuitos adaptados o configurados para realizar, o para usarse en la realización de, los procesos relevantes.

A pesar también de que las realizaciones descritas comprenden dispositivos informáticos, la invención también se extiende a programas informáticos, más particularmente a programas informáticos en unos medios portadores, adaptados para llevar a cabo la invención. El programa informático puede estar en forma de código fuente, código objeto o un código intermedio entre código fuente y código objeto, tal como en una forma parcialmente compilada, o en cualquier otra forma adecuada para su uso en la implementación de los procesos de acuerdo con la invención. El medio portador puede ser cualquier entidad o dispositivo capaz de portar el programa.

Por ejemplo, el medio portador puede comprender un medio de almacenamiento, tal como una ROM, por ejemplo un CD ROM o una ROM semiconductora, o un medio de grabación magnético, por ejemplo un floppy disc o un disco duro. Además, el medio portador puede ser un medio portador transmisible tal como una señal eléctrica u óptica que puede transmitirse vía cable eléctrico u óptico o mediante radio u otros medios.

Cuando el programa informático está contenido en una señal que puede transmitirse directamente mediante un cable u otro dispositivo o medio, el medio portador puede estar constituido por dicho cable u otro dispositivo o medio.

REIVINDICACIONES

1. Un dispositivo para realizar una operación deseada de al menos un primer número en coma fija pre-procesado con N+1 dígitos para generar al menos un segundo número en coma fija pre-procesado con Z+1 dígitos, el dispositivo comprende:

al menos una unidad aritmética con una primera entrada para recibir los N MSDs de dicho al menos primer número en coma fija pre-procesado,

donde dicha al menos una unidad aritmética está configurada para generar los Z MSDs de el al menos segundo número en coma fija pre-procesado, mientras que el Digito Menos Significativo (LSD) de todos los números en coma fija pre-procesados es igual a B/2, siendo B la base del sistema numérico.

2. Dispositivo según reivindicación 1, en el que B=2 y los dígitos son bits.

3. Dispositivo según reivindicación 1 ó 2, en el que la al menos una unidad aritmética comprende además:

al menos una segunda entrada para recibir los L MSDs de un tercer número en coma fija pre-procesado con L+1 dígitos, y en el que L≥ N y el LSD es igual a B/2, y

un módulo de suma para generar un valor, correspondiente al segundo número en coma fija pre-procesado, dicho segundo número en coma fija pre-procesado siendo el resultado, redondeado al más cercano, de la suma del primer y el tercer número en coma fija pre-procesado.

4. Dispositivo según reivindicación 3, en el que el módulo de suma comprende un sumador configurado para recibir los N MSBs del primer y tercer número en coma fija pre-procesado, en una primera y segunda entrada, respectivamente.

5. Dispositivo según reivindicación 4, en el que, cuando Z≤N, dicho sumador está configurado para generar los Z MSBs del valor equivalente a sumar dichas dos entradas más un acarreo de entrada igual al (N+1)-ésimo bit del tercer número en coma fija pre-procesado.

6. Dispositivo según reivindicación 4, en el que, cuando Z>N, dicho sumador está configurado para generar los N MSBs del segundo número en coma fija pre-procesado produciendo un valor equivalente a sumar dichas dos entradas más un acarreo de entrada igual al (N+1)-ésimo bit del tercer número en coma fija pre-procesado,

en el que el módulo de suma está configurado además para fijar el (N+1)-ésimo bit del segundo número en coma fija pre-procesado igual al inverso del (N+1)-ésimo bit del tercer número en coma fija pre-procesado y configurado para fijar los restantes Z-N-1 LSBs de los Z MSBs del segundo número en coma fija pre-procesado igual a los Z-N-1 LSBs de los Z MSBs del tercer número en coma fija pre-procesado.

7. Dispositivo según reivindicación 5, en el que, cuando Z=N=L, el módulo de suma está configurado además para fijar a cero el segundo LSB del segundo número en coma fija pre-procesado.

8. Dispositivo según cualquiera de las reivindicaciones 3 a 7, en el que el módulo de suma está configurado además para negar uno de los números de entrada.

9. Dispositivo según reivindicación 8, en el que dicha operación de negación se realiza selectivamente de acuerdo a una señal de control.

10. Dispositivo según cualquiera de las reivindicaciones 1 a 9, en el que la al menos una unidad aritmética comprende un módulo de multiplicación para generar un valor correspondiente al segundo número en coma fija pre-procesado.

11. Dispositivo según reivindicación 10, en el que el módulo de multiplicación es un elevador al cuadrado.

12. Dispositivo según reivindicación 11, en el que cuando el primer número en coma fija pre-procesado es con signo, el elevador al cuadrado comprende un módulo configurado para generar los N+1 MSBs de la magnitud del primer número en coma fija pre-procesado.

13. Dispositivo según reivindicación 10, en el que el módulo de multiplicación está configurado para generar dicho valor, correspondiente al segundo número en coma fija pre-procesado, el cual es el resultado, redondeado al más cercano, de la multiplicación

del primer y un cuarto número en coma fija pre-procesado de T+1 dígitos, teniendo el LSD igual a B/2.

14. Dispositivo según reivindicación 13, en el que cuando el cuarto número en coma fija pre-procesado es un número constante, el módulo de multiplicación es un multiplicador por constante.

15. Dispositivo según reivindicación 13, en el que la al menos una unidad aritmética comprende al menos una segunda entrada para recibir los T MSDs del cuarto número en coma fija pre-procesado.

16. Dispositivo según cualquiera de las reivindicaciones 13 a 15, en el que el módulo de multiplicación comprende

un multiplicador configurado para generar los N+T+1 MSBs del resultado de la multiplicación, y

un módulo de truncado, conectado a la salida del multiplicador para recibir la salida del multiplicador y generar los Z MSBs del segundo número truncando dicha salida.

17. Dispositivo según cual quiera de las reivindicaciones 13 a 15, en el que el módulo de multiplicación comprende

un módulo multiplicación redundante configurado para recibir, en una primera entrada, los N MSBs del primer número en coma fija pre-procesado y generar, en un formato de representación redundante, como mucho los N+T+1 MSDs del valor correspondiente a la operación de multiplicación entre dicho primer número pre-procesado y el cuarto número en coma fija pre-procesado;

un módulo de conversión, conectado a la salida de dicho módulo de multiplicación, configurado para recibir los Z MSDs de la salida de dicho multiplicador redundante y un bit de acarreo, y generar una salida de Z bits correspondiente a la conversión del valor redundante recibido a formato de representación no redundante;

un módulo de red de acarreo configurado para recibir los N+T+1-Z LSDs de la salida de dicho módulo multiplicador redundante y generar dicho bit de acarreo

correspondiente al acarreo de salida de la conversión de los N+T+1-Z LSDs de la salida de dicho módulo de multiplicación redundante a representación no redundante.

18. Dispositivo según reivindicación 17, en el que el módulo de multiplicación redundante comprende

un generador de productos parciales configurado para recibir, en una primera entrada, los N MSBs del primer número en coma fija pre-procesado y generar, en una salida, los productos parciales correspondientes a la multiplicación de dicha entrada y los T MSBs del cuarto número en coma fija pre-procesado,

un árbol de compresores, con una primera entrada conectada a la salida del generador de productos parciales y una segunda entrada configurada para recibir los N MSBs y los T MSBs del primer y cuarto número pre-procesado, respectivamente, dicho árbol de compresores configurado para generar, en una representación redundante, como mucho los N+T+1 MSDs de un valor correspondiente a la operación de multiplicación entre dichos números pre-procesados, en una salida.

19. Dispositivo según cualquiera de las reivindicaciones 1 a 18, en el que una unidad aritmética comprende un módulo de desplazamiento a la izquierda para generar un valor, correspondiente al segundo número en coma fija pre-procesado, el cual es el resultado, redondeado al más cercano, del desplazamiento a la izquierda del primer número en coma fija pre-procesado.

20. Dispositivo según reivindicación 19, en el que el módulo de desplazamiento a la izquierda está configurado además para completar las posiciones vacantes, debidas al desplazamiento a la izquierda, fijando el MSB de la posiciones vacantes a cero y el resto a uno, o fijando el MSB de la posiciones vacantes uno y el resto a cero.

21. Dispositivo según reivindicación 20, en el que el módulo de desplazamiento a la izquierda está configurado para, selectivamente, completar dichas posiciones vacantes, aleatoriamente, basándose en el valor de un bit seleccionado, o de una combinación de bits seleccionados.

22. Dispositivo según cual quiera de las reivindicaciones 19 a 21 en el que el módulo de desplazamiento a la izquierda está configurado para recibir la cantidad de desplazamiento para seleccionar el número de bits a desplazar

23. Dispositivo según reivindicación 22, en el que el módulo de desplazamiento a la izquierda comprende un desplazador variable configurado para recibir un bit para completar las posiciones vacantes.

24. Dispositivo según la reivindicación 23, en el que dicho desplazador variable comprende un número de sucesivos multiplexores que es igual al primer entero mayor o igual que el logaritmo en base 2 de la máxima cantidad de desplazamiento [log2(máxima cantidad de desplazamiento)], con cada multiplexor configurado para efectuar una operación de desplazamiento a la izquierda de 2^i posiciones, iϵ[0, número de multiplexores-1], y cada multiplexor configurado para completar las posiciones vacantes usando el valor de dicho bit recibido.

25. Dispositivo según cualquiera de las reivindicaciones 1 a 24, en el que al menos una unidad aritmética comprende un módulo de valor absoluto para generar un valor correspondiente al segundo número en coma fija pre-procesado, el cual es el resultado del valor absoluto del primer número en coma fija pre-procesado, en el que dicho módulo comprende

un inversor de bit condicional configurado para recibir, en una primera entrada los N MSBs del primer número en coma fija pre-procesado, y generar un valor correspondiente al complemento a uno de la primera entrada si su MSB es uno.

26. Dispositivo según cualquiera de las reivindicaciones anteriores, que comprende además un conversor de números no procesados a números pre-procesados en coma fija conectado a una entrada de la unidad aritmética y configurado para recibir un número en coma fija no procesado de E+1 bits y generar un número en coma fija pre-procesado.

27. Dispositivo según la reivindicación 26, en el que cuando el número en coma fija pre-procesado tiene E+1-K1 bits, K1<E entonces el conversor comprende:

una unidad de redondeo configurada para eliminar los K1+1 LSBs del número en coma fija no procesado, para generar los E-K1 MSBs del número coma fija pre-procesado, donde el LSB de dicho número en coma fija pre-procesado es igual a B/2.

28. Dispositivo según la reivindicación 27, en el que la unidad de redondeo está configurada además para, selectivamente, poner a cero el segundo LSB del número en coma fija pre-procesado si todos los K1+1 LSBs del número en coma fija no procesado son iguales a cero.

29. Dispositivo según la reivindicación 26, en el que cuando el número en coma fija pre-procesado tiene E+1+K2 bits entonces el conversor comprende:

un módulo de rellenado, configurado para recibir el número en coma fija no procesado y generar los E+K2 MSBs del número en coma fija pre-procesado fijando los E+1 MSBs del número en coma fija pre-procesado al mismo valor que los E+1 bits del número en coma fija no procesado y los restantes bits a cero, donde el LSB del número en coma fija pre-procesado es igual a uno

30 Dispositivo según la reivindicación 29, en el que el módulo de rellenado está configurado además para generar selectivamente el valor correspondiente a restar uno del segundo LSB del mencionado número en coma fija pre-procesado cuando un bit seleccionado, o una combinación de bit seleccionados, del número no procesado de entrada es igual a uno.

31. Dispositivo según cualquiera de las reivindicaciones anteriores, que comprende además un conversor de números coma fija pre-procesados a números coma fija pre-procesados conectado a una entrada y/o una salida de la unidad aritmética, y configurado para recibir un número inicial coma fija pre-procesado de J+1 bits y generar un subsecuente número coma fija pre-procesado de diferente tamaño.

32. Dispositivo según la reivindicación 31, en el que cuando el subsecuente número en coma fija pre-procesado tiene J+1-P1 bits, P1<J, entonces el conversor comprende:

una unidad de redondeo para eliminar los P1+1 LSBs de los J+1 bits del número inicial pre-procesado, para generar los J-P1 MSBs del subsecuente número en coma

fija pre-procesado, donde el LSB del subsecuente número coma fija pre-procesado es igual a B/2.

33. Dispositivo según la reivindicación 31, en el que cuando el subsecuente número en coma fija pre-procesado tiene J+1+P2 bits, entonces el conversor comprende:

un módulo de rellenado, configurado para recibir los J MSBs del número en coma fija pre-procesado inicial y generar los J+P2 MSBs del subsecuente número en coma fija pre-procesado fijando el MSB de los P2 LSBs a uno o a cero, y los restante P2-1 bits de dichos P2 LSBs al inverso del mencionado MSB, mientras los J MSBs del subsecuente número en coma fija pre-procesado son los mismos que los J MSBs del número en coma fija pre-procesado inicial.

34. Dispositivo según la reivindicación 33, en el que el módulo de rellenado está configurado para fijar aleatoriamente dicho MSB basándose en el valor de un bit seleccionado, o de una combinación de bits seleccionados.

35. Dispositivo según cualquiera de las reivindicaciones anteriores, comprende además un conversor de números coma fija pre-procesados a números coma fija no procesados conectado a la salida de la unidad aritmética y configurado para recibir un número en coma fija pre-procesado de W+1 bits y generar un número en coma fija no procesado.

36. Dispositivo según la reivindicación 35, en el que cuando el número en coma fija no procesado tiene W+1-V1 bits, V1<W, entonces el conversor comprende:

un módulo de redondeo, configurado para recibir los W+2-V1 MSBs del número en coma fija pre-procesado y generar los W+1-V1 bits del número en coma fija no procesado.

37. Dispositivo según la reivindicación 36, en el que el módulo de redondeo comprende un sumador; dicho sumador está configurado para recibir, en una entrada, los W+1-V1 MSBs del número en coma fija pre-procesado e incrementar dicho valor de entrada si el (W+2-V1)-ésimo MSB de dicho número pre-procesado es igual a 1

38. Dispositivo según la reivindicación 35, en el que cuando el número en coma fija no procesado tiene W+1+V2 bits entonces el conversor comprende:

un módulo de rellenado, configurado para recibir los W MSBs del número en coma fija pre-procesado y generar los W+V2+1 bits del número en coma fija no procesado poniendo el MSB de los V2+1 LSBs a uno y los restantes bits a cero

39. Dispositivo según cualquiera de las reivindicaciones anteriores, en el que comprende además un conversor de números coma flotante pre-procesados a números coma fija pre-procesados conectado a una entrada de la unidad aritmética configurado para convertir un número en coma flotante pre-procesado con una mantisa de F+2 bits en un número en coma fija.

40. Dispositivo según la reivindicación 39, en el que el número en coma fija pre-procesado tiene G bits, con G<F+4, el conversor comprende:

un calculador de la cantidad de desplazamiento que recibe el exponente del número en coma flotante pre-procesado en una entrada y que genera una cantidad de desplazamiento en una salida,

un segundo módulo de desplazamiento con una primera entrada para recibir los G-1 MSBs de la mantisa del número en coma flotante pre-procesado y una segunda entrada acoplada a la salida del calculador de cantidad de desplazamiento y una tercera entrada para recibir el signo del mencionado número en coma flotante, para generar los G-1 MSBs del número en coma fija pre-procesado en una salida.

41. Dispositivo según la reivindicación 40, en el que el módulo de desplazamiento comprende un desplazador aritmético a la derecha acoplado a un inversor de bits condicional.

42. Dispositivo según reivindicación 39, en el que cuando el número en coma fija pre-procesado comprende F+C +3 bits, C >0, el conversor comprende:

un calculador de cantidad de desplazamiento que recibe el exponente del número pre-procesado en una entrada y que genera una cantidad de desplazamiento en una salida,

un módulo de desplazamiento aritmético a la derecha con una primera entrada conectada a la salida del calculador de desplazamiento, configurado para generar los F+C+2 MSBs del número en coma fija pre-procesado mediante el desplazamiento aritmético a la derecha de un valor intermedio de F+C +2 bits formado, de izquierda a derecha, por el bit de signo, los F+1 MSBs de la mantisa del número en coma flotante pre-procesado, y el MSB de los C LSBs puesto a cero y el resto a uno, o el MSB de los C LSBs puesto a uno y el resto a cero.

43. Dispositivo según la reivindicación 42, en el que el módulo de desplazamiento aritmético a la derecha está configurado para poner aleatoriamente dicho MSB de los C LSBs del mencionado valor intermedio de F+C +2 bits en base al valor de un bit seleccionado, o de una combinación de bits seleccionados.

44. Dispositivo según las reivindicaciones 42 o 43, en el que el módulo de desplazamiento aritmético a la derecha está configurado para generar selectivamente el complemento a uno del resultado de la mencionada operación de desplazamiento.

45. Dispositivo según cualquiera de las reivindicaciones anteriores, en el que comprende además un conversor de números coma fija pre-procesados a números coma flotante pre-procesados, conectado a la salida de una unidad aritmética, configurado para convertir un número coma fija de Q+2 bits a un número coma flotante con una mantisa de M+2 bits.

46. Dispositivo según reivindicación 45 en el que dicho conversor de números coma fija pre-procesados a números coma flotante pre-procesados comprende:

un calculador de cantidad de desplazamiento,

un módulo para calcular el exponente, con una primera entrada para recibir la tercera cantidad de desplazamiento del calculador de cantidad de desplazamiento, y una salida para generar el exponente del número coma flotante pre-procesado; y

un módulo de normalización con

una primera entrada para recibir los Q MSBs de los Q+1 LSBs del número coma fija pre-procesado y una segunda para recibir la tercera cantidad de desplazamiento; dicho módulo de normalización configurado para desplazar a la

izquierda dichos Q MSBs de acuerdo con dicha cantidad de desplazamiento, completando el MSB de las posiciones vacantes con cero y el resto con unos, o el MSB con uno y el resto con ceros, para generar como mucho los M+1 MSBs de la mantisa,

mientras que el signo del número coma flotante pre-procesado corresponde al MSB del número coma fija pre-procesado.

47. Dispositivo según reivindicación 46 en el que el módulo de normalización está configurado además para, completar dichas posiciones vacantes, aleatoriamente, basándose en un bit seleccionado, o en una combinación de bits seleccionados.

48. Dispositivo según reivindicación 46 ó 47, en el que dicho módulo de normalización está configurado además para generar selectivamente el complemento a uno del resultado de dicho desplazamiento.

49. Dispositivo según cualquiera de las reivindicaciones anteriores, que comprende además un conversor de números coma fija pre-procesados a números coma flotante no procesados conectado a una salida de una unidad aritmética y configurado para convertir un número coma fija de H+2 bits a un número coma flotante con una mantisa de R+1 bits.

50. Dispositivo según reivindicación 49 en el que dicho conversor de números coma fija pre-procesados a números coma flotante no procesados comprende:

un calculador de cantidad de desplazamiento,

un módulo para calcular el exponente, con una primera entrada para recibir la cantidad de desplazamiento del calculador de cantidad de desplazamiento, y una salida para generar el exponente del número coma flotante no procesado; y

un módulo calculador de mantisa, comprendiendo:

un módulo de normalización con una primera entrada para recibir los H MSBs de los H+1 LSBs del número coma fija pre-procesado y una segunda para recibir la cantidad de desplazamiento; dicho módulo de normalización configurado para generar un valor correspondiente a como mucho los R+2 MSBs de los H+1 LSBs del número en coma fija pre-procesado desplazado a la izquierda de acuerdo con dicha cantidad de desplazamiento; and

un módulo de redondeo configurado para recibir la salida del módulo de normalización y generar como mucho los R+1 MSBs de la mantisa del número coma flotante no procesado,

mientras que su signo corresponde al MSB del número coma fija pre-procesado.

51. Dispositivo según reivindicación 50, en el que dicho módulo de normalización está configurado además para generar selectivamente la negación de dicho valor de como mucho R+2 bits.

52. Dispositivo según reivindicación 50 o 51, en el cual el módulo de redondeo comprende un sumador, dicho sumador configurado para recibir, en una entrada, los como mucho R+1 MSBs de la salida del módulo de normalización e incrementar dicha valor de entrada si el LSB de dicha salida es igual a 1.

53. Dispositivo según cualquiera de las reivindicaciones anteriores, que comprende además un conversor de números coma flotante no procesados a números coma fija pre-procesados conectado a una entrada de la unidad aritmética y configurado para convertir un número en coma flotante no procesado con una mantisa de S bits en un número en coma fija pre-procesado de A+2 bits.

54. Dispositivo según la reivindicación 53, en el que el conversor comprende:

un calculador de la cantidad de desplazamiento que recibe el exponente del número en coma flotante no procesado en una entrada y que genera una cantidad de desplazamiento en una salida,

un conversor de números en coma fija no procesados a números en coma fija pre-procesados de acuerdo a alguna de las reivindicaciones 26 a 30, configurado para recibir como mucho los S bits de la mantisa del número en coma flotante no procesado y generar los A MSBs de un número en coma fija pre-procesado; y

un módulo de desplazamiento con una primera entrada para recibir la salida de A bits de dicho conversor , una segunda entrada conectada a la salida del calculador de cantidad de desplazamiento y una tercera entrada para recibir el signo del mencionado número en coma flotante, para generar los A+1 MSBs del número en

coma fija pre-procesado de salida, desplazando a la derecha, de acuerdo a la segunda entrada, la primera entrada aumentada por la izquierda con el bit de signo, en el que el LSB de dicho número en coma fija pre-procesado es igual a B/2.

55. Dispositivo según reivindicación 54, en el que dicho módulo de desplazamiento está configurado además para generar selectivamente un valor igual al complemento a uno del resultado de dicho desplazamiento.

56. Dispositivo según la reivindicación 55, en el que dicho módulo de desplazamiento comprende un desplazador aritmético a la derecha acoplado a un inversor de bits condicional.

57. Dispositivo según cualquiera de las reivindicaciones anteriores, en el que al menos una unidad aritmética comprende además una entrada y/o salida de un dígito con el valor de B/2.

58. Dispositivo según cualquiera de las reivindicaciones anteriores, que comprendiendo

una pluralidad de unidades aritméticas; y

una entrada de selección de operación para recibir una señal sobre la operación deseada,

en el que el dispositivo está configurado para seleccionar la salida de una unidad aritmética de la pluralidad de unidades aritméticas basándose en la señal sobre la operación deseada recibida.

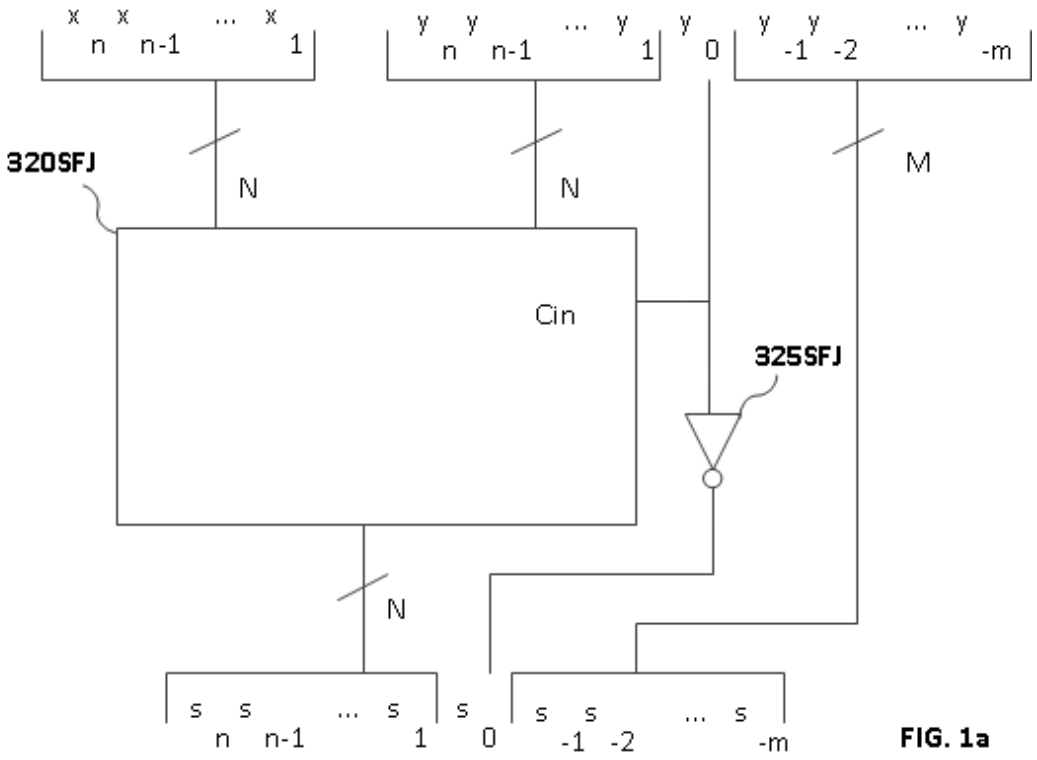
300SFJ
x x ... x
n n-1 1
y y ... y y
n n-1 1 0
y y ... y
-1 -2 -m
320SFJ
N
N
M
Cin
325SFJ
N
s s ... s s
n n-1 1 0
s s ... s
-1 -2 -m

FIG. 1a

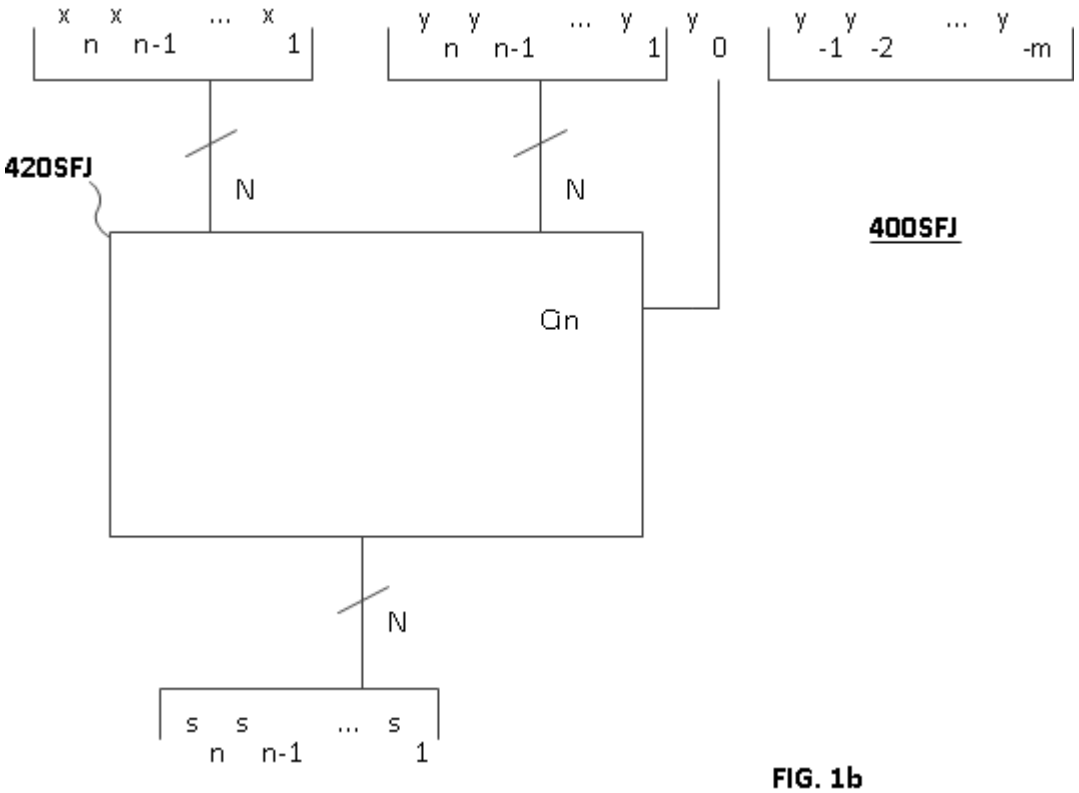

FIG. 1b

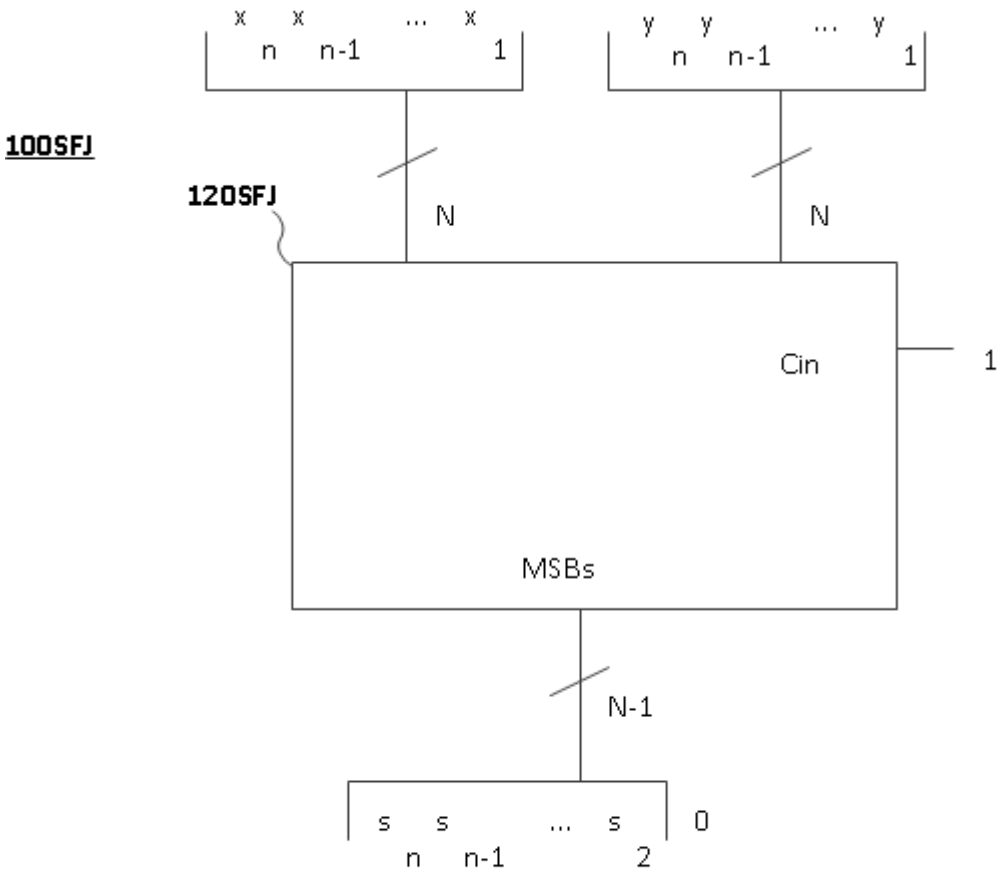

FIG. 2a

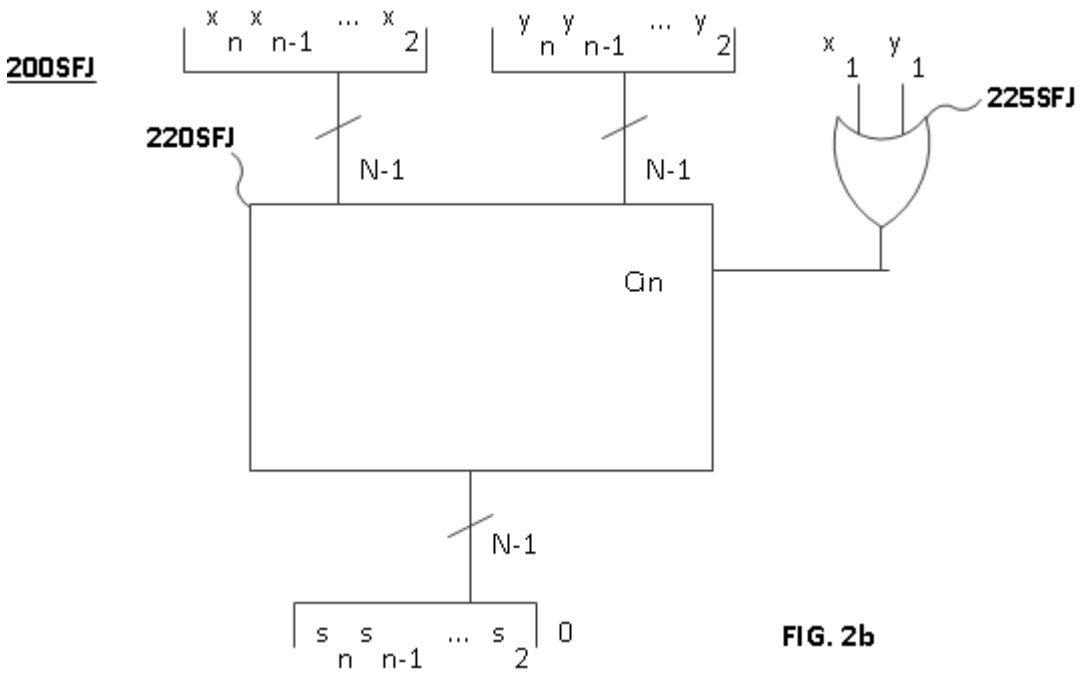
200SFJ
x x ... x
n n-1 2
y y ... y
n n-1 2
x y
1 1
225SFJ
220SFJ
N-1
N-1
Qn
N-1
s s ... s 0
n n-1 2

FIG. 2b

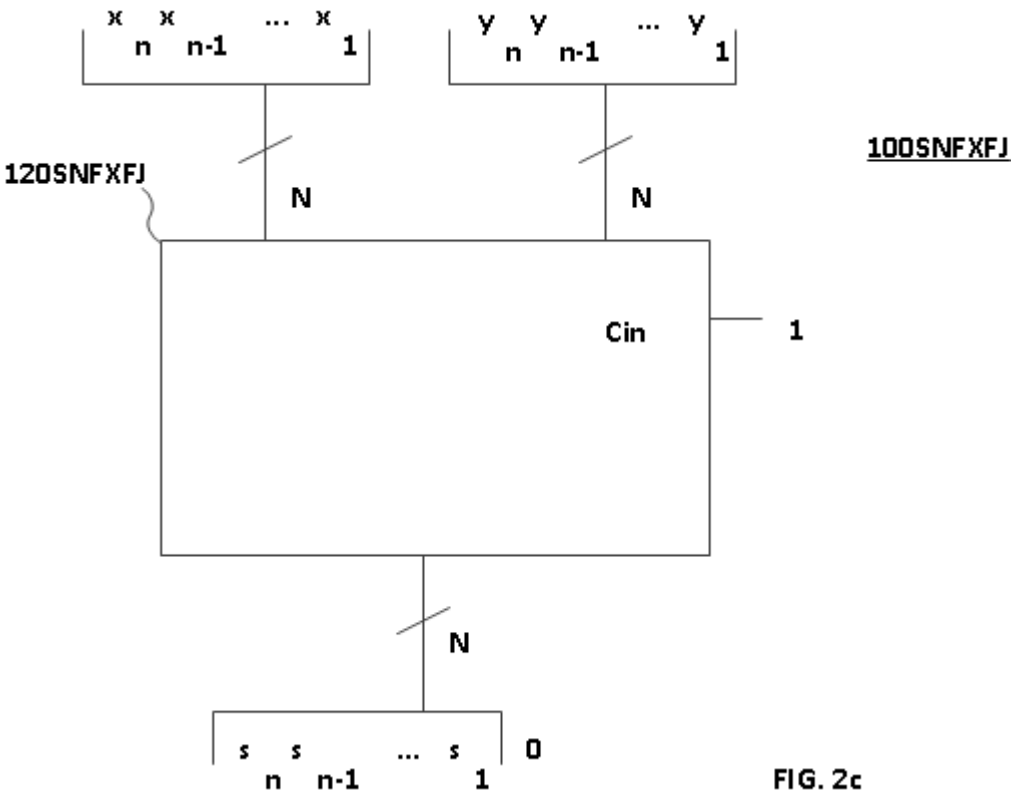

FIG. 2c

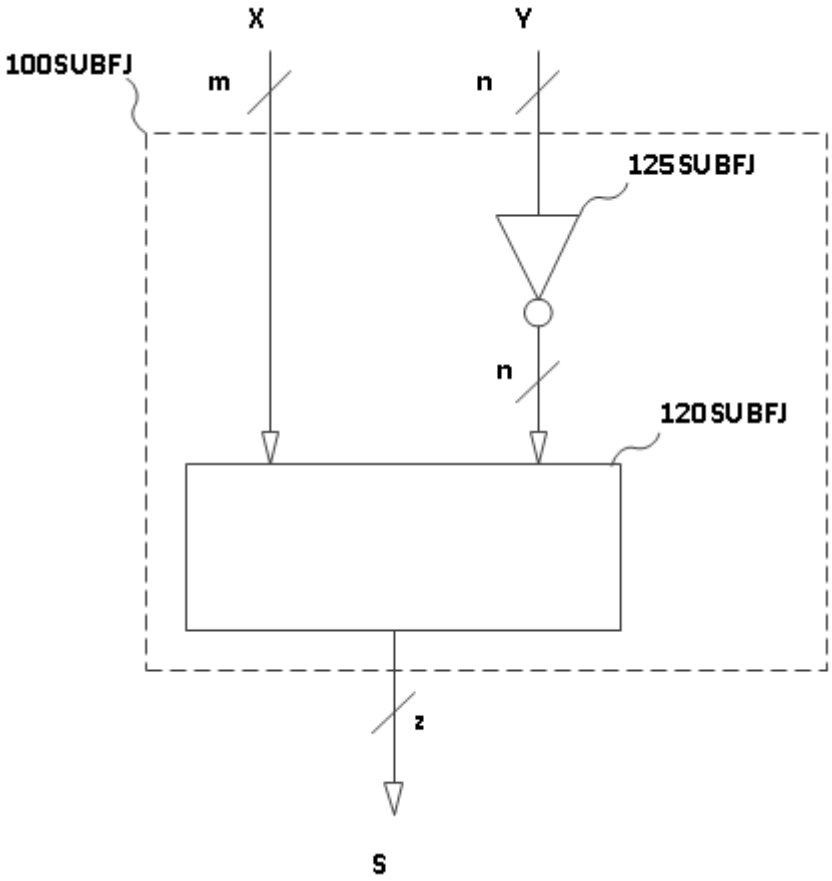

FIG. 3

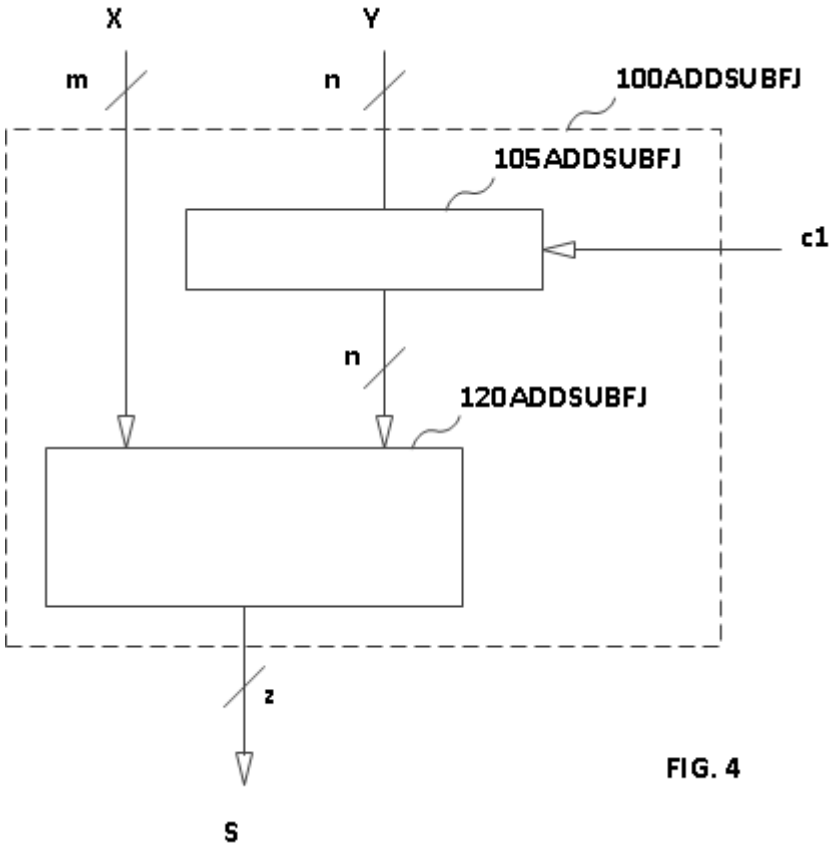

FIG. 4

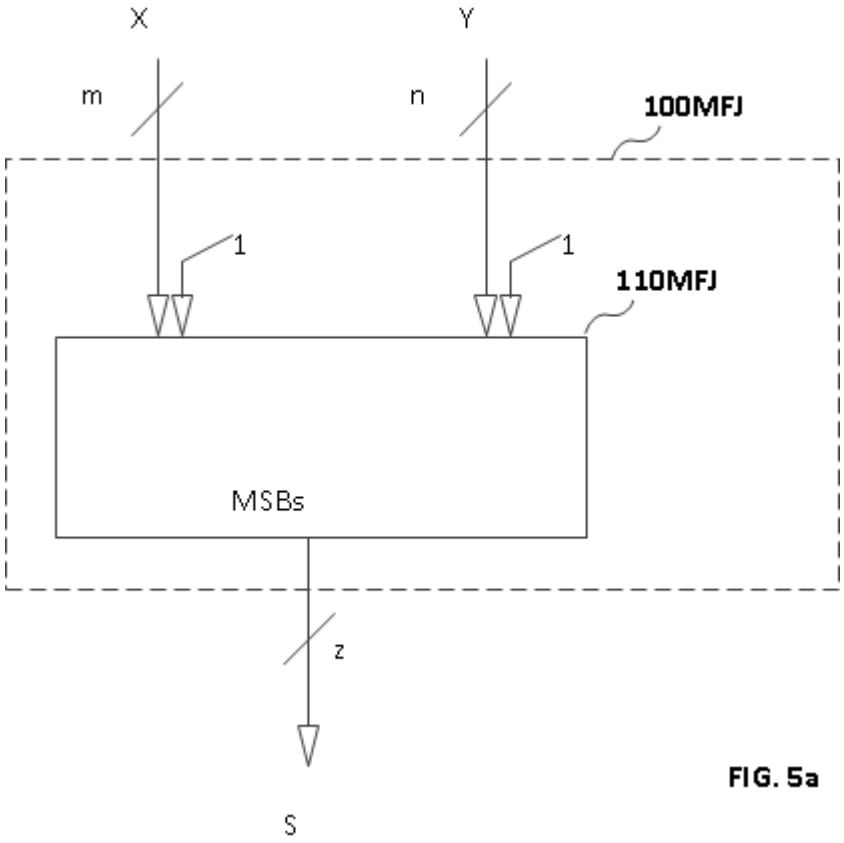

FIG. 5a

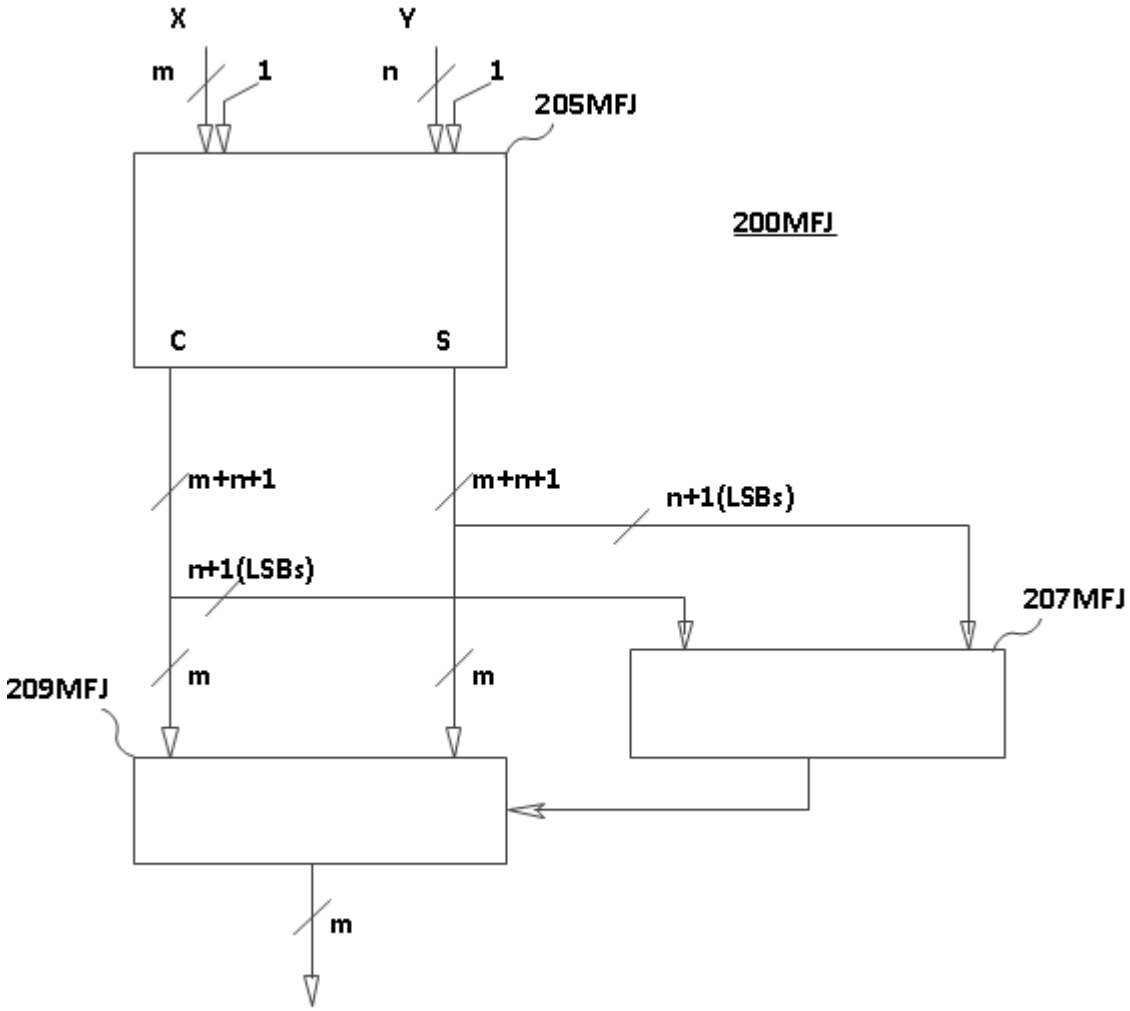
X
Y
m
1
n
1
205MFJ
200MFJ
C
S
m+n+1
m+n+1
n+1(LSBs)
m+1(LSBs)
207MFJ
209MFJ
m
m
m

FIG. 5b

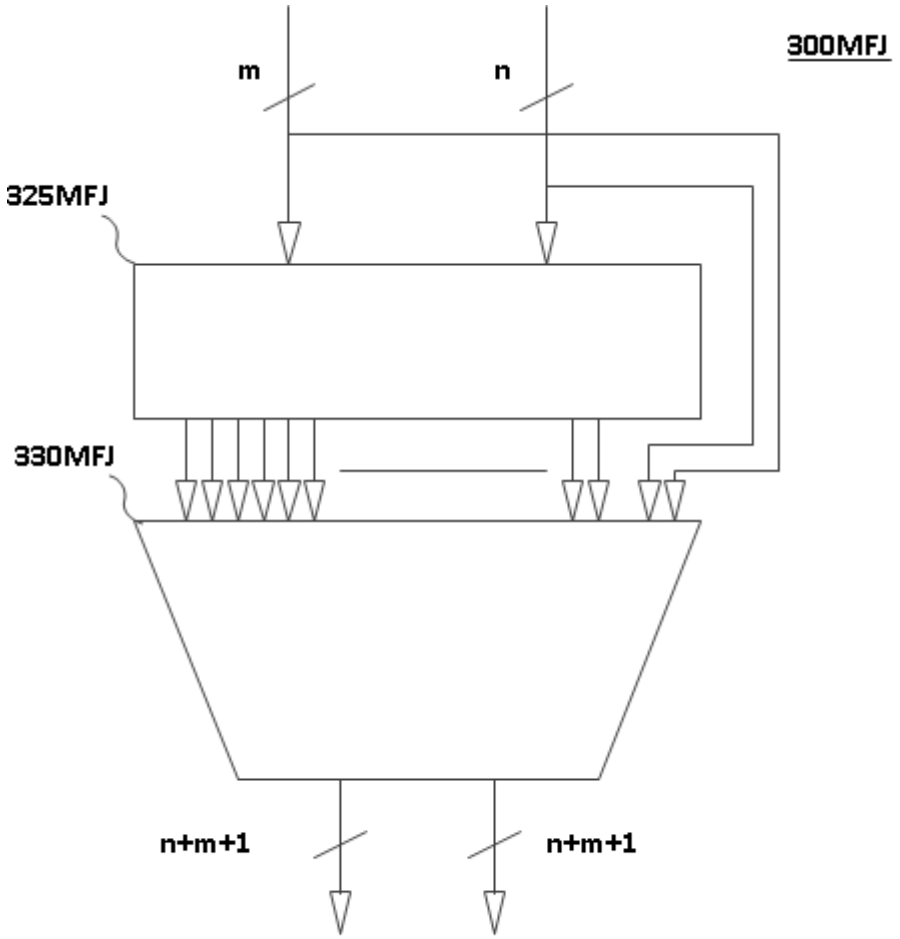

FIG. 6a

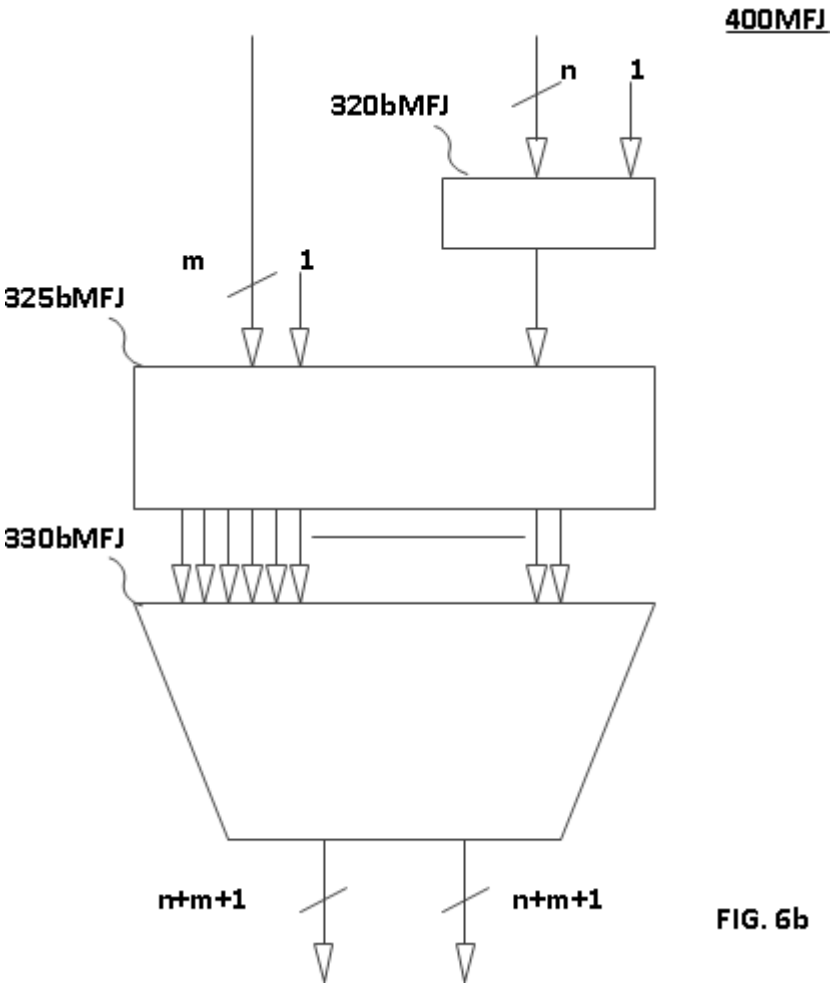

FIG. 6b

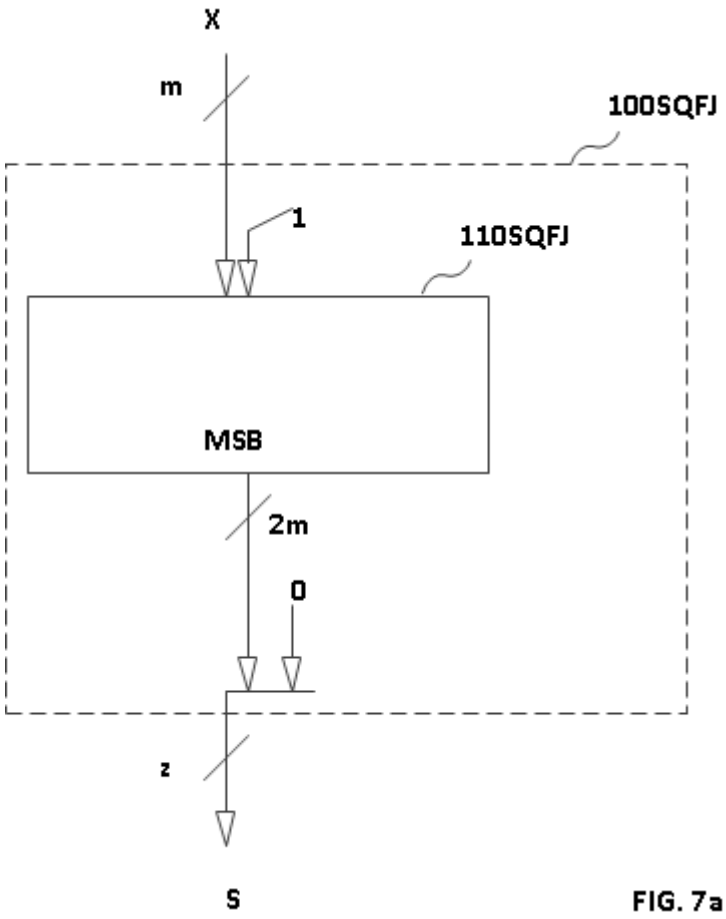

FIG. 7a

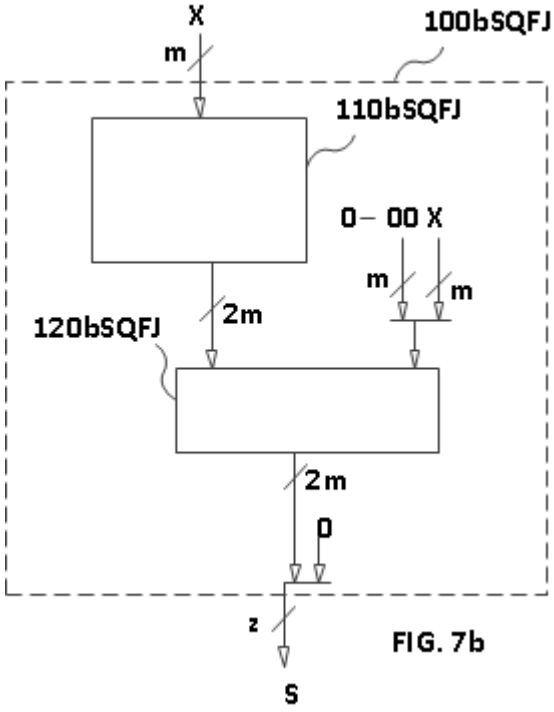

FIG. 7b

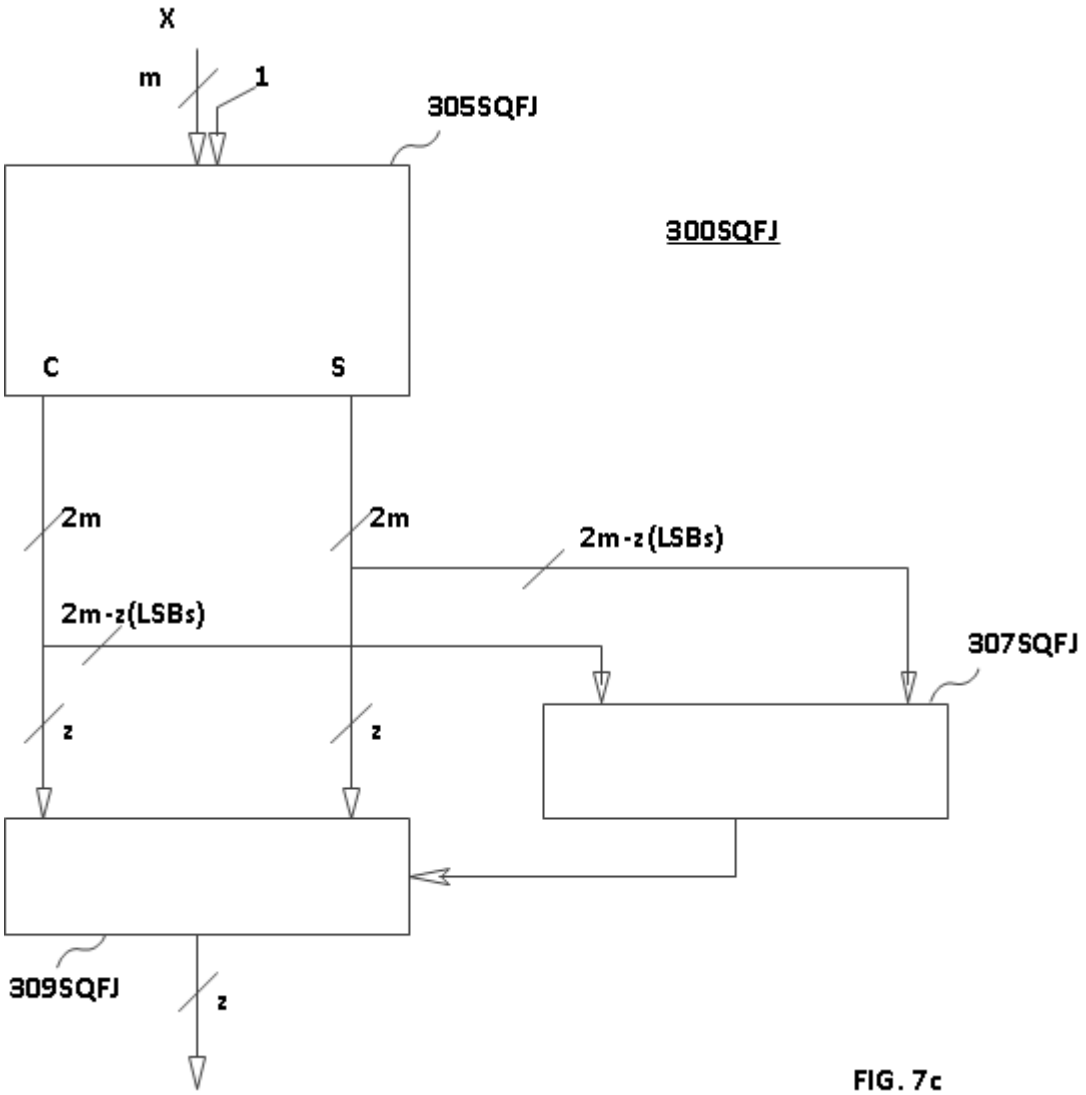
X
m
1
305SQFJ
300SQFJ
C
S
2m
2m
2m-z(LSBs)
2m-z(LSBs)
307SQFJ
z
z
309SQFJ
z

FIG. 7c

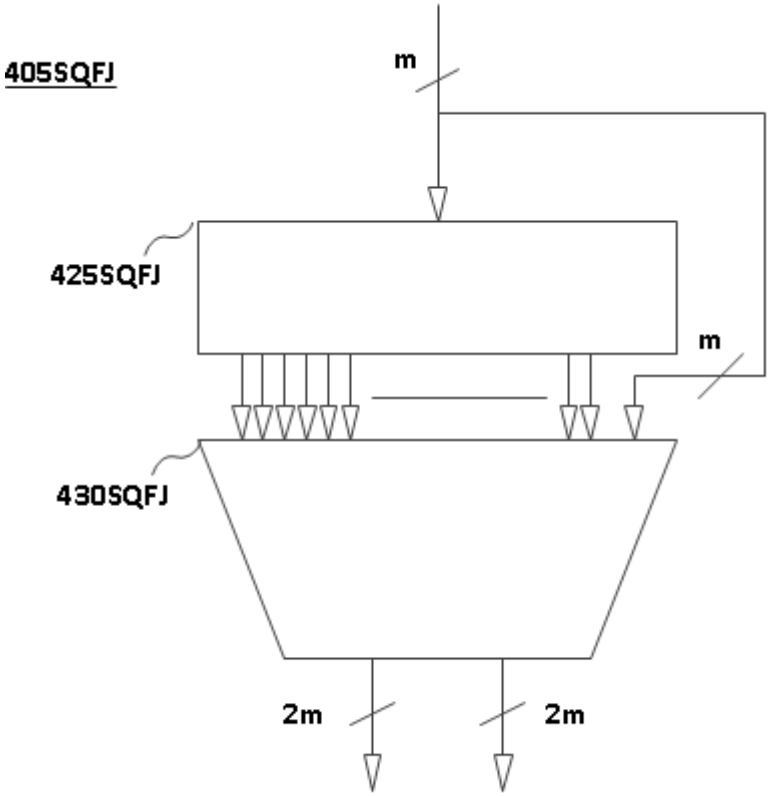

FIG. 8

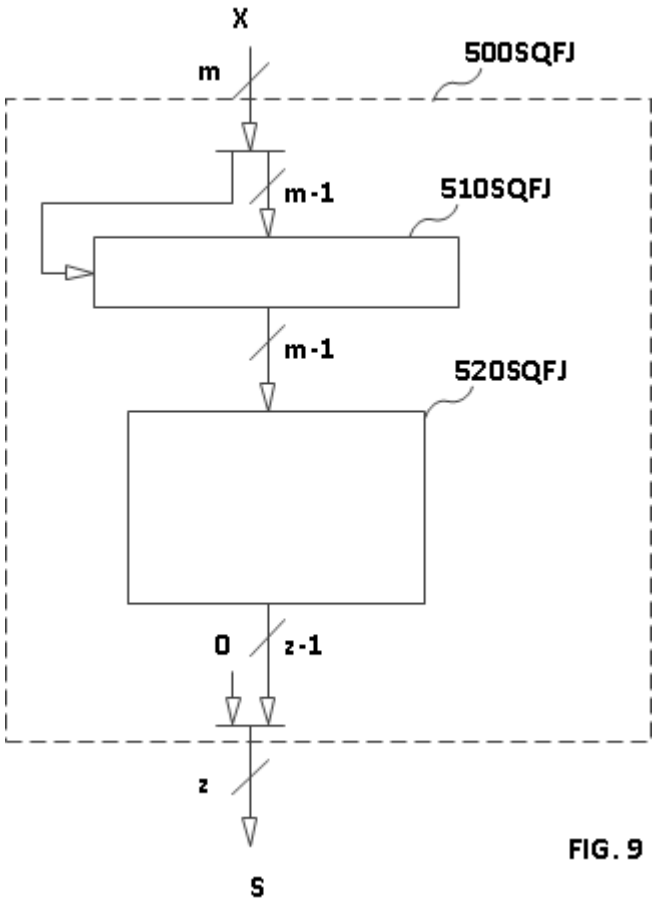

FIG. 9

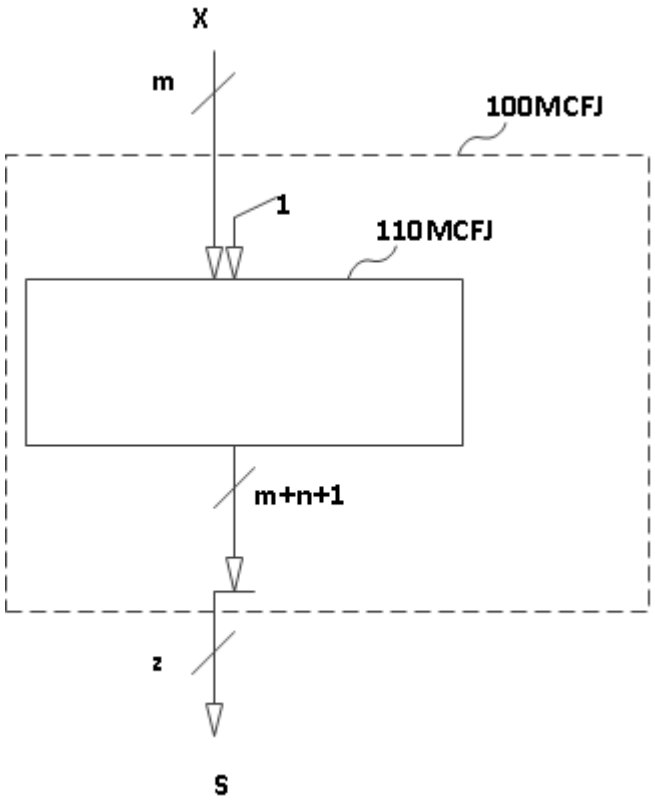

FIG. 10a

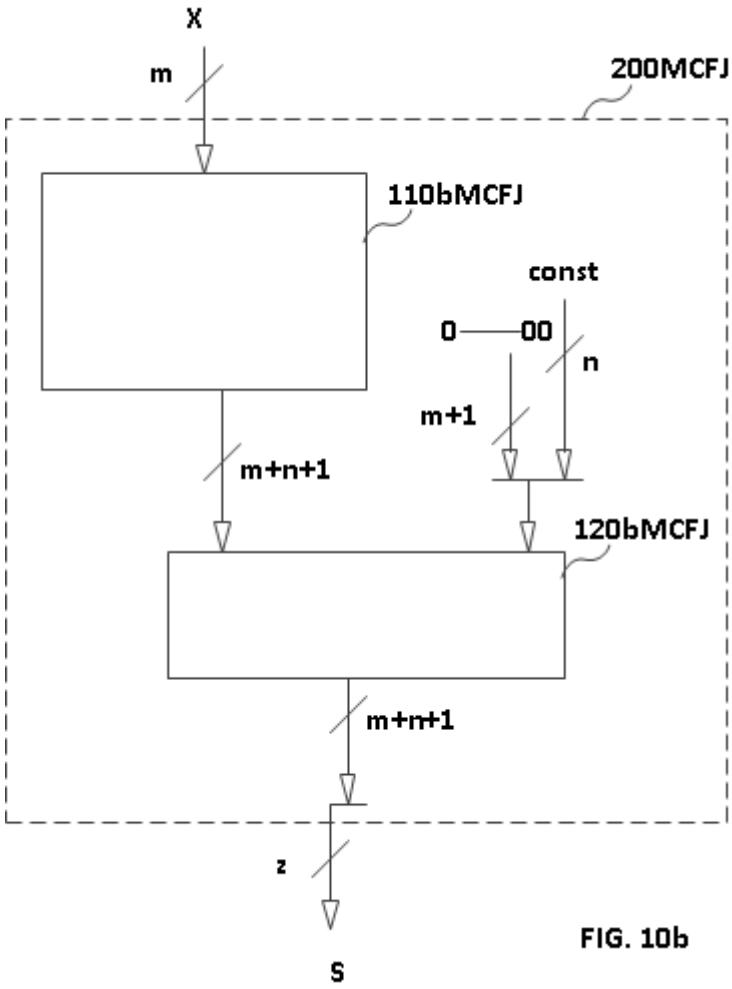

FIG. 10b

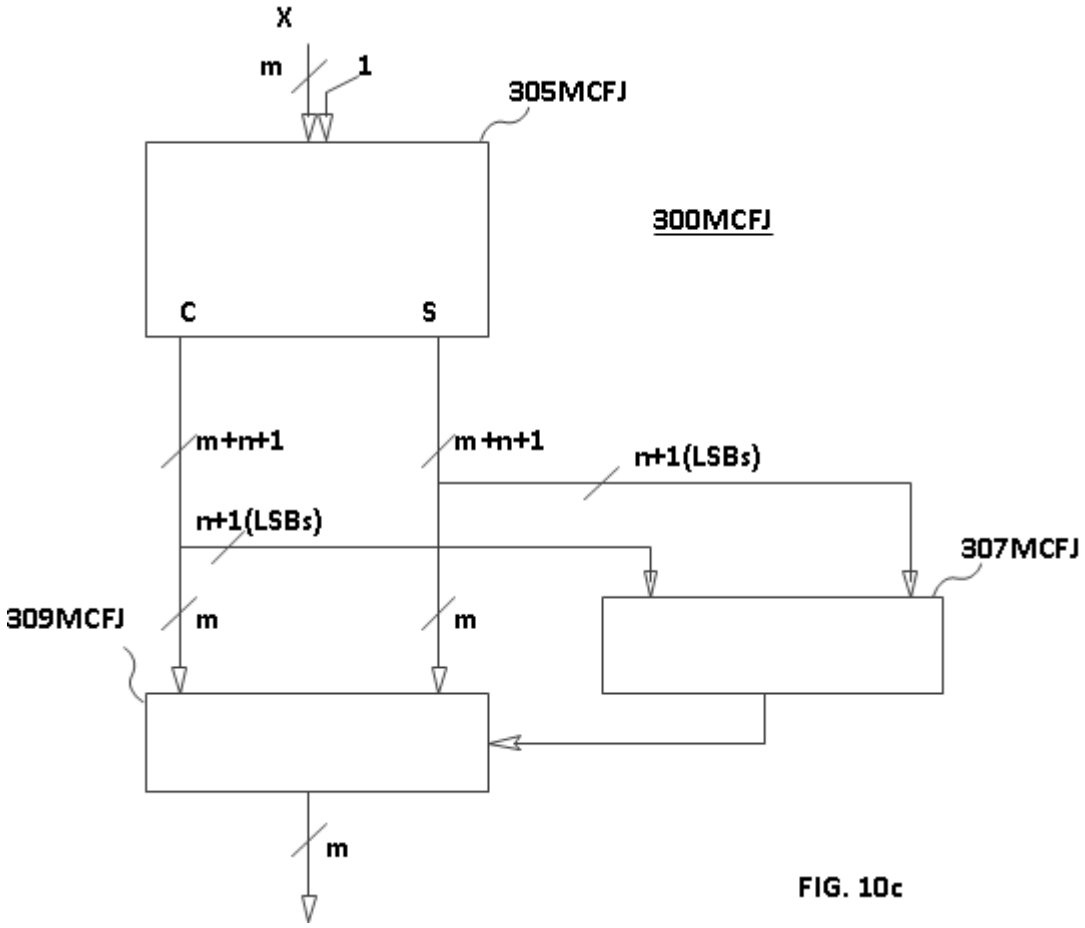
X
m
1
305MCFJ
300MCFJ
C
S
m+n+1
m+n+1
m+1(LSBs)
m+1(LSBs)
307MCFJ
309MCFJ
m
m
m

FIG. 10c

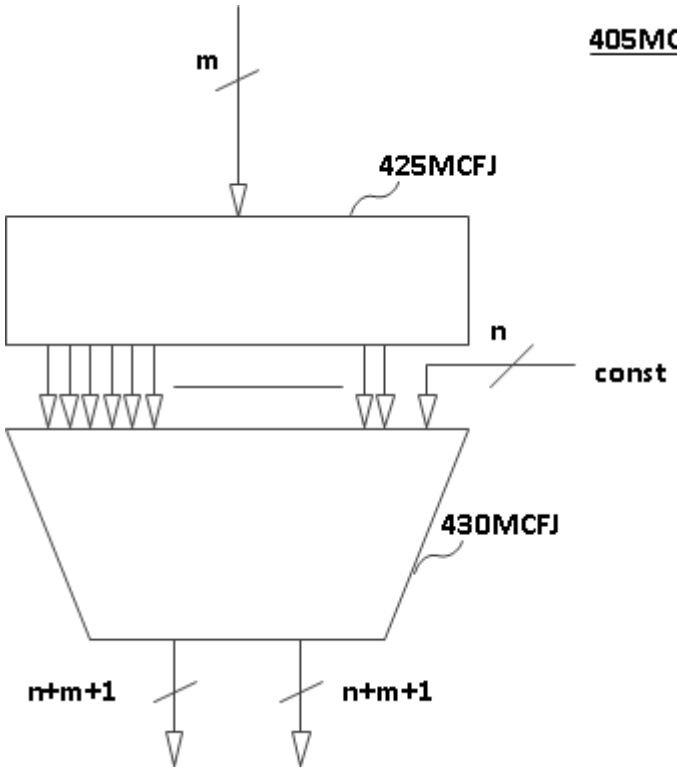

FIG. 11

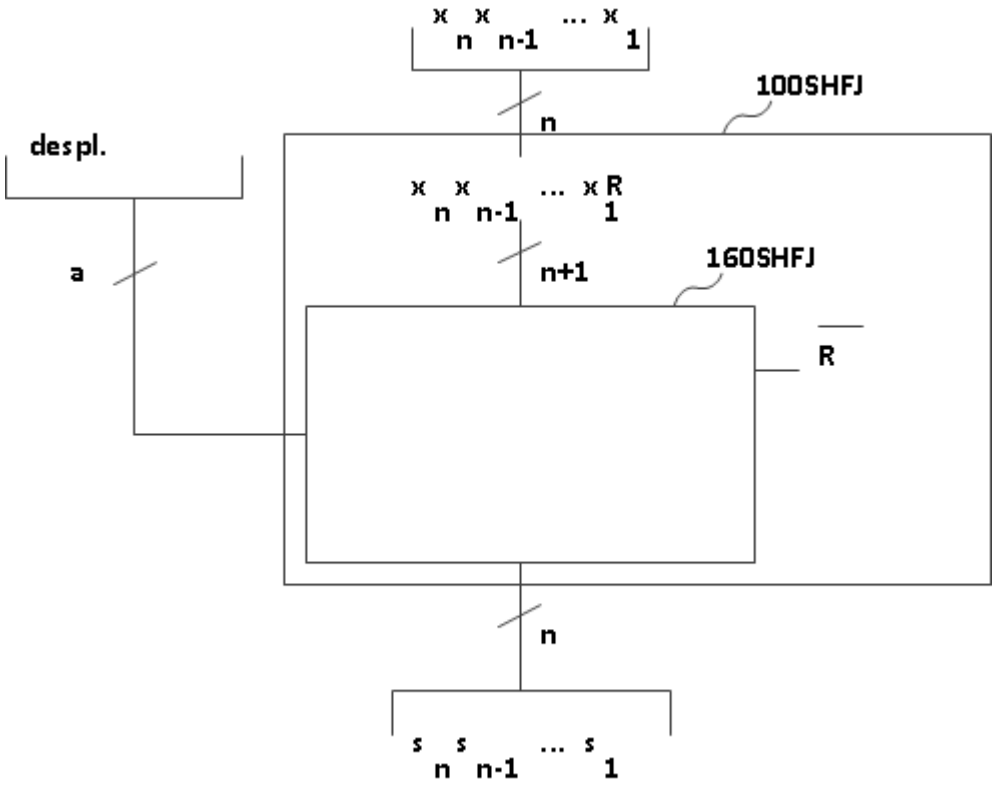

FIG. 12

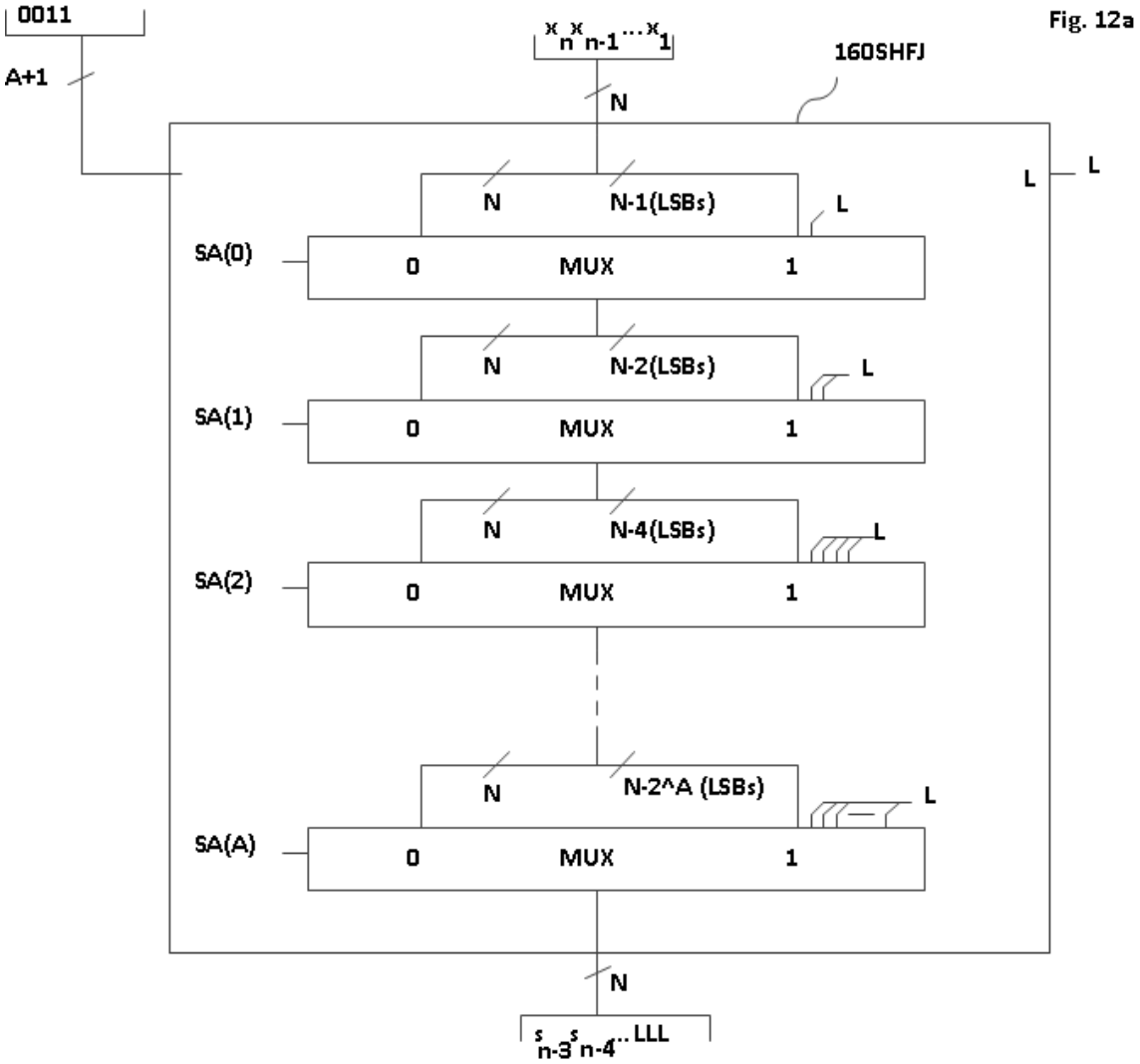
0011
A+1
$x_n x_{n-1} \cdots x_1$
N
160SHFJ
L
L
N
N-1(LSBs)
L
SA(0)
0
MUX
1
N
N-2(LSBs)
L
SA(1)
0
MUX
1
N
N-4(LSBs)
L
SA(2)
0
MUX
1
N
N-2^A (LSBs)
L
SA(A)
0
MUX
1
N
$s_{n-3} s_{n-4} \cdots LLL$

Fig. 12a

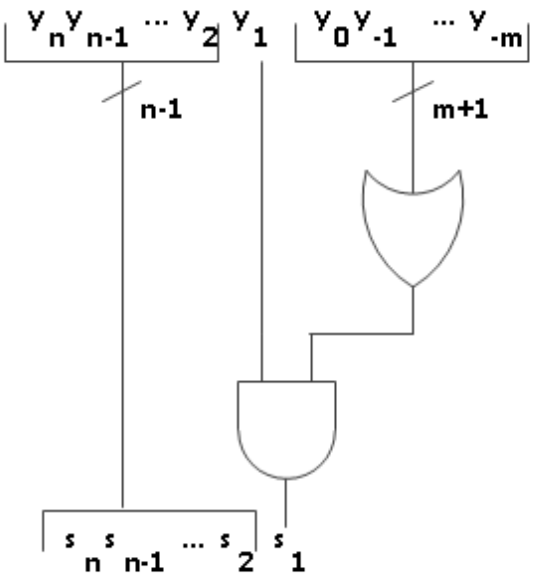

Fig. 13a

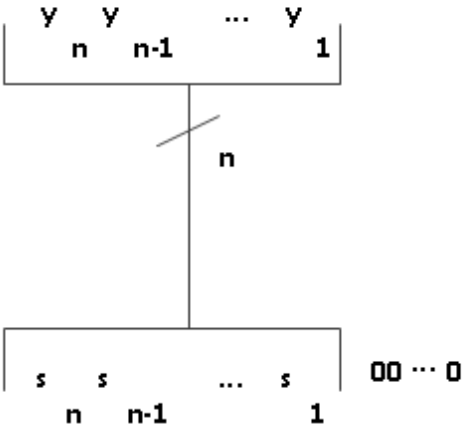

Fig. 13b

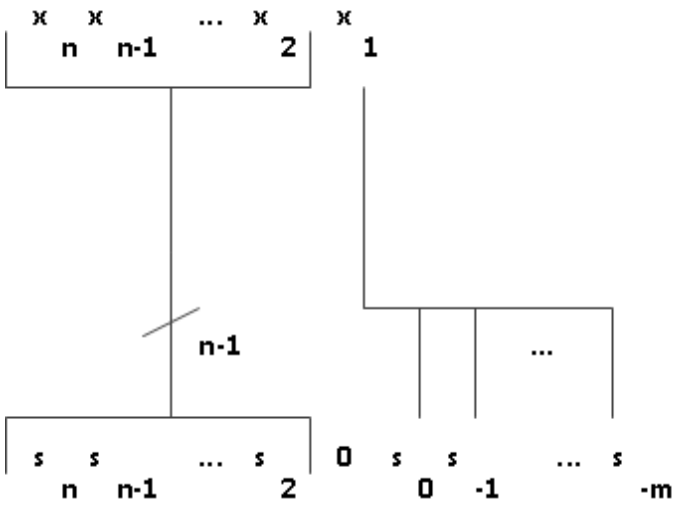

Fig. 13c

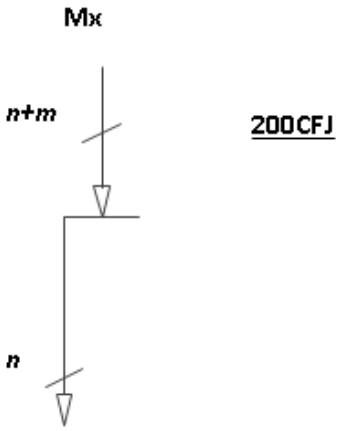

FIG. 14a

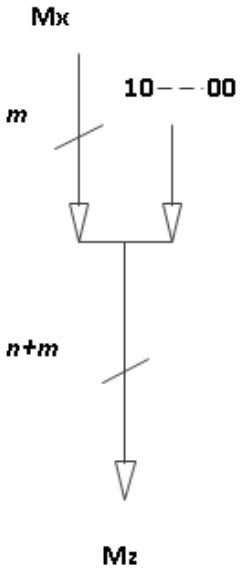

FIG. 14b

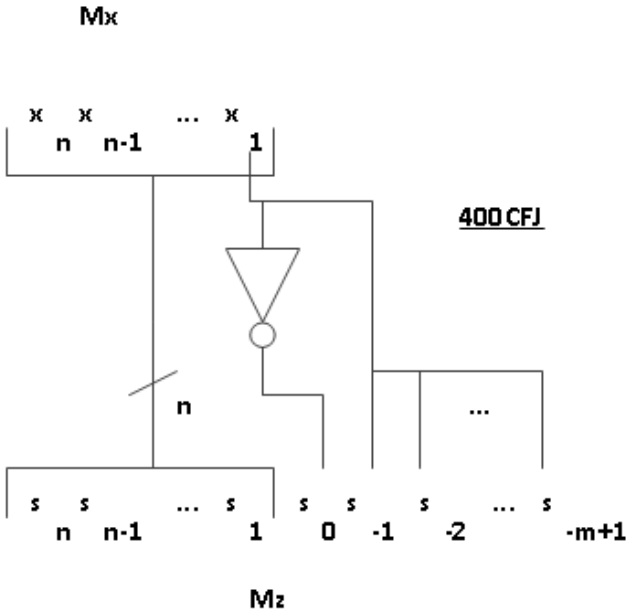

FIG. 14c

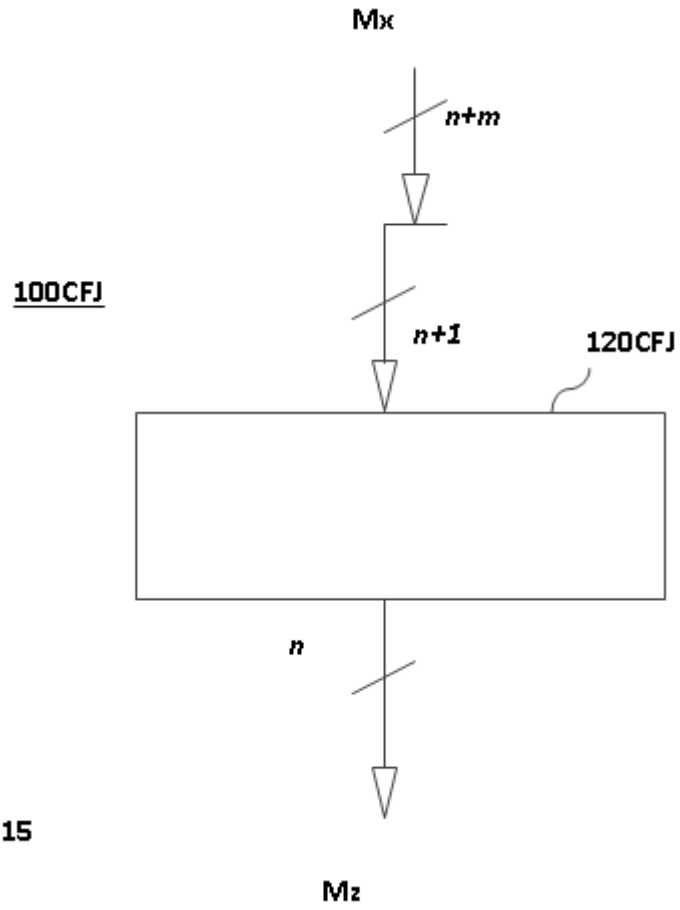

FIG. 15

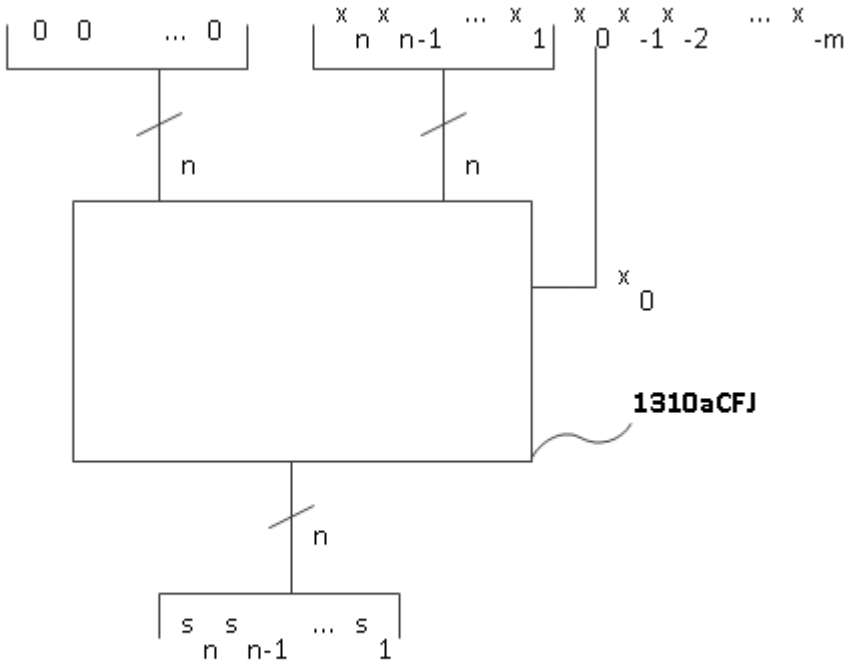

Fig. 15a

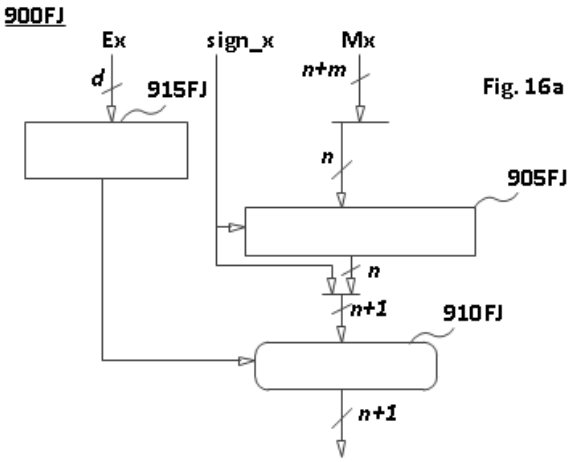

Fig. 16a

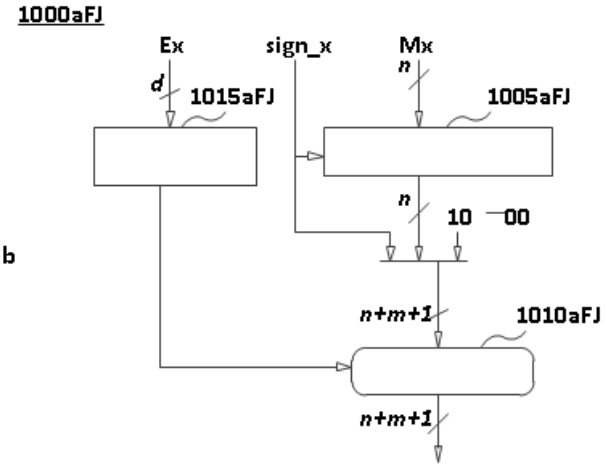

Fig. 16b

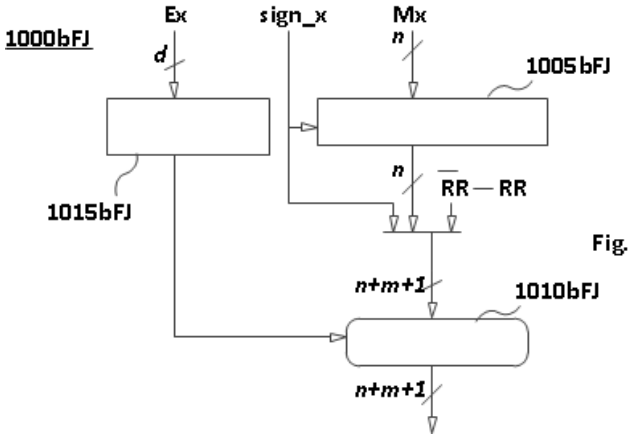

Fig. 16c

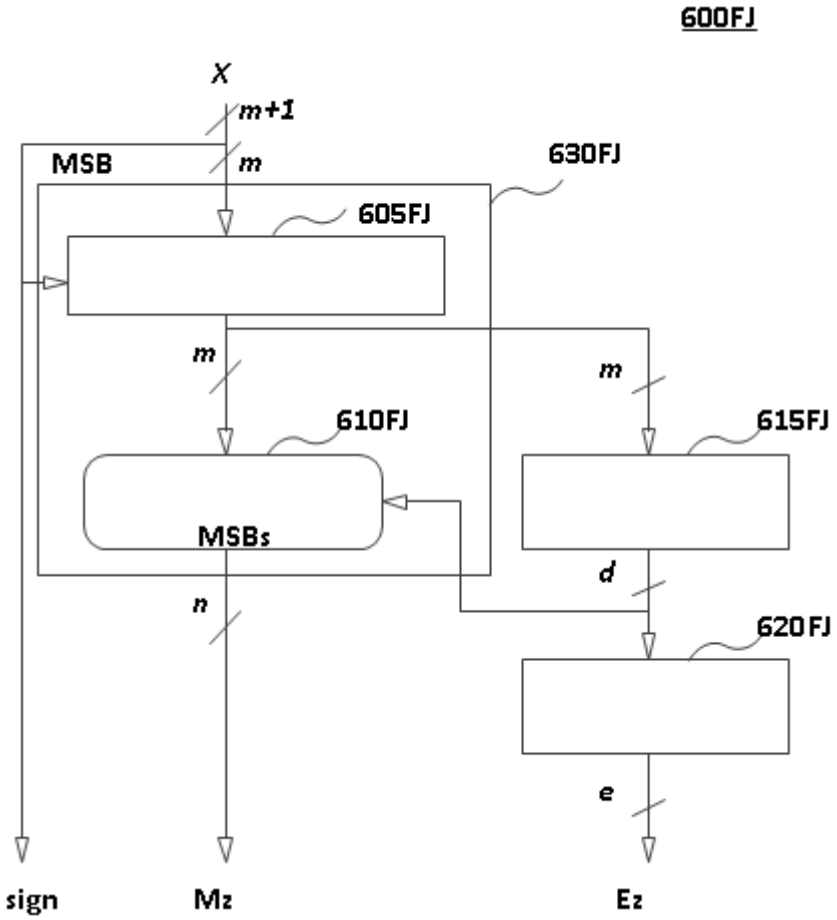
600FJ
X
m+1
MSB
m
630FJ
605FJ
m
m
610FJ
615FJ
MSBs
d
n
620FJ
e
sign
Mz
Ez

Fig. 17

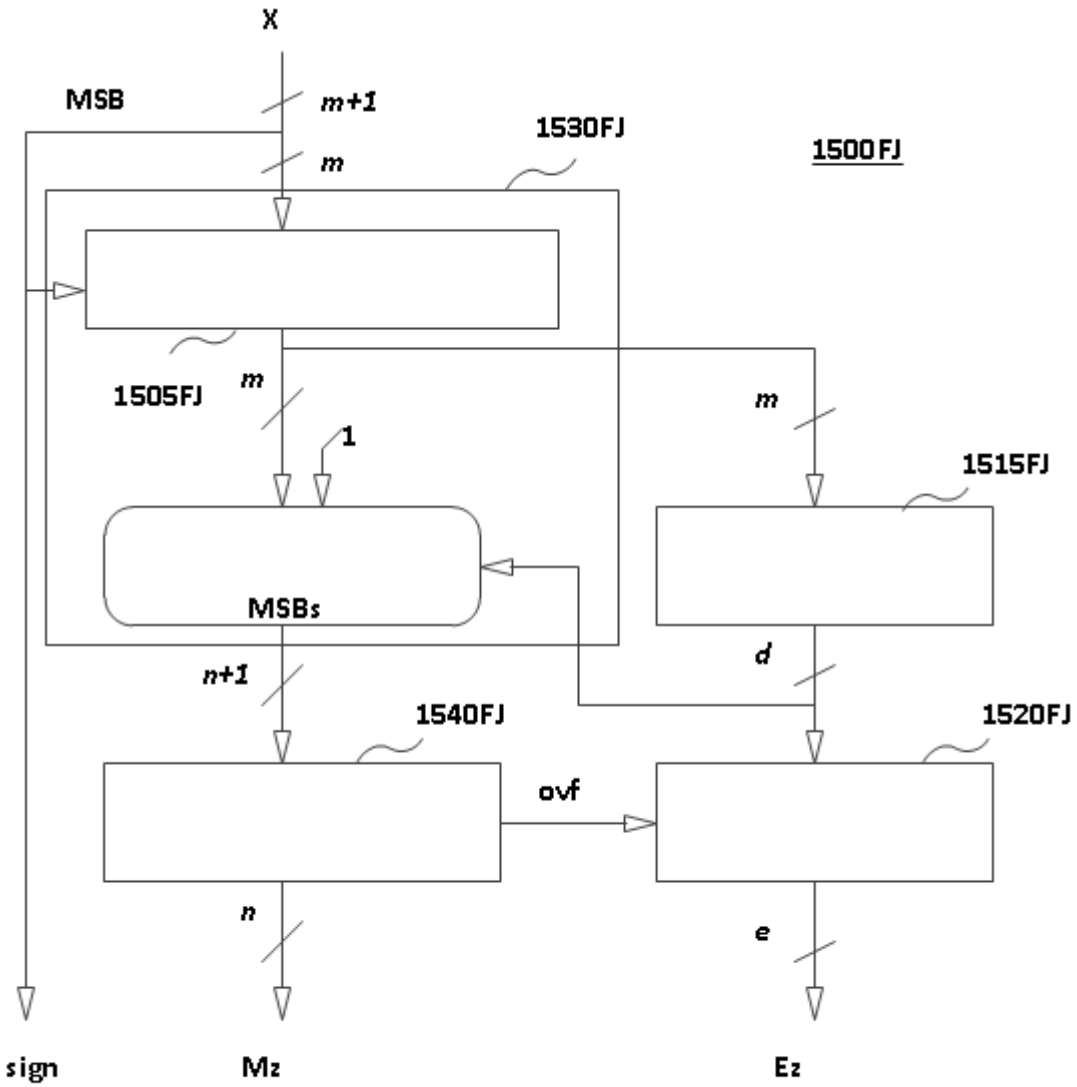
X
MSB
m+1
m
1530FJ
1500FJ
1505FJ
m
m
1
1515FJ
MSBs
d
n+1
1540FJ
1520FJ
ovf
n
e
sign
Mz
Ez

Fig. 18

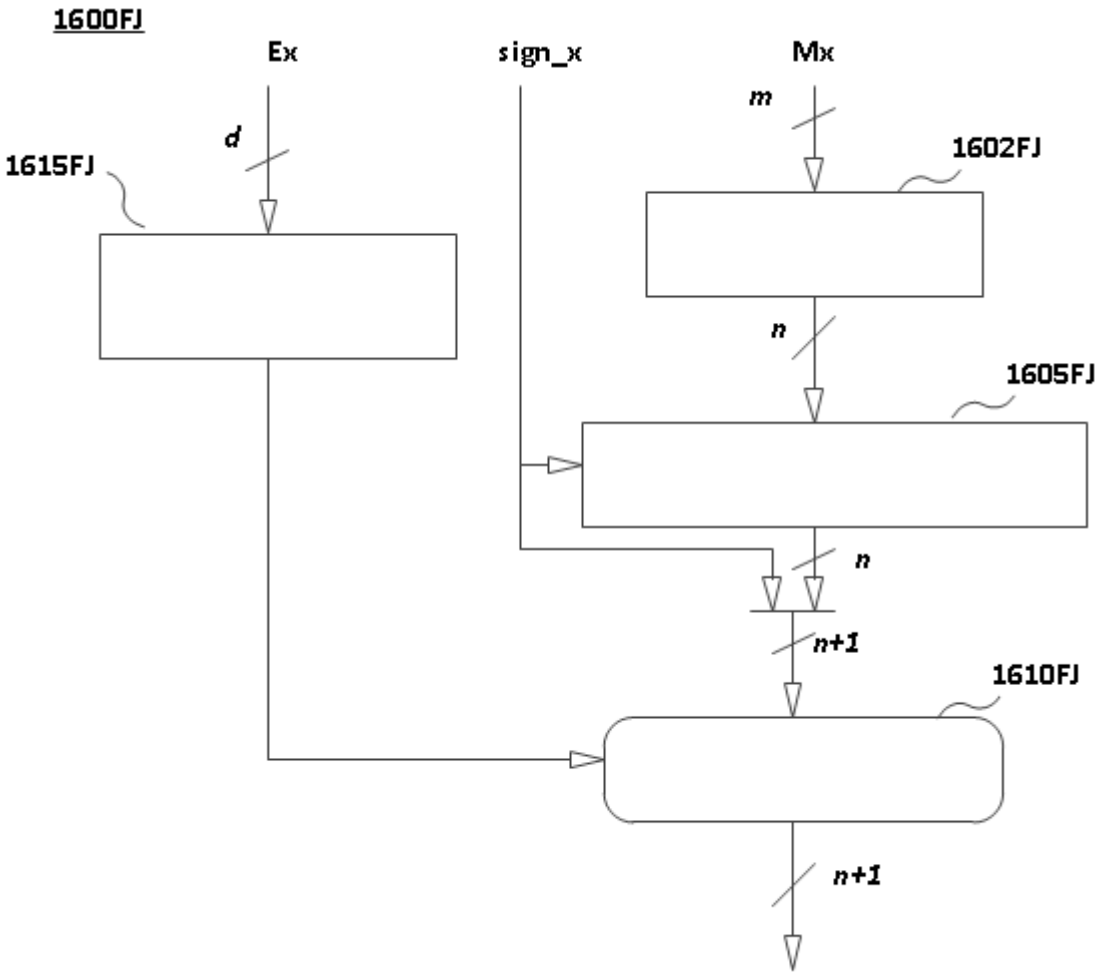

Fig. 19

OFICINA ESPAÑOLA
DE PATENTES Y MARCAS

ESPAÑA

(21) N.º solicitud: 201430456

(22) Fecha de presentación de la solicitud: 28.03.2014

(32) Fecha de prioridad:

INFORME SOBRE EL ESTADO DE LA TECNICA

(51) **Int. Cl. :** ***G06F7/38*** (2006.01)

DOCUMENTOS RELEVANTES

Categoría	(56) **Documentos citados**	**Reivindicaciones afectadas**
A	US 2010125621 A1 (OLIVER DAVID S et al.) 20.05.2010, todo el documento.	1
A	US 8484262 B1 (BRYAN TOM; MATHWORKS INC) 09.07.2013, todo el documento.	1
A	SOMSUBHRA GHOSH et al. FPGA based implementation of a double precision IEEE floatingpoint adder. Intelligent Systems and Control (ISCO), 2013 7th International Conference on, 20130104 IEEE 04.01.2013 VOL: Págs: 271-275 ISBN 978-1-4673-4359-6; ISBN 1-4673-4359-5 doi:10.1109/ISCO.2013.6481161. Todo el documento.	1

Categoría de los documentos citados

X: de particular relevancia
Y: de particular relevancia combinado con otro/s de la misma categoría
A: refleja el estado de la técnica
O: referido a divulgación no escrita
P: publicado entre la fecha de prioridad y la de presentación de la solicitud
E: documento anterior, pero publicado después de la fecha de presentación de la solicitud

El presente informe ha sido realizado

☒ para todas las reivindicaciones ☐ para las reivindicaciones nº:

Fecha de realización del informe	**Examinador**	**Página**
06.03.2015	M. Muñoz Sánchez	1/4

Documentación mínima buscada (sistema de clasificación seguido de los símbolos de clasificación)

G06F

Bases de datos electrónicas consultadas durante la búsqueda (nombre de la base de datos y, si es posible, términos de búsqueda utilizados)

INVENES, EPODOC, WPI

Fecha de Realización de la Opinión Escrita: 06.03.2015

Declaración

Novedad (Art. 6.1 LP 11/1986)	Reivindicaciones	1-58	**SI**
	Reivindicaciones		**NO**
Actividad inventiva (Art. 8.1 LP11/1986)	Reivindicaciones	1-58	**SI**
	Reivindicaciones		**NO**

Se considera que la solicitud cumple con el requisito de aplicación industrial. Este requisito fue evaluado durante la fase de examen formal y técnico de la solicitud (Artículo 31.2 Ley 11/1986).

Base de la Opinión.-

La presente opinión se ha realizado sobre la base de la solicitud de patente tal y como se publica.

1. Documentos considerados.-

A continuación se relacionan los documentos pertenecientes al estado de la técnica tomados en consideración para la realización de esta opinión.

Documento	Número Publicación o Identificación	Fecha Publicación
D01	US 2010125621 A1 (OLIVER DAVID S et al.)	20.05.2010
D02	US 8484262 B1 (BRYAN TOM; MATHWORKS INC)	09.07.2013
D03	SOMSUBHRA GHOSH et al. FPGA based implementation of a double precision IEEE floating point adder. Intelligent Systems and Control (ISCO), 2013 7th International Conference on, 20130104 IEEE VOL: Págs: 271-275 ISBN 978-1-4673-4359-6; ISBN 1-4673-4359-5 doi:10.1109/ISCO.2013.6481161. Todo el documento.	04.01.2013

2. Declaración motivada según los artículos 29.6 y 29.7 del Reglamento de ejecución de la Ley 11/1986, de 20 de marzo, de Patentes sobre la novedad y la actividad inventiva; citas y explicaciones en apoyo de esta declaración

Se considera D01 el documento más próximo del estado de la técnica al objeto de la solicitud.

Reivindicaciones independientes

Reivindicación 1: El documento D01, divulga una unidad de cómputo aritmético para realizar operaciones de suma o multiplicación de coma flotante con caminos de datos para la mantisa y el exponente respectivo. Los operandos tienen un bit de valor implícito 1 (el más significativo). Los resultados se redondean y normalizan.
Las diferencias entre el documento D01 y la reivindicación 1 son:
- el bit de valor implícito es el menos significativo
- los operandos son de coma fija.

El efecto técnico de estas diferencias es la simplificación de los cálculos de redondeo y truncamiento para operandos de coma fija. El problema técnico objetivo consistiría así en cómo simplificar los cálculos habituales asociados a las operaciones aritméticas con operandos de coma fija.

El documento D02 por su parte divulga un método para determinar los atributos de los resultados de operaciones aritméticas con operandos de coma fija en el que se fijan distintos criterios para determinar la posición de la coma. En este documento tampoco se menciona que el bit menos significativo sea de valor implícito igual a 1.

El documento D03 por su parte divulga una implementación de un sumador de coma flotante en el que el bit implícito es el más significativo. La suma se realiza tras el preprocesamiento de los operandos. En este documento tampoco se recoge la diferencia relativa al valor del bit menos significativo mencionada en el análisis de los documentos D01 o D02 por lo que la reivindicación 1 posee actividad inventiva según el art. 8.1 de la Ley de Patentes.

Reivindicaciones dependientes

Reivindicaciones 2-58: estas reivindicaciones poseen actividad inventiva según el art. 8.1 de la Ley de Patentes porque dependen de la reivindicación 1 que, como se ha mencionado, también la tiene.