(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 295 829**

(51) Int. Cl.:
G06F 7/58 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Número de solicitud europea: **04714771 .5**
(86) Fecha de presentación : **26.02.2004**
(87) Número de publicación de la solicitud: **1719052**
(87) Fecha de publicación de la solicitud: **08.11.2006**

(54) Título: **Procedimiento y circuito para generar números aleatorios y producto informático para ordenar del mismo.**

(45) Fecha de publicación de la mención BOPI:
16.04.2008

(45) Fecha de la publicación del folleto de la patente:
16.04.2008

(73) Titular/es: **Telecom Italia S.p.A.**
Piazza Degli Affari 2
20123 Milano, IT

(72) Inventor/es: **Ghigo, Giovanni y**
Bollea, Loris

(74) Agente: **Ponti Sales, Adelaida**

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín europeo de patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre concesión de Patentes Europeas).

Venta de fascículos: Oficina Española de Patentes y Marcas. Pº de la Castellana, 75 – 28071 Madrid

DESCRIPCIÓN

Procedimiento y circuito para generar números aleatorios y producto informático para ordenar del mismo.

Campo de la invención

La presente invención se refiere a generadores de números aleatorios que podrían ser implementados en circuitos integrados digitales tales como circuitos integrados para aplicaciones específicas (ASICs siglas de *Aplicación Specific Integrated Circuits*) y Arrays de puertas programables de campo (FPGAs, siglas de *Field Programmable Gate Arrays*), y más especialmente se refiere a un generador de números aleatorios de alta velocidad del tipo en el que un generador de números pseudo-aleatorios utiliza como semillas los números aleatorios generados por un verdadero generador de números aleatorios.

Antecedentes de la invención

Muchas aplicaciones, tales como la simulación, los juegos, la estadística, la criptografía y las aplicaciones de seguridad en general, etc, precisan del uso de generadores de números aleatorios.

En general, los generadores de números aleatorios existentes se pueden subdividir en dos categorías principales: los generadores basados en *hardware* y los generadores basados en *software*.

Los generadores basados en *hardware*, llamados comúnmente generadores de números aleatorios verdaderos (TRNG, siglas de True Random Number), sacan provecho de fenómenos físicos que ocurren aleatóriamente tales como ruido eléctrico, fenómenos físicos que ocurren aleatóriamente tales como ruido eléctrico, oscilaciones inestables, biestables metaestables, etc. Cuando se utiliza un generador de números aleatorios verdadero, se deben adoptar medidas para evitar que la secuencia generada se vea afectada por el *biasing* (la tendencia a generar preferentemente uno de dos valores digitales), correlación a corto plazo y otras desviaciones de la aleatoriedad, tal como se describe por ejemplo en el artículo "Generadores de números aleatorios con *hardware*" ("*Hardware número aleatorio generador*" presentado por R. Davies en la 15ª Conferencia de Estadística Australiana, en julio de 2000, y publicado el 14 de octubre de 2000 en la página de Internet http: //www.robertnz.net. El principal problema de estos generadores es que, al basarse en un principio físico realmente aleatorio, no garantizan una determinada tasa de bits y la propia tasa de bits no es muy alta. Esto hace que los TRNGs no sean adecuados para aplicaciones tales como las simulaciones, la criptografía en tiempo real, como por ejemplo en telecomunicaciones, etc.

Los generadores mediante *software*, llamados comúnmente pseudo-Generadores de números aleatorios (PRNG), son generadores deterministas que utilizan un algoritmo algebraico para crear una secuencia periódica de números aleatorios con un periodo muy largo que depende de la arquitectura del generador. Se describen varios de estos generadores en "Criptografía aplicada: Protocolos, algoritmos y código fuente en C"("*Applied Criptografía: Protocols, Algorithms y Source Code in C*"), de B. Schneier, 2ª Edición, 1996, J. Wiley & Sons, Inc., en especial en el capítulo 16. Estos generadores pueden operar con una tasa de bits predeterminada y relativamente alta. Aunque, puesto que la secuencia se basa en un proceso algorítmico, es posible deducir el número aleatorio n+1 si se conoce el número aleatorio n. Esto permite recuperar la "semilla" empleada para la generación. En algunas aplicaciones, por ejemplo en simulación, este hecho no es importante y no supone un problema para el funcionamiento correcto de la aplicación. En otros casos, por ejemplo en aplicaciones de criptografía y seguridad, la posibilidad de recuperar la semilla, aunque sea con un esfuerzo considerable, es, por supuesto, una debilidad en la cadena de seguridad.

Una solución adoptada comúnmente para superar los problemas mencionados es combinar un generador aleatorio verdadero con un generador pseudo-aleatorio y utilizar la salida del generador aleatorio verdadero para modificar la secuencia pseudo-aleatoria, que constituye la secuencia de salida. Por ejemplo, los números aleatorios verdaderos son utilizados como semilla para el generador pseudo-aleatorio. Esta disposición en la que unos números aleatorios verdaderos son usados como semillas por el generador pseudo-aleatorio se describe en US2004/0019619. La utilización de un generador pseudo-aleatorio resuelve el problema de la baja tasa de bits del generador basado en *hardware*. Modificando continuamente la salida del generador pseudo-aleatorio utilizando la secuencia aleatoria verdadera se intenta superar el problema de la naturaleza determinista del pseudo-generador de números aleatorios.

Son conocidas en el sector de la técnica diversas implementaciones de esta solución.

El artículo antes mencionado de R. Davies sugiere combinar los *bit-streaming* del PRNG y del TRNG mediante una simple operación EX-OR (suma lógica exclusiva).

Existen técnicas algo más sofisticadas en la literatura de patentes.

En US-B 6,408,317 se modifica un PRNG en forma de un registro de desplazamiento con retroalimentación lineal (LFSR, siglas de *Linear Feedback Registro de desplazamiento*) utilizando una función condensadora para calcular un flujo de bits a los que hay que aplicar un EX-OR con la salida del propio LFSR. Además, la función de retroalimentación del LFSR es modificada continuamente por los bits aleatorios que provienen de un TRNG a una tasa de bits constante. Por lo tanto, el flujo de bits de salida debería tener la misma tasa de bits que el TRNG. El dispositivo des-

crito en ese documento no puede alcanzar altas tasas de bits y, además, únicamente utiliza un factor de aleatoriedad, es decir el valor aleatorio de la Secuencia TRNG.

En US-A 2003/0059046 se incluye un generador determinista compuesto de tres LFSRs, cuya interacción se modifica mediante una función de mezclado dirigida por un TRNG que emite una secuencia a una tasa de bits constante. También en este caso se utiliza únicamente un factor de aleatoriedad.

En US-A 2002/0156819 también se modifica un LFSR que actúa como PRNG mediante un operador EX-OR en la entrada del propio LFSR. En este caso, la señal modificadora es generada mediante la interacción de tres osciladores libres que tienen frecuencias de oscilación diferentes, también diferentes del reloj de sistema que dirige el LFSR. También en este caso se puede utilizar el valor aleatorio de la salida EX-OR, puesto que la dependencia temporal en la señal modificadora es muy débil, fuertemente correlacionada y en su mayor parte determinista. Realmente, la señal modificadora es suministrada al LFSR después de haber sido adaptada utilizando el reloj del sistema de manera que se obtiene una señal modificadora a una tasa de bits constante.

Por lo tanto, todas las propuestas del estado de la técnica para modificar el comportamiento del generador pseudo-aleatorio están basadas en técnicas que utilizan un único factor de aleatoriedad, de manera que son bastante simples y no garantizan una buena y profunda interacción con el PRNG. Por lo tanto, sigue habiendo una posibilidad no despreciable de ataque.

Es un objetivo principal de la invención proporcionar un generador de números aleatorios y un procedimiento de generación de números aleatorios, basados en la modificación de una secuencia pseudo-aleatoria mediante una secuencia aleatoria verdadera, y en la que se utiliza una técnica más sofisticada para modificar el comportamiento del generador pseudo-aleatorio, aprovechando tanto el valor aleatorio como la llegada aleatoria de los bits en la secuencia aleatoria verdadera.

Resumen de la invención

Según la invención, se proporciona un generador de números aleatorios de alta velocidad, que comprende: un generador de números aleatorios verdaderos; un generador de números pseudo-aleatorios, dispuestos para generar una secuencia pseudo-aleatoria utilizando los números aleatorios verdaderos generados por dicho generador de números aleatorios verdaderos como semillas aleatorias; y a una lógica de mezclado conectada entre dicho generador de números aleatorios verdaderos y dicho generador de números pseudo-aleatorios y dispuesta para modificar el comportamiento de dicho generador de números pseudo-aleatorios utilizando las semillas aleatorias. Dicho generador se caracteriza por el hecho de que dicho generador aleatorio verdadero está dispuesto para generar una secuencia aleatoria de bits de tasa variable, por ejemplo un tiempo entre llegadas de bits aleatorio, mediante el cual las semillas se vuelven disponibles en instantes aleatorios; y por el hecho de que dicha lógica de mezclado comprende un generador de una señal de modificación destinada a cambiar el comportamiento de dicho generador de números pseudo-aleatorios en múltiples instantes aleatorios en el intervalo entre dos semillas consecutivas, de modo que se obtienen en dicho intervalo varias secuencias pseudo-aleatorias de longitudes variables más cortas que la longitud aleatoria determinada por la llegada de dos valores de semilla consecutivas, estando dicho generador de la señal de modificación conectado de modo que recibe dicha semilla y genera dicha señal de modificación procesando dicha semilla mediante la secuencia generada por dicho pseudo-generador de números aleatorios.

La invención también proporciona un procedimiento de generación de alta velocidad de números aleatorios alterando una secuencia pseudo-aleatoria mediante números aleatorios verdaderos que forman semillas aleatorias para la generación de dicha secuencia pseudo-aleatoria, comprendiendo el procedimiento las etapas de: obtener las semillas aleatorias de una secuencia aleatoria de bits que tienen tasa variable; procesar una semilla aleatoria para generar una señal de modificación que explota el tiempo de llegada aleatorio de los bits de dicha secuencia de bits; y cambiar la secuencia pseudo-aleatoria mediante dicha señal de modificación en instantes aleatorios entre la llegada de dos semillas consecutivas, para así obtener en dicho intervalo varias secuencias pseudo-aleatorios de longitudes aleatorias más cortas que las longitudes determinadas por la llegada de dos semillas consecutivas, siendo dicha señal de modificación generada bajo control de la secuencia pseudo-aleatoria.

La invención también proporciona un producto informático para ordenador que se puede cargar en la memoria de al menos un ordenador y que comprende porciones de código de software para realizar las etapas del procedimiento de la invención cuando el programa se ejecuta en un ordenador. Tal como se utiliza aquí, la referencia a tal producto de programa para ordenador pretende ser un equivalente a la referencia a un medio legible por un ordenador que contiene instrucciones para controlar un sistema de ordenador para coordinar la ejecución del procedimiento de la invención. La referencia a "al menos un" ordenador pretende obviamente resaltar la posibilidad de adaptar la invención para ser implementada de forma descentralizada.

Por lo tanto, la invención proporciona un generador y un procedimiento en los cuales se modifica la salida determinista del PRNG mediante una técnica de modificación más sofisticada que explota una pluralidad de acontecimientos aleatorios, de manera que se reduce drásticamente la posibilidad de deducir la ley de generación observando la secuencia de salida y se obtiene una seguridad mucho mayor con respecto a los sistemas del estado de la técnica que están basados únicamente o casi únicamente en el valor aleatorio de una TRNG.

Breve descripción de los dibujos

A continuación, se describirá una realización preferida de la invención, ofrecida a título de ejemplo no limitativo haciendo referencia a los dibujos adjuntos en los cuales:

- la figura 1 es un diagrama de bloques general del generador según la invención;

- la figura 2 es un diagrama de bloques y de circuito más detallado de los bloques 102 -104 de la figura 1;

- la figura 3 es un diagrama de bloques que muestra con mayor detalle el circuito mezclador de la figura 2; y

- la figura 4 es un diagrama de la matriz de conmutación del circuito mezclador.

Descripción detallada de la realización preferida

La figura 1 muestra los bloques funcionales principales del generador de números aleatorios de alta velocidad según la invención, que se designará por 100. Igual que en los generadores del estado de la técnica mencionados, el generador 100 se basa en la utilización de un generador de números aleatorios verdaderos (TRNG) 101 y un generador de números pseudo-aleatorios (PRNG) 102.

El TRNG 101 es la fuente de una secuencia de bits aleatorios de tasa de bits variable, es decir un tiempo entre llegadas variable, a partir de la cual se genera una semilla aleatoria para la secuencia generada por el PRNG 102. Una lógica de mezclado 103 recibe la secuencia de salida del TRNG 101, presente en la línea 109, y obtiene de ella la semilla aleatoria a aplicar al PRNG 102 para introducir modificaciones aleatorias en su comportamiento. Una lógica de salida 104 recibe el flujo de bits en serie emitido por el PRNG 102 en la línea 105 y proporciona una salida paralela en la línea 106 convirtiendo la señal de 1 bit en palabras que tienen una longitud deseada. La salida lógica 104 puede ser controlada por la lógica de mezclado 103, tal como se esquematiza en la línea 110 y tal como se explicará en detalle más adelante.

El núcleo de la invención es la lógica de mezclado 103, junto con su interacción con el PRNG 102 mediante señales de retroalimentación y de proalimentación (líneas 107 y 108, respectivamente).

Según la invención, la lógica de mezclado 103 explota varias fuentes de aleatoriedad, a saber:

- los valores aleatorios de la secuencia de bits a la salida de TRNG 101;

- el tiempo de llegada aleatorio de los bits generados por TRNG 101. Para que esta fuente de aleatoriedad también pueda ser explotada, estos bits deberían tener una tasa de bits variable, es decir un tiempo entre llegadas, entre dos o más bits aleatorios, que no debería ser constante. Esta condición puede satisfacerse con un TRNG basado en fenómenos físicos tales como la metaestabilidad y el ruido térmico, entre otros. Preferentemente, se utiliza un TRNG que explota la metaestabilidad de un flip-flop. Un ejemplo de TRNG de este tipo, que proporciona una secuencia de bits con la variabilidad requerida, se describe en WO-A 01/67231.

La figura 2 muestra un diagrama más detallado del diagrama de bloques de PRNG 102, de la lógica de mezclado 103 y de la salida lógica 104.

En esta realización se supone a modo de ejemplo no limitativo que el PRNG 102 se basa en un registro de desplazamiento de retroalimentación lineal (LFSR, siglas de *Linear Feedback Shift Register*). Las referencias 204a, 204b respectivamente designan el registro de desplazamiento actual y la lógica de retroalimentación, que suele ser una puerta EX-OR, que combina y retroalimenta la entrada del registro de desplazamiento con la salida del registro de desplazamiento. Para mayor claridad en la descripción y únicamente a modo de ejemplo, se supone que el registro de desplazamiento 204a tiene una longitud de 40 bits. De aquí en adelante, se utilizarán los términos LFSR y PRNG indiferentemente.

La entrada de la lógica de mezclado 103 es una señal BITOUT que representa el flujo de bits en serie que viene del TRNG 101 a través de una línea adecuada 109. En asociación con esta señal, la señal BITVALID en otro cable de línea 109 indica si un bit del flujo se debe considerar como aleatorio (es decir que representa realmente una condición metaestable, en el ejemplo de realización de TRNG que se considera aquí). La señal BITVALID es la señal que habilita un primer registro de desplazamiento 202 (el cual también se denomina como entrada FIFO) y un primer contador 201 (el cual también se denomina como contador de datos FIFO). La entrada FIFO 202 carga los bits válidos en el flujo BITOUT y actúa como convertidor de serie a paralelo. Las señales puestas en paralelo salen a un bus 107a, que pertenece a la línea 107 de la figura 1 como palabras de un contador FIFO (FIFOCOUNT) de longitud determinada, que forman una semilla aleatoria para el LFSR 102. En la realización del ejemplo de un LFSR de 40 bits de longitud, el FIFOCOUNT también es de 40 bits. El contador de datos FIFO 201 cuenta los bits cargados en el FIFO de entrada 202 y, cuando alcanza el valor FIFO COUNT, genera una señal de “Datos Válidos”, la cual indica que se dispone de una nueva semilla aleatoria en el bus 107a.

El bus de salida 107a de FIFO 202 está conectado a las entradas paralelas del LFSR 102 y de un segundo registro de desplazamiento 207 (también denominado registro de reciclaje). La semilla aleatoria presente en el bus 107a se carga directamente en el registro de reciclaje 207 cada vez que la señal de "Datos Válidos" es generada por el contador de datos FIFO 201. En cambio, la carga en el LFSR 102 es indirecta, y está controlada por una señal de carga generada por un segundo contador 205 (también denominado contador de muestras) en el cable 107b de la línea 107, cuando este contador recibe la señal de datos válidos en su entrada de carga. La señal de carga en el cable 107b también se utiliza como puesta a cero del contador de datos FIFO 201 e indica el comienzo de una nueva fase de acumulación de bits aleatorios en el FIFO de entrada 202.

Este tipo de control de conformidad generado por el contador de muestras 205 es necesario porque es el enlace entre los dos dominios de reloj: el primer reloj (reloj CLK de secuencia BITOUT en la entrada de la lógica de mezclado 103) es un reloj de alta velocidad y el segundo (llamado Clk1M) es un reloj utilizado por el LFSR 102 de velocidad menor, por los bloques de bajada de la lógica de mezclado 103 y por la salida lógica 104. El reloj de velocidad menor se escoge de modo da un buen compromiso entre la tasa de bits de salida y la seguridad (cuanto más veloz es el reloj, más largas son las sub-secuencias que resultan de las modificaciones del comportamiento del LFSR 102 entre dos semillas consecutivas, tal como se explicará más adelante). Unos valores adecuados para el reloj más lento Clk1M y el reloj más rápido CLK podrían ser 1 MHz y 25 MHz, respectivamente. El reloj de velocidad más lenta se deriva directamente del reloj de velocidad más rápida por un tercer contador 203: de manera que el circuito es síncrono y proporciona una tasa de bits de salida constante independiente e incluso múltiple de la tasa de bits variable de los datos aleatorios.

Hay que destacar que el reloj Clk1M corresponde en general a una tasa de bits mucho mayor que la de los bits válidos en BITOUT, de manera que la presente invención puede introducir una multiplicación de la tasa de bits.

En ausencia de otras señales de modificación aparte de la semilla aleatoria, el LFSR 102 generaría bits pseudo aleatorios de una secuencia iniciada con una semilla actual y continuaría según esta secuencia hasta que se cargue una nueva semilla. Cuando llega una nueva semilla, el LFSR 102 salta a un nuevo punto de la secuencia, especificada por la nueva semilla, y sigue generando bits a partir de este punto. Puesto que el tiempo de llegada de una nueva semilla es aleatorio, ello no garantiza una frecuencia mínima de repuesto para la semilla del LFSR 102. Consecuentemente, hay la posibilidad de que se produzcan secuencias pseudo-aleatorias largas que podrían reducir la seguridad criptográfica del generador.

Para evitar esta posibilidad, la lógica de mezclado 103 tiene un circuito especial que saca provecho del tiempo de llegada variable de nuevas semillas para añadir más aleatoriedad al LFSR 102. Este circuito consiste en un contador de muestras 205, un cuarto contador 206 (también denominado contador de reciclaje) y un registro de reciclaje 207.

Más concretamente, cuando llega una semilla (Señal de Dato Válido del contador 201), el contador de muestras 205 se carga con algunos bits de la semilla, suministrados por el registro de reciclaje 207 mediante el bus 212. El número de bits que se deben cargar en el contador 205 se indica por un parámetro SAMPLEMAXCOUNT. Con la longitud de semilla de 40 bits del ejemplo, un valor adecuado de SAMPLEMAXCOUNT podría ser 10.

El contador de muestras 205 es un contador de cuenta atrás que cuenta hacía abajo a partir del valor indicado por los bits en el bus 212 hasta cero. Al final de su cuenta, el contador 205 genera una señal de final de cuenta TC1 que retroalimenta el EXOR 204b del LFSR 102 mediante el cable 107c y modifica, para un ciclo de reloj, el polinomio de registro de desplazamiento 204a. Esto causa un cambio en la secuencia del LFSR, lo cual resulta en una secuencia pseudo-aleatoria más corta que si se esperara una nueva semilla. Además, el contador de muestras 205 está construido de tal manera que, tras la generación de TC1, se vuelve a cargar con un nuevo valor. Por lo tanto, la modificación del comportamiento del LFSR 102 ocurre en múltiples instantes aleatorios en el intervalo entre dos señales de Datos Válidos consecutivos. Obviamente, siempre que llega una nueva señal de Datos Válidos al contador de muestras 205, se carga un nuevo conjunto de bits, aunque el contador 205 no haya alcanzado su cuenta final, de manera que la modificación también explota los tiempos aleatorios de llegada de semillas.

La señal TC1 también se envía al contador de reciclaje 206 como señal de carga, con los fines que se explicarán más adelante.

Tal como se ha dicho, los bits SAMPLEMAXCOUNT que deben ser cargados en el contador de muestras 205 se toman del registro de reciclaje 207 que, al principio, contiene una copia de la señal.

El registro de reciclaje 207 es un registro de desplazamiento de recirculación que, durante la cuenta atrás del contador de muestras 205, rota la semilla en una cantidad especificada por el contador de reciclaje 206. El contador de reciclaje 206 también es un contador de cuenta atrás que se carga con algunos bits tomados del LFSR 102, y cuenta atrás desde el valor especificado por dichos bits hasta 0. La señal de fin de cuenta TC2 del contador 206 se suministra al registro de reciclaje 207 para parar la rotación.

La cantidad de bits que se deben cargar en el contador de reciclaje 206 se especifica con un parámetro RECYCLE COUNT. Un valor de este parámetro adecuado, con un determinado valor de SAMPLEMAXCOUNT, podría ser 5. Estos bits son transferidos al contador de reciclaje 206 mediante la línea 108 y se cargan en dicho contador cuando el contador de muestras 205 genera una señal de fin de cuenta TC1. Por lo tanto, el valor de conteo de inicio del contador

de reciclaje 206 es un valor aleatorio y causa la rotación de la semilla en una cantidad aleatoria en cada ciclo. Así, cada vez que se carga el contador de muestras 205, se carga con un número SAMPLEMAXCOUNT de bits diferentes y aleatóriamente variables, los cuales a su vez son extraídos de una secuencia aleatoria.

La forma escogida para suministrar los bits SAMPLEMAX COUNT y RECYCLECOUNT al contador de muestras 205 y al contador de reciclaje 206 no tiene ninguna relevancia. Obviamente, la solución más simple es utilizar bits contiguos, aunque también se pueden utilizar bits extraídos de cualquier posición de los registros de desplazamientos 204a y 207.

En conclusión, el comportamiento del LFSR es modificado aleatóriamente por el contador de muestras 205 y, sucesivamente, la secuencia del LFSR modificada aleatóriamente se utiliza para modificar aleatóriamente el valor contado por el contador de muestras 205. Esta clase de interacción de proalimentación y retroalimentación entre la lógica de mezclado 103 y el generador pseudo-aleatorio 102 ayuda a evitar ciclos y bucles.

Gracias a la cooperación entre los contadores 205, 206 y el registro 207, el LFSR 102 genera, en lugar de una única secuencia pseudo-aleatoria de bits bastante larga definida por la llegada de dos semillas consecutivas, un número aleatorio de secuencias más cortas de longitudes aleatorias. Esto hace extremadamente difícil intentar deducir la ley de generación de la secuencia de salida incluso tras una observación a largo plazo de la misma.

La secuencia pseudo-aleatoria de bits que sale en serie del LFSR 102 por la línea 105 es transferida a un tercer registro de desplazamiento 208 (Registro de desplazamiento de salida), que pertenece a la salida lógica 104 y que actúa como conversor serie-paralelo. El registro de desplazamiento 208 genera palabras paralelas que tienen una longitud MAXCOUNT, por ejemplo de 32 bits. Estas palabras son transferidas a un registro de salida 210 que puede cargar una nueva palabra cuando un quinto contador 209 (contador de datos de salida) reconoce que MAXCOUNT bits se han cargado en el registro de desplazamiento 208. Entonces la palabra almacenada en el registro de salida 210 se enviará a la salida 213.

La salida 213 ya podría ser la salida del generador. Sin embargo, con el circuito descrito más abajo, la secuencia de bits generada por el PRNG 102 aún no es completamente una secuencia aleatoria entre dos perturbaciones consecutivas del comportamiento del PRNG 102. De hecho, entre dos perturbaciones, la salida del PRNG sigue siendo determinista y por lo tanto, observando la salida actual y conociendo la arquitectura del PRNG, aún sería posible, teóricamente, predecir el futuro de la secuencia de bits. Este problema se ha resuelto utilizando un circuito mezclador 211 que mezcla de forma aleatoria los bits MAXCOUNT de cada palabra de la salida 213 del registro 210.

Tal como se muestra en la figura 3, el mezclador 211 consiste esencialmente en una matriz de conmutación 301, que recibe las palabras paralelas de los bits MAXCOUNT de la salida 213 del registro 210 y está controlada por una señal de selección SEL que hay en la línea 110. La señal SEL representa un número aleatorio ya disponible en la lógica de mezclado 103. SEL podría obtenerse por ejemplo de algunos de los bits presentes en la salida 212 del registro de reciclaje 207. La salida de la matriz de conmutación 301 está conectada a un registro de salida 302 similar al registro 210 y también es habilitada por la salida de contador 209 (la figura 2).

Tal como se muestra en la figura 4, la matriz de conmutación 301 podría ser un árbol binario de n niveles de multiplexores o conmutadores 400. Con n niveles de multiplexores se necesita un selector de señales SEL de 2^n- 1 bits. Los bits de selección se designan por s(i) (i = 0.... 2^n-1) en el dibujo. Cada conmutador 400 está dispuesto para dejar pasar los bits de entrada que no han cambiado o para intercambiar mutuamente las dos mitades del grupo de bits de entrada dependiendo del valor de selección del bit. Como los bits de la señal de selección SEL son bits aleatorios, cada palabra de entrada se mezcla aleatóriamente produciendo de ese modo otra palabra cuya simple observación de algunos bits no permite predecir los futuros. A modo de ejemplo, en el caso de palabras de entrada de 32 bits en la matriz de conmutación 301, n podría ser igual a 3.

A partir de la descripción anterior el experto en la materia apreciará inmediatamente que todo el generador puede implementarse fácilmente en un circuito integrado, por ejemplo un ASIC o un FPL/FPGA.

El experto en la materia también puede apreciar que el circuito descrito, aparte de ser extremadamente seguro, también es muy flexible. Efectivamente, todos los parámetros funcionales (las velocidades de los relojes, la longitud LFSR, los valores iniciales de los contadores 205, 206) se pueden programar y adaptar a la aplicación específica.

El generador de números aleatorios de alta velocidad de la presente invención funciona según un procedimiento de generación de números aleatorios en el cual los números aleatorios son generados alterando una secuencia pseudo-aleatoria mediante números aleatorios verdaderos que forman semillas aleatorias para la generación de la secuencia pseudo-aleatoria.

Tal procedimiento comprende las etapas de:

- obtener las semillas aleatorias de una secuencia aleatoria de bits que tienen tasa variable;

- procesar una semilla aleatoria para generar una señal de modificación TC1 que explota el tiempo de llegada aleatorio de los bits de la secuencia de bits; y

- cambiar la secuencia pseudo-aleatoria mediante la señal de modificación TC1 en instantes aleatorios entre la llegada de dos semillas consecutivas, obteniendo de ese modo en el intervalo una secuencia pseudo-aleatoria múltiple de longitudes aleatorias más cortas que las longitudes determinadas por la llegada de dos semillas consecutivas, siendo generada la señal de modificación TC1 bajo el control de la secuencia pseudo-aleatoria.

La señal de modificación TC1 se genera al final de una primera cuenta atrás que empieza desde un primer número aleatorio representado por un grupo de bits variable aleatóriamente que son parte de una semilla recibida obtenida rotando la semilla en una cantidad indicada por un segundo número aleatorio representado por un grupo de bits de la secuencia pseudo-aleatoria. La primera cuenta empieza siempre que se genera una nueva semilla y vuelve a empezar, entre la llegada de dos bits consecutivos, siempre que la propia cuenta atrás termine. El segundo número aleatorio es el valor inicial de la segunda cuanta atrás que empieza siempre que acaba la primera cuenta atrás, de modo que el final de dicha segunda cuanta atrás finaliza dicha rotación de semillas.

La secuencia pseudo-aleatoria es generada por un registro de desplazamiento 102 de retroalimentación lineal, y la señal de modificación TC1 es transferida a la lógica de retroalimentación 204b del registro de desplazamiento 102 de retroalimentación lineal.

Entonces la secuencia pseudo-aleatoria modificada es puesta en paralelo para crear palabras de longitud deseada MAX COUNT, y entonces se realiza un mezclado aleatorio de dichas palabras.

La función de mezclado está controlada por una señal de selección aleatoria obtenida de los bits utilizados para el primer número aleatorio.

Para obtener una tasa de bits de salida independiente de la tasa de la secuencia de bits aleatoria, el procedimiento comprende además una etapa de generación, que empieza a partir de una señal del primer reloj que da cadencia a la generación de semillas, de la señal de un segundo reloj que da cadencia a la generación de la secuencia pseudo-aleatoria y de la señal de modificación (TC1), para la puesta en paralelo de las palabras de salida y para la función de mezclado.

Obviamente, la descripción anterior se ha ofrecido a modo de ejemplo no limitativo. Por ejemplo, aunque la realización descrita utiliza un registro de desplazamiento de retroalimentación lineal como PRNG 102, se podrían utilizar otras implementaciones para el generador pseudo-aleatorio, por ejemplo, generadores congruentes o de suma con acarreo (AWC, siglas de *Add With Carry*), o de resta con adeudo (SWB, siglas de *Subtract With Borrow*) y generadores de multiplicación con acarreo (MWC siglas de *Multiply With Carry*). Está al alcance del experto en la materia implementar una interacción entre la lógica de mezclado 103 y el generador pseudo-aleatorio 102 como el descrito en cualquiera de estos generadores. Hay que destacar, que cuando se usa un generador pseudo-aleatorio diferente de un LFSR, la salida lógica 104 podría recibir la salida del PRNG 102 mediante la lógica de mezclado 103.

Aunque la invención se ha descrito principalmente haciendo referencia a una implementación en hardware, es posible implementarla, parcial o totalmente, mediante un ordenador o cualquier otro hardware programable adecuado con un código adaptado para realizar algunas o todas las etapas del procedimiento. En particular, la función de generador de números aleatorios verdaderos puede ser implementada por una parte del mismo ordenador o hardware, o mediante componentes hardware adicionales adecuados.

Referencias citadas en la descripción

Esta lista de referencias citadas por el solicitante es sólo para conveniencia del lector. No forma parte del documento de Patente Europea. Aunque se haya tenido un gran cuidado en recoger las referencias, no puede excluirse la presencia de errores u omisiones y por ello la EPO declina cualquier responsabilidad a este respecto.

Documentos de patentes citados en la descripción

- US 20040019619 A [0006]
- US 6408317 B [0010]
- US 20030059046 A [0011]
- US 20020156819 A [0012]
- WO 0167231 A [0023].

Bibliografía no correspondiente a patentes citada en la descripción

• R. **DAVIES**. Hardware Random Number Generators. 15th Australian Statistics Conference, July 2000, 14 October 2000, http://www.robertnz.net [0004]

• B. **SCHNEIER**. Applied Cryptography: Protocols, Algorithms and Source Code in C. J. Wiley & Sons, Inc, 1996 [0005].

REIVINDICACIONES

1. Generador de números aleatorios (100), que comprende:

- un generador de números aleatorios verdaderos (101);

- un generador de números pseudo-aleatorios (102), dispuestos para generar una secuencia pseudo-aleatoria utilizando los números aleatorios verdaderos producidos por dicho generador de números aleatorios verdaderos (101) como semillas aleatorias; y - una lógica de mezclado (103) conectada entre dicho generador de números aleatorios verdaderos (101) y dicho generador de números pseudo-aleatorios (102) y dispuesta para modificar el comportamiento de dicho generador de números pseudo-aleatorios (102) utilizando la semilla aleatoria, **caracterizado** por el hecho de que dicho generador aleatorio verdadero (101) está dispuesto para generar una secuencia aleatoria de tasa de bits variable, y por el hecho de que dicha lógica de mezclado (103) comprende un generador (205, 206, 207) de una señal de modificación (TC1) destinada a cambiar el comportamiento de dicho generador de números pseudo-aleatorios (102) en múltiples instantes aleatorios en el intervalo entre dos semillas consecutivas, de modo que se obtiene en dicho intervalo unas secuencias pseudo-aleatorias múltiples de longitudes aleatorias más cortas que la longitud aleatoria determinada por la llegada de dos semillas consecutivas, estando dicho generador (205, 206, 207) de la señal de modificación conectado para recibir dicha semilla y generar dicha señal de modificación (TC1) procesando dicha semilla mediante la secuencia generada por dicho generador de números pseudo-aleatorios (102).

2. Generador de números aleatorios (100) según la reivindicación 1, en el que dicho generador (205, 206, 207) de la señal de modificación (TC1) comprende:

- un primer contador (205) de cuenta atrás adaptado para contar hacia atrás desde un primer número aleatorio, representado por un primer grupo de bits que son parte de una versión rotada aleatóriamente de la semilla recibida por dicho generador de la señal de modificación (205, 206, 207), cargando dicho primer contador (205) dicho primer número aleatorio y empezando su cuenta atrás siempre que haya una semilla disponible y, entre la llegada de dos semillas consecutivas, cada vez que genere una señal de fin de cuenta, siendo dicha señal de fin de cuenta suministrada a dicho generador de números pseudo-aleatorios (102) como señal de modificación (TC1);

- un segundo contador de cuenta atrás (206), adaptado para contar hacia atrás desde un segundo número aleatorio representado por un grupo de bits de la secuencia generada por dicho generador de números pseudo-aleatorios (102) y dispuesto para cargar un nuevo valor de dicho segundo número aleatorio y para empezar de nuevo su cuenta atrás siempre que dicho primer contador de cuenta atrás genere su señal de fin de cuenta (TC1);

- un registro de desplazamiento de recirculación (207), que recibe las semillas y las suministra a dicho primer contador de cuenta atrás (205) con dicho primer número aleatorio, y que está dispuesto para generar dicha versión rotada aleatóriamente de la semilla, en los intervalos entre las llegadas de dos semillas consecutivas, rotando los bits de la semilla en una cantidad determinada por el valor de dicho segundo número aleatorio.

3. Generador de números aleatorios (100) según cualquiera de las reivindicaciones 1 ó 2, **caracterizado** por el hecho de que dicho generador pseudo-aleatorio (102) es un registro de desplazamiento de retroalimentación lineal, y dicho generador de la señal de modificación (205, 206, 207) suministra dicha señal de modificación (TC1) a la lógica de retroalimentación (204b) de dicho registro de desplazamiento de retroalimentación lineal.

4. Generador de números aleatorios (100) según cualquiera de las reivindicaciones anteriores, **caracterizado** por el hecho de que dicha lógica de mezclado (103) comprende además unos circuitos de entrada (201, 202) adaptados para recibir la secuencia aleatoria de bits generada por dicho generador aleatorio verdadero (101), para construir dicha semilla poniendo en paralelo los bits de dicha secuencia aleatoria y para generar una señal (Datos Válidos) que indica la disponibilidad de una semilla.

5. Generador de números aleatorios (100) según cualquiera de las reivindicaciones 2 a 4, **caracterizado** por el hecho de que dicho registro de desplazamiento de recirculación (207) está adaptado para cargar directamente una semilla, siempre que reciba dicha señal (Datos Válidos) que indica la disponibilidad de la semilla, y dicho generador pseudo-aleatorio (102) está adaptado para cargar una nueva semilla según las órdenes de dicho primer contador (205), cada vez que este último reciba dicha señal (Datos Válidos) que indica la disponibilidad de la semilla.

6. Generador de números aleatorios (100) según cualquiera de las reivindicaciones anteriores, **caracterizado** por el hecho de que dicho de que los circuitos de entrada (201, 202) comprenden un generador de señales de reloj (203) para generar, a partir de la señal de un primer reloj (CLK) que da cadencia a las funciones de dichos circuitos de entrada (201, 202), una señal de un segundo reloj, para dar cadencia a dicho generador pseudo-aleatorio (102) y dicho generador de la señal de modificación (205, 206, 207), por lo cual la tasa de bits de salida del generador de números aleatorios (100) es independiente de la tasa de bits de la secuencia aleatoria suministrada por el generador aleatorio verdadero (101).

7. Generador de números aleatorios (100) según cualquiera de las reivindicaciones anteriores, **caracterizado** por el hecho de que además comprende una salida lógica (104) para poner en paralelo la secuencia pseudo-aleatoria modificada y construir palabras de una determinada longitud, comprendiendo dicha salida lógica (104) un mezclador (211) para mezclar aleatóriamente los bits en cada palabra.

8. Generador de números aleatorios (100) según la reivindicación 7, **caracterizado** por el hecho de que dicho mezclador (211) está controlado por una señal de selección aleatoria (SEL) transferida por dicho generador (205, 206, 207) de la señal de modificación.

9. Generador de números aleatorios (100) según las reivindicaciones 2 y 8, **caracterizado** por el hecho de que dicho la señal de selección aleatoria es transferida por dicho registro de desplazamiento de recirculación (207).

10. Generador de números aleatorios según cualquiera de las reivindicaciones 7 a 9, **caracterizado** por el hecho de que dicho circuito mezclador (211) comprende una matriz de conmutación (301) compuesta de un árbol binario de n niveles de conmutadores (400), cada uno controlado por un bit respectivo de dicha señal de selección aleatoria para mezclar o dejar pasar sus bits de entrada sin cambios.

11. Generador de números aleatorios según cualquiera de las reivindicaciones 7 a 9, **caracterizado** por el hecho de que está implementado en un circuito integrado.

12. Procedimiento de generación de números aleatorios, en el que dichos números aleatorios son generados alterando una secuencia pseudo-aleatoria mediante números aleatorios verdaderos que forman semillas aleatorias para la generación de dicha secuencia pseudo-aleatoria, **caracterizado** por el hecho de que comprende las etapas de:

- obtener las semillas aleatorias de una secuencia aleatoria de tasa de bits variable;
- procesar una semilla aleatoria para generar una señal de modificación (TC1) que explota el tiempo de llegada aleatorio de los bits de dicha secuencia de bits; y
- cambiar la secuencia pseudo-aleatoria por dicha señal de modificación (TC1) en instantes aleatorios entre la llegada de dos semillas consecutivas, por lo cual se obtiene en dicho intervalo unas secuencia pseudo-aleatorias múltiples de longitudes aleatorias más cortas que las longitudes determinadas por la llegada de dos semillas consecutivas, siendo dicha señal de modificación (TC1) generada bajo el control de la secuencia pseudo-aleatoria.

13. Procedimiento según la reivindicación 12, **caracterizado** por el hecho de que dicha señal de modificación (TC1) es generada al final de una primera cuenta atrás que empieza desde un primer número aleatorio representado por un grupo de bits aleatóriamente variable que son parte de una versión rotada de una semilla recibida obtenida rotando la semilla en una cantidad indicada por un segundo número aleatorio representado por un grupo de bits de la secuencia pseudo-aleatoria, empezando la primera cuenta atrás siempre que una semilla sea generada y volviendo a empezar, entre la llegada de dos bits consecutivos, siempre que la propia cuenta atrás acabe; y por el hecho de que dicho segundo número aleatorio es el valor de inicio de una segunda cuenta atrás que empieza siempre que la primera cuanta atrás acabe, parando dicha segunda cuenta atrás dicha rotación de la semilla.

14. Procedimiento según cualquiera de las reivindicaciones 12 ó 13, **caracterizado** por el hecho de que dicha secuencia pseudo-aleatoria está generada por un registro de desplazamiento de retroalimentación lineal (102), y dicha señal de modificación (TC1) es transferida a la lógica de retroalimentación (204b) de dicho registro de desplazamiento de retroalimentación lineal (102).

15. Procedimiento según cualquiera de las reivindicaciones 12 a 14, en el que la secuencia pseudo-aleatoria modificada es puesta en paralelo para crear palabras de longitud deseada (MAX COUNT), **caracterizado** por el hecho de que el procedimiento comprende además un mezclado aleatorio de dichas palabras.

16. Procedimiento según la reivindicación 15, **caracterizado** por el hecho de que dicho mezclado está controlado por una señal de selección aleatoria obtenida de los bits usados para formar dicho primer número aleatorio.

17. Procedimiento según cualquiera de las reivindicaciones 12 a 14, **caracterizado** por el hecho de que comprende además la etapa de generar, a partir de una señal de un primer reloj que da cadencia a la generación de semillas, una señal de un segundo reloj, para dar cadencia a la generación de dicha secuencia pseudo-aleatoria y de dicha señal de modificación (TC1), la puesta en paralelo de las palabras de salida y el mezclado, por lo cual se obtiene una tasa de bits de salida independiente de la tasa de bits de la secuencia aleatoria.

18. Producto de tipo programa informático que se puede cargar en la memoria de al menos un ordenador y que incluye partes de código de *software* para realizar el procedimiento según cualquiera de las reivindicaciones 12 a 17.

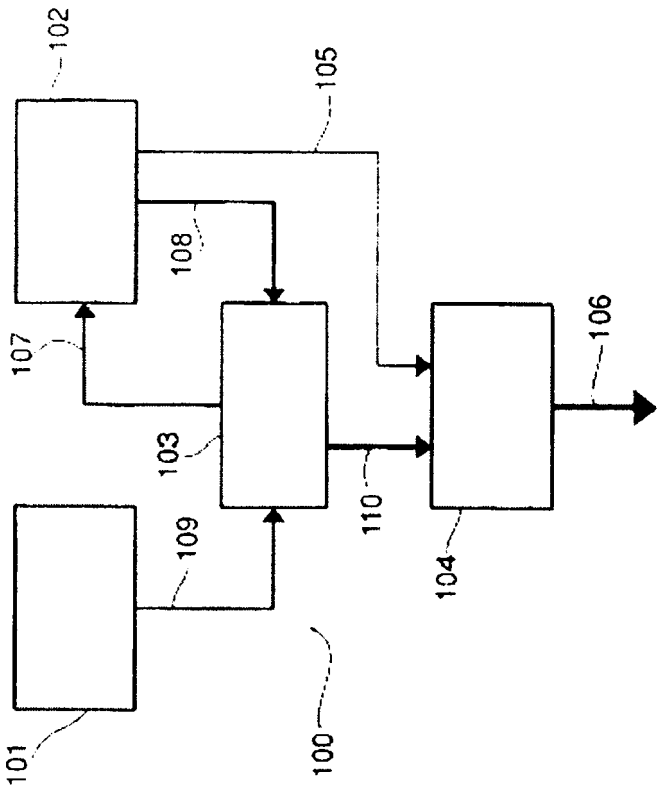

Fig. 1

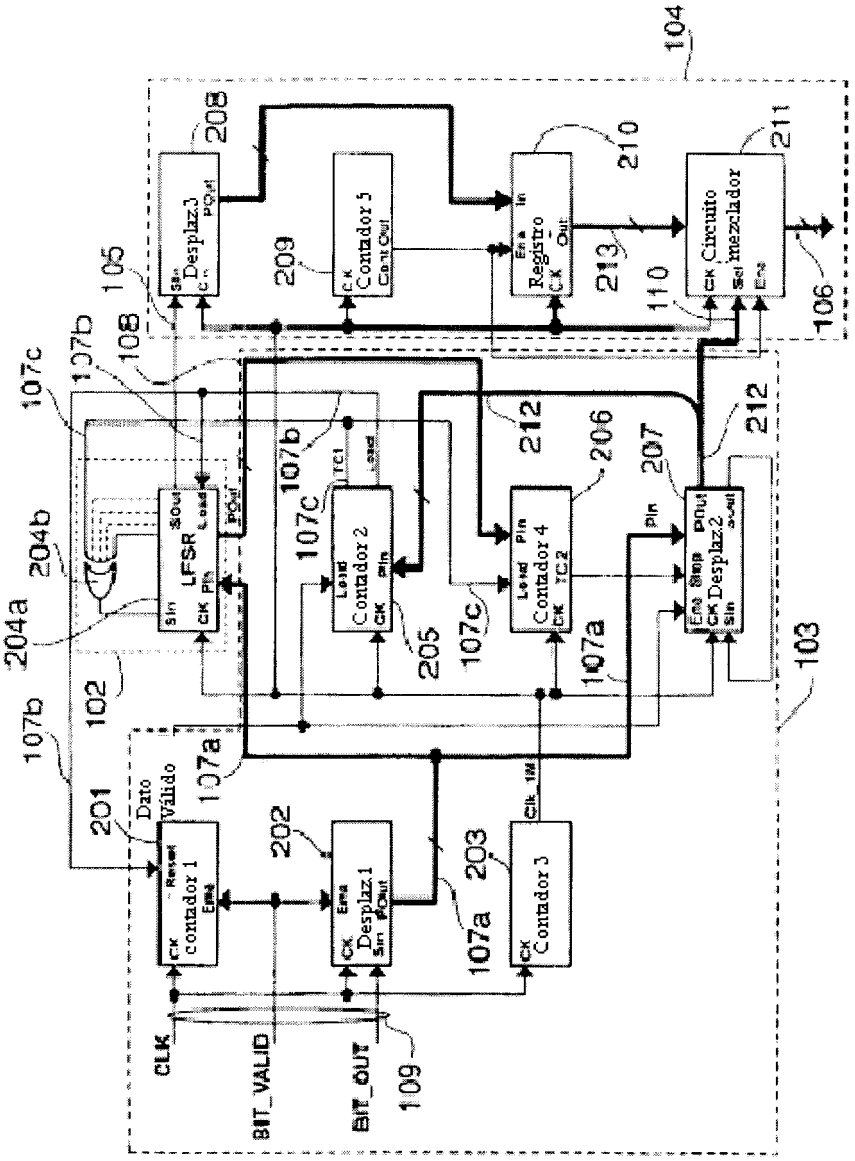
104
208
Desplaz.3
209
Contador 5
210
Registro
211
Circuito mezclador
213
110
106
105
108
107b
107c
212
206
207
LFSR
204b
204a
Contador 2
Contador 4
Desplaz.2
205
107a
102
103
201
Dato Válido
Contador 1
202
Desplaz.1
203
Contador 3
CLK
BIT_VALID
BIT_OUT
109

Fig. 2

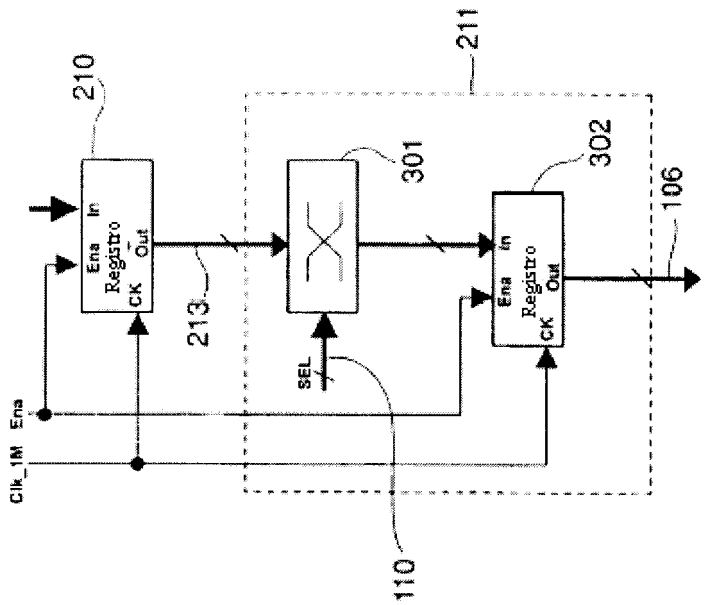
211
210
301
302
106
213
SEL
Ena
Clk_1M
110
Ena
In
Registro
CK
Out

Fig. 3

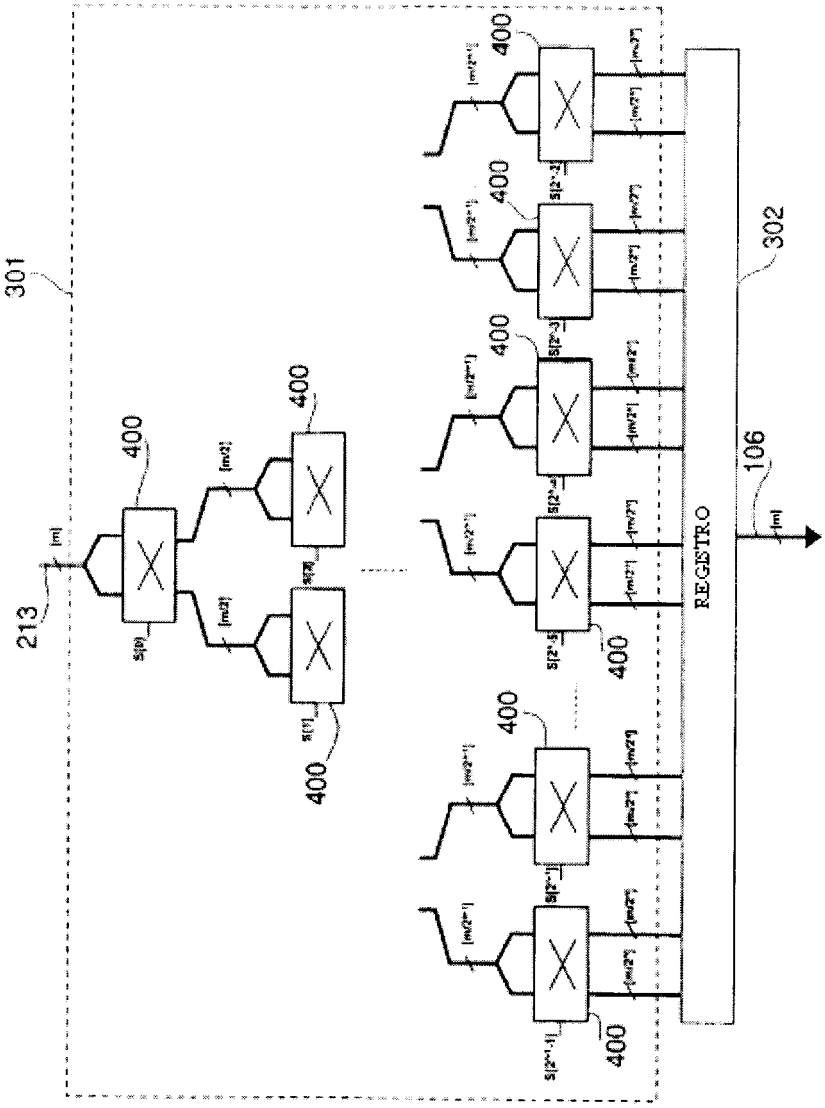

Fig. 4