(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 557 008**

(51) Int. CI.:

G06F 11/16 (2006.01)
G06F 21/55 (2013.01)
G06F 21/79 (2013.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **29.11.2012 E 12791783 (9)**

(97) Fecha y número de publicación de la concesión europea: **14.10.2015 EP 2786253**

(54) Título: **Procedimiento de detección de un error de lectura de un dato**

(30) Prioridad:

01.12.2011 FR 1161029

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
21.01.2016

(73) Titular/es:

VIACCESS (100.0%)
Les Collines de l'Arche Tour Opéra C 76, Route de la Demi-Lune
92057 Paris La Défense Cedex, FR

(72) Inventor/es:

BARAU, EMMANUEL y
SOQUET, PATRICK

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

DESCRIPCIÓN

Procedimiento de detección de un error de lectura de un dato

El invento se refiere a un procedimiento de detección de un error de lectura de un dato y a un procedimiento de protección de un procesador de seguridad. El invento se refiere igualmente a un soporte de grabación de informaciones así como a un procesador de seguridad para la puesta en práctica de estos procedimientos.

Los procedimientos conocidos detectan un error de lectura realizando las operaciones siguientes:

a) la grabación de una primera copia del dato en una primera zona de una memoria electrónica y la grabación de una segunda copia del dato en una segunda zona de una memoria electrónica,

en respuesta a una solicitud de lectura del dato;

b) la lectura de los valores de la primera y segunda copias del dato respectivamente en la primera y segunda zonas,

c) la comparación de los valores leídos de la primera y segunda copias del dato,

d) si los valores leídos de la primera y segunda copias son idénticos, entonces no es detectado ningún error de lectura de este dato.

Una zona memoria o una zona de memoria es una memoria electrónica o una porción de una memoria electrónica dividida en varios bloques de memorias. Cada bloque de memoria está destinado a contener un dato. Típicamente, un bloque de memoria es una página o un múltiplo entero de una página. Una página es el menor número de octetos que pueden ser escrito en una sola operación de escritura. Así, incluso si el dato grabado en una página tiene un tamaño menor que la página, se considera como que la totalidad de la página está ocupada por este dato y no es posible grabar en esta página un dato suplementario.

El estado de la técnica relativo a estos procedimientos conocidos está por ejemplo divulgado en las solicitudes de patentes siguientes: GB 2 404 261, US 2006 053308, WO 2008 23297, US 2007 0033417, US 2005 0160310, US 2008/140962A1 y US 2009/113546A1.

Los procedimientos conocidos se dirigen únicamente a protegerse de las consecuencias de un fallo de un bloque de memoria. Así, si los valores de la primera y segunda copias son diferentes, entonces un error es señalado y se disparan medidas correctivas. Una de las medidas correctivas clásicas consiste en corregir el valor del dato grabado. Sin embargo, estos procedimientos detectan también un error de lectura del dato. En el caso de un error de lectura de datos, la ausencia de igualdad entre los valores de la primera y segunda copias del dato no proviene de una corrupción de los datos grabados en las zonas de memorias sino de un error durante el proceso de lectura de estos datos. Aquí, se habla de corrupción o de datos corrompidos cuando el valor físicamente grabado de este dato es erróneo a causa de un fallo de un bloque de memoria. Así, aunque los valores grabados de la primera y segunda copias sean perfectamente idénticos, durante la operación c) anterior, se constata que los valores son diferentes. Por ejemplo, un error de lectura puede ser provocado por una perturbación de las señales del bus de lectura o por una corrupción de los datos copiados en una memoria no volátil después de haber sido leídos en una zona de memoria no volátil.

Los procedimientos conocidos no hacen distinción entre estos dos tipos de error y lanzan sistemáticamente, después de la detección de un error, la misma medida correctiva. Típicamente, la medida correctiva consiste en corregir el valor erróneo. Ahora bien, en el caso de un error de lectura, tal corrección es inútil y se traduce por un desperdicio de recursos informáticos tales como del tiempo de un microprocesador que pone en práctica este procedimiento.

En el dominio de los procesadores de seguridad, existen igualmente otra buena razón para distinguir estos dos tipos de error. Un procesador de seguridad es un procesador electrónico que está reforzado para ser lo más resistente posible frente a ataques de piratas informáticos. Es por tanto generalmente utilizado para almacenar datos confidenciales. Ahora bien, un ataque clásico contra un procesador de seguridad consiste en corromper los datos grabados para provocar un comportamiento no previsto del procesador de seguridad que puede revelar la totalidad o parte de los datos confidenciales que contiene. Sin embargo, es más simple para los piratas informáticos provocar un error de lectura que corromper los datos grabados. Cuando el pirata informático provoca voluntariamente errores de lectura utilizando picos de tensión o un haz láser u otros, se dice que el procesador de seguridad es víctima de un ataque en lectura. Así, en el contexto de los procesadores de seguridad, un error de lectura permite detectar de un modo bastante seguro una tentativa de criptoanálisis mientras que un fallo de un bloque de memoria puede ser un fallo accidental causado, por ejemplo, por el envejecimiento.

El invento pretende remediar este inconveniente. Tiene pues por objeto un procedimiento de detección de un error de lectura conforme a la reivindicación 1.

El procedimiento anterior permite distinguir, y por tanto detectar, un error de lectura causado por la corrupción de los datos grabados. En efecto, durante la segunda iteración de las operaciones b) y c) anteriores, los valores de la primera y

segunda copias son necesariamente diferentes si al menos uno de estos valores está corrompido. A la inversa, la identidad de estos valores durante la segunda iteración de las operaciones b) y c) identifica de manera cierta un error de lectura del dato.

A partir de entonces, es posible poner en práctica medidas apropiadas diferentes para responder a un error de lectura o a una corrupción de los datos grabados. Por ejemplo, el hecho de detectar un error de lectura permite evitar poner en práctica inútilmente medidas que pretenden corregir los datos grabados mientras que estos no están corrompidos. Además, en el caso de un procesador de seguridad, la detección de un error del día permite disparar la puesta en práctica de contramedidas para impedir el criptoanálisis.

Los modos de realización de este procedimiento de detección pueden incluir una o varias de las características de las reivindicaciones dependientes.

Los modos de realización del procedimiento de detección anterior presentan además las ventajas siguientes,

- la utilización de un código de corrección de error permite restaurar el funcionamiento del procedimiento de detección de error de lectura incluso después de un fallo de un bloque de memoria,

- la utilización del código de corrección de error permite restaurar el funcionamiento del procedimiento de detección de error de lectura incluso después de que los valores de la primera y segunda copias del dato hayan sido corrompidos limitando al mismo tiempo el número de códigos de corrección de error grabados,

- la elección de bloques de memorias para grabar en ellos un nuevo dato únicamente entre los bloques de memorias no memorizados como defectuosos permite evitar reutilizar bloques de memorias defectuosos para la grabación de datos,

- la grabación de un valor transformado en la primera zona de memoria diferente del valor grabado en la segunda zona de memoria permite aumentar la probabilidad de que los valores leídos para la primera y segunda copias sean diferentes en caso de ataques en lectura.

El invento tiene igualmente por objeto un procedimiento de protección de un procesador de seguridad contra un ataque en lectura conforme a la reivindicación 6.

El invento tiene igualmente por objeto un soporte de grabación de informaciones que incluyen instrucciones para la puesta en práctica del procedimiento de detección o de protección anterior cuando estas instrucciones son ejecutadas por un calculador electrónico.

Finalmente, el invento tiene igualmente por objeto un procesador de seguridad conforme a la reivindicación 8.

Los modos de realización de este procesador de seguridad pueden incluir una o varias de las características de las reivindicaciones dependientes.

El invento será mejor comprendido con la lectura de la descripción que va a seguir, dada únicamente a título de ejemplo no limitativo y hecha con referencia a los dibujos en los que:

La fig. 1 es una ilustración esquemática de un terminal de recepción de contenidos multimedia enredados asociado a un procesador de seguridad.

La fig. 2 es una ilustración esquemática de una organización de una memoria del procesador de seguridad de la fig. 1.

La fig. 3 es un organigrama de una fase de escritura de datos en la memoria de la fig. 2.

La fig. 4 es un organigrama de un procedimiento de protección del procesador de seguridad de la fig. 1.

La fig. 5 es un organigrama de otro modo de realización de un procedimiento de protección del procesador de seguridad de la fig. 1.

En estas figuras las mismas referencias son utilizadas para designar los mismos elementos.

En la continuación de esta descripción, las características y funciones bien conocidas por el experto en la técnica no se han descrito en detalle. Además, la terminología utilizada es la de los sistemas de acceso condicionales a contenidos multimedia. Para más informaciones sobre esta terminología, el lector puede dirigirse al documento siguiente:

« Functional Model of Conditional Access System », EBU Review, Technical European Broadcasting Union, Brussels, BE, nº 266, 21 de diciembre de 1995.

El invento se aplica en particular al dominio del control de acceso para el suministro de contenidos multimedia de pago tales como la televisión de pago.

Es conocido el hecho de difundir varios contenidos multimedia al mismo tiempo. Para ello, cada contenido multimedia es

difundido sobre su propio canal o « cadena ». Un canal corresponde típicamente a una cadena de televisión.

En esta descripción, se designa más específicamente por « contenido multimedia » un contenido de audio y/o visual destinado a ser restituido en una forma directamente perceptible y comprensible por un ser humano. Típicamente, un contenido multimedia corresponde a una sucesión de imágenes que forman una película, una emisión de televisión o publicidad. Un contenido multimedia puede igualmente ser un contenido interactivo tal como un juego.

Para asegurar y someter la visualización de los contenidos multimedia a ciertas condiciones, como la suscripción de un abono de pago por ejemplo, los contenidos multimedia son difundidos en forma enredada y no libre.

Más precisamente, cada contenido multimedia es dividido en una sucesión de criptoperíodos. Durante la duración de un criptoperíodo, las condiciones de acceso al contenido multimedia enredado permanecen sin cambios. En particular, durante un criptoperíodo, el contenido multimedia es enredado con la misma palabra de control. Generalmente, la palabra de control varía de un criptoperíodo al otro.

Además, la palabra de control es generalmente específica de un contenido multimedia, siendo este último extraído aleatoria o pseudo-aleatoriamente.

Aquí los términos « enredar » y « cifrar » son considerados como sinónimos. Lo mismo sucede para los términos « desenredar » y « descifrar ».

El contenido multimedia libre corresponde al contenido multimedia antes de que éste sea enredado. Esto puede ser hecho directamente comprensible por un ser humano sin haber recurrido a operaciones de desenredado y sin que su visualización sea sometida a ciertas condiciones.

Las palabras de control necesarios para desenredar los contenidos multimedia son transmitidas de manera sincronizada con los contenidos multimedia. Para ello, por ejemplo, las palabras de control son multiplexadas con el contenido multimedia enredado.

Para asegurar la transmisión de las palabras de control, éstas son transmitidas a los terminales en forma de criptogramas contenidos en mensajes ECM (Entitlement Control Message). Se designa aquí por « criptograma » una información insuficiente por sí sola para encontrar la palabra de control libre. Así, si la transmisión de la palabra de control es interceptada, tan sólo el conocimiento del criptograma de la palabra de control no permite encontrar la palabra de control que permita desenredar el contenido multimedia.

Para encontrar la palabra de control libre, es decir la palabra de control que permite desenredar directamente el contenido multimedia, ésta debe ser combinada con una información secreta. Por ejemplo, el criptograma de la palabra de control es obtenido cifrando la palabra de control libre con una clave criptográfica. En este caso, la información secreta es la clave criptográfica que permite descifrar el criptograma. El criptograma de la palabra de control puede también ser una referencia a una palabra de control almacenada en una tabla que contiene una multitud de palabras de control posibles. En este caso, la información secreta es la tabla que asocia a cada referencia una palabra de control libre.

La información secreta debe ser preservada en lugar seguro. Para ello, se ha propuesto ya almacenar la información secreta en procesadores de seguridad tales como tarjetas con chip directamente conectadas a cada uno de los terminales.

La fig. 1 representa un terminal 8 destinado a ser utilizado en tal sistema de control de acceso condicional. El terminal 8 desenreda un canal para presentar libre sobre un dispositivo de presentación.

El terminal 8 comprende un receptor 10 de contenidos multimedia difundidos. Este receptor 10 está conectado a la entrada de un desmultiplexor 12 que transmite el contenido multimedia a un dispositivo de desenredar 14 y los mensajes ECM y EMM (Entitlement Management Message) a un procesador de seguridad 16.

El dispositivo de desenredar 14 desenreda el contenido multimedia enredado a partir de la palabra de control transmitida por el procesador 16. El contenido multimedia desenredado es transmitido a un descodificador 18 que lo descodifica. El contenido multimedia descomprimido o descodificado es transmitido a una tarjeta gráfica 20 que pilota la presentación de este contenido multimedia sobre un dispositivo de presentación 22 equipado con una pantalla 24. El dispositivo de presentación 22 presenta en abierto el contenido multimedia sobre la pantalla 24. Por ejemplo, el dispositivo de presentación 22 es una televisión, un ordenador o un teléfono fijo o móvil. Aquí, el dispositivo de presentación 22 es una televisión.

Típicamente, la interfaz entre el terminal 8 y el procesador 16 comprende un lector 26 gestionado por un módulo 28 de control de acceso. Aquí, el lector 26 es un lector de tarjetas con chip. El módulo 28 gestiona en particular:

- la transmisión de los mensajes ECM y EMM desmultiplexados al procesador 16, y

- la recepción de las palabras de control descifradas por el procesador 16 y su transmisión al dispositivo para desenredar

14.

El procesador 16 trata informaciones confidenciales tales como claves criptográficas o títulos de acceso a los contenidos multimedia. Para preservar la confidencialidad de estas informaciones, está concebido ser lo más robusto posible frente a tentativas de ataques realizados por piratas informáticos. Es por tanto más robusto frente a estos ataques que los otros componentes del terminal 8. En particular, las memorias que incluye son únicamente accesibles y utilizadas por este procesador 16. Aquí, el procesador 16 es el procesador de seguridad de una tarjeta con chip 30.

El procesador 16 comprende en particular un calculador electrónico 32 programable conectado por medio de un bus 34 de transmisión de informaciones a una memoria electrónica volátil 36 y a una memoria electrónica no volátil 38.

La memoria 36 es típicamente conocida bajo el acrónimo de RAM (Random Access Memory) ("Memoria de Acceso Aleatorio"). La memoria 38 conserva los datos que son grabados incluso en ausencia de alimentación del procesador 16. Además la memoria 38 es una memoria sobre la que se puede reescribir. Típicamente, se trata de una EEPROM (Electrically Erasable Programmable Read Only Memory) ("Memoria de Solo Lectura Programable y que se Puede Borrar Eléctricamente") o de una memoria flash. La memoria 38 contienen informaciones confidenciales necesarias para el desenredado de los contenidos multimedia. Aquí, contiene también las instrucciones necesarias para la ejecución del procedimiento de la fig. 4 ó 5.

La fig. 2 representa más en detalle diferentes zonas de la memoria 38. Aquí, la memoria 38 comprende las zonas de memoria siguientes:

- una zona 42 de control,
- una primera zona 44 de grabación de datos, y
- una segunda zona 46 de grabación de datos.

Cada zona de memoria está definida por una dirección de comienzo y una dirección de final. Aquí cada zona ocupa una región de direcciones contiguas en la memoria 38. Cada zona está dividida en varios bloques de memorias. Por ejemplo, las zonas 42, 44 y 46 contienen cada una más de 100 o 400 bloques de memorias. El tamaño de cada bloque de memoria es de varios octetos. Aquí, los bloques de memorias de estas zonas tienen todos el mismo tamaño. Por ejemplo, este tamaño es superior o igual a 64o o 128o. Los tamaños de las zonas 44 y 46 son idénticos.

La zona 44 está dividida en P bloques de memorias B_{1j} del mismo tamaño donde el índice j identifica la posición del comienzo del bloque B_{1j} con relación a la dirección de comienzo de la zona 44.

La zona 46 está igualmente dividida en P bloques de memorias B_{2j} del mismo tamaño de los bloques B_{1j}. El índice j identifica la posición del comienzo del bloque B_{2j} con relación a la dirección de comienzo de la zona 46. Aquí, para simplificar la realización, el desplazamiento entre el comienzo del bloque B_{2j} y la dirección de comienzo de la zona 46 es idéntico al desplazamiento que existe entre el comienzo del bloque B_{1j} y la dirección del comienzo de la zona 44. En estas condiciones, los bloques B_{1j} y B_{2j} son llamados « emparejados ».

La zona 42 comprende 2P bloques de memorias CB_{ij} de los mismos tamaños. Cada bloque CB_{ij} está asociado a un bloque B_{ij} de la zona 44 ó 46. El índice i toma el valor « 1 » para identificar la zona 44 y el valor « 2 » para identificar la zona 46.

Aquí, cada bloque CB_{ij} comprende en particular las informaciones siguientes:

- la indicación de si el bloque B_{ij} asociado está libre, es decir que puede ser utilizado para grabar un nuevo dato,
- eventualmente, la indicación del bloque de memoria siguiente unido a este bloque B_{ij} o la indicación de que este bloque B_{ij} es el último bloque de una cadena de bloques unidos,
- un código CD_{ij} de detección de error,
- una máscara MR_{ij} de recubrimiento, y
- una marca MD_{ij} que indica si el bloque B_{ij} es defectuoso o no.

El código CD_{ij} es construido únicamente a partir del valor del dato obtenido en el bloque B_{ij}. Este código CD_{ij} añade suficiente redundancia al dato contenido en el bloque B_{ij} para que sea posible detectar uno o varios bits erróneos en el valor del dato grabado en este bloque.

Por ejemplo, el código CD_{ij} es un control de redundancia cíclica más conocido bajo el acrónimo CRC (Cyclic Redundancy Check) ("Comprobación de Redundancia Cíclica"). Por ejemplo el código CD_{ij} es un CRC 32.

La máscara MR_{ij} de recubrimiento es un valor utilizado para transformar de manera irreversible el valor D_{ij} del dato

grabado en el bloque B_{ij} en un valor transformado D'_{ij} que está grabado en este bloque B_{ij}. Esta transformación es reversible de manera que, a partir del valor de la máscara dMR_{ij} y del valor transformado D'_{ij}, es posible encontrar el valor D_{ij} del dato.

La marca MD_{ij} permiten memorizar si el bloque B_{ij} es defectuoso o no. Un bloque B_{ij} defectuoso es por ejemplo un bloque de memoria que incluye bits de informaciones cuyos valores no pueden ya ser reinscritos o modificados, lo que entraña la aparición de errores en el valor grabado en este bloque de memoria.

La zona 42 comprende igualmente códigos correctores de error CC_0, CC_1 y CC_2. El código CC_0 es construido a partir del contenido de las zonas 44 y 46. Añade suficiente redundancia al contenido memorizado en las zonas 44 y 46 para que sea posible no solamente detectar sino igualmente por recibir uno o varios bits erróneos de los datos grabados en estas zonas 44 y 46. De manera similar, los códigos CC_1 y CC_2 añaden suficiente redundancia al contenido, respectivamente, de las zonas 44 y 46, para permitir corregir k bits erróneos en las zonas, respectivamente, 44 y 46, donde k es un número entero natural superior o igual a uno y, de preferencia, superior o igual a cinco o diez. Contrariamente al código CC_0 los códigos CC_1 y CC_2 permiten únicamente corregir bits erróneos, respectivamente, en las zonas 44 y 46. Por ejemplo, estos códigos de corrección de error son códigos de Reed Solomon.

La fig. 3 representa una fase 50 de grabación de un dato confidencial en la memoria 38. El dato confidencial es típicamente una clave criptográfica para descifrar palabras de control o títulos de acceso que autorizan o no el acceso y el desenredado de contenidos multimedia.

Si el tamaño del dato es superior al tamaño de un bloque de memoria, entonces éste es dividido en primer lugar en varias porciones de tamaño inferior cada una al tamaño de un bloque de memoria para llegar al caso de un dato cuyo tamaño es inferior al tamaño de un bloque de memoria. En este caso, las diferentes porciones del mismo dato son por ejemplo encadenadas unas con otras indicando en cada bloque BC_{ij} la dirección del bloque de memoria siguiente.

Inicialmente, durante una operación 52, el calculador 32 elige entre los diferentes bloques de memorias de las zonas 44 y 46 un par de bloques de memorias B_{ij} que satisfagan las condiciones siguientes:

- los bloques B_{1j} y B_{2j} están emparejados,
- los bloques B_{1j} y B_{2j} son libres, y
- los bloques B_{1j} y B_{2j} no están marcados como defectuosos.

El calculado 32 verifica si los bloques B_{1j} y B_{2j} elegidos son libres y no defectuosos a partir de las informaciones contenidas en los bloques BC_{1j} y BC_{2j} de la zona de control 42.

A continuación, el valor del dato a grabar en los bloques B_{1j} y B_{2j} es denominado, respectivamente, D_{1j} y D_{2j}. Estos valores son idénticos.

Durante una operación 54, el calculador 32 calcula el nuevo valor de los códigos CD_{1j} y CD_{2j} que permiten detectar un error, respectivamente, en los valores D_{1j} y D_{2j}. Durante esta operación 54, el calculador 32 calcula igualmente los nuevos valores de los códigos CC_0, CC_1 y CC_2 de corrección de error y los graba en la zona 42.

Los nuevos valores de los códigos CD_{1j} y CD_{2j} son grabados, respectivamente, en los bloques BC_{1j} y BC_{2j}.

El calculador 32 registra igualmente en sus bloques BC_{1j} y BC_{2j} una indicación según la cual los bloques de memorias B_{1j} y B_{2j} ya no están libres.

A continuación, durante una operación 56, los valores D_{1j} y D_{2j} son transformados, respectivamente, en valor D'_{1j} y D'_{2j} en función del valor de las máscaras, respectivamente, MR_{1j} y MR_{2j}. El valor de las máscaras MR_{1j} y MR_{2j} está contenido en los bloques BC_{1j} y BC_{2j}. Los valores de las máscaras MR_{1j} y MR_{2j} son diferentes de manera que los valores transformados D'_{1j} y D'_{2j} sean diferentes.

Por ejemplo, la transformación es realizada con ayuda de la relación siguiente:

$D'_{ij} = D_{ij} \oplus MR_{ij}$, donde « $\oplus$ » es la operación XOR.

A continuación, durante una operación 58, los valores D'_{1j} y D'_{2j} son físicamente grabados, respectivamente, en los bloques de memorias B_{1j} y B_{2j}. La fase 50 se termina entonces.

Durante su utilización, el procesador 16 ejecuta un programa por ejemplo para descifrar palabras de control. Durante la ejecución de este programa, instrucciones requieren la lectura de un dato grabado en la memoria 38, tal como una clave criptográfica o un título de acceso. El procedimiento de la fig. 4 es entonces ejecutado.

Inicialmente, durante una operación 66, la dirección del bloque de memoria a leer es grabada en un registro no volátil de dirección de lectura. Por ejemplo, este registro está contenido en la memoria 38.

A continuación, se procede a una operación 68 de lectura del dato en la memoria 38 en la dirección especificada. Más precisamente, durante una operación 70, los valores D'_{1j} y D'_{2j} contenidos en los bloques de memorias emparejados, respectivamente, B_{1j} y B_{2j} son leídos.

A continuación, durante una operación 72, el calculador 32 aplica la transformación inversa a la aplicada durante la operación 56 del procedimiento de la fig. 3. Para ello, utiliza los valores de las máscaras MR_{1j} y MR_{2j}. los valores obtenidos por aplicación de esta transformación inversa para los valores D'_{1j} y D'_{2j} son denominados en lo que sigue, respectivamente, valores D_{1j} y D_{2j}. Se observará que en caso de error de lectura o de corrupción de los datos grabados, los valores D_{1j} y D_{2j} no son necesariamente idénticos a los valores D_{1j} y D_{2j} grabados durante la fase 50 de escritura.

Durante una operación 74, los valores D_{1j} y D_{2j} leídos son comparados. Si estos valores son iguales, entonces se procede a una operación 76 durante la cual el programa ejecutado por el procesador 16 trata el valor D_{1j} y prosigue su ejecución normal. Por ejemplo, el calculador 32 descifra una palabra de control con ayuda del valor D_{1j}. Durante la operación 76, no se ha detectado ningún error de lectura. Además, durante la operación 76, el registro de dirección de lectura es borrado.

A la inversa, si los valores D_{1j} y D_{2j} leídos no son idénticos, entonces el programa en curso de ejecución es interrumpido y una rutina de verificación es ejecutada por el calculador 32. Por ejemplo, el procesador de seguridad es reinicializado y durante la nueva puesta en marcha del procesador de seguridad, la rutina de verificación es sistemáticamente ejecutada si el registro de dirección de lectura no está vacío. La rutina de verificación puede igualmente ser lanzada para una desviación o confusión sobre error de la ejecución del programa.

Una vez lanzada esta rutina de verificación, durante una operación 78, el calculador 32 procede a una nueva tentativa de lectura del dato grabado en la memoria 38. La nueva tentativa de lectura consiste en leer el dato correspondiente a la dirección grabada en el registro de dirección de lectura. La operación 78 es por ejemplo idéntica a la operación 68.

A continuación, durante una operación 80, el calculador procede a una nueva comparación de los nuevos valores D_{1j} y D_{2j} leídos durante la operación 78.

Si esta vez, los valores D_{1j} y D_{2j} son idénticos, durante una operación 82, es detectado un error de lectura. En efecto, la diferencia entre los valores D_{1j} y D_{2j} leídos durante la operación 68 no proviene de una corrupción de los datos grabados en la memoria 38. De hecho, la detección de un error de lectura indica en el caso de un procesador de seguridad, con una probabilidad muy grande, que la primera tentativa de lectura durante la operación 68 ha fallado a causa de un ataque en lectura.

A partir de entonces, en respuesta, durante una operación 84, el calculador 32 dispara una contramedida que limita el desenredado de los contenidos multimedia con ayuda del procesador 16. Típicamente, para ello el descifrado de las palabras de control es inhibido.

Más precisamente, la contramedida puede ser una de las contramedidas siguientes:

- el borrado de los datos confidenciales contenidos en la memoria 38 tales como las claves criptográficas y los títulos de acceso,

- el disparo de la autodestrucción del procesador 16 de manera que le haga definitivamente inutilizable, y

- la detención del descifrado de las palabras de control temporalmente o definitivo.

Durante la operación 84, si el procesador 16 es aún utilizable a pesar de la puesta en práctica de una contramedida, el registro de dirección de lectura es borrado.

Si los valores D_{1j} y D_{2j} leídos durante la operación 78 son diferentes, entonces ello significa que los datos grabados están ciertamente corrompidos. No se trata por tanto de un error de lectura.

En este caso, durante una operación 86, el calculador 32 verificar con ayuda del código CD_{1j} si el valor D_{1j} es erróneo.

Si el valor D_{1j} no es erróneo, entonces se procede a una operación 88 de grabación del valor D_{1j} en nuevos bloques de memorias emparejados de las zonas 44 y 46. Por ejemplo, la operación 88 es realizada de manera similar a la fase 50 de escritura. A partir de entonces, es el valor grabado en estos nuevos bloques el que será utilizado durante la próxima lectura del mismo dato.

A continuación, durante una operación 90, las marcas MD_{1j} y MD_{2j} son actualizadas para indicar y memorizar que los bloques precedentes B_{1j} y B_{2j} son defectuosos. Aquí, el bloque B_{1j} es indicado como defectuoso mientras que el valor que estaba grabado en él era correcto. Ello permite conservar una gestión simple del emparejamiento de los bloques de memorias.

Durante la operación 90, el registro de dirección de lectura es igualmente borrado. A continuación, el valor D_{1j} es tratado por el programa que prosigue su ejecución por la operación 76.

Si durante la operación 86, el valor D_{1j} es erróneo, entonces se procede a una operación 92 de verificación con ayuda del código CD_{2j} sobre si el valor D_{2j} es erróneo o correcto.

Si el dato D_{2j} es correcto, entonces se procede a una operación 94 idéntica a la operación 88 salvo en que es el valor D_{2j} el que es utilizado en lugar del valor D_{1j}. La operación 94 se continua igualmente por la operación 90.

Si los valores D_{1j} y D_{2j} son erróneos, entonces, el calculador procede a una operación 98 durante la cual efectúa una primera tentativa de corrección de estos valores utilizando el código CC_0. esta operación 98 permite al calculador 32 corregir k bits erróneos repartidos en las zonas 44 y 46. Si hay menos de k bits erróneos, la corrección es entonces considerada como que ha tenido éxito. En este caso, se obtienen los valores D_{c1j} y D_{c2j} corregidos, respectivamente para los valores D_{1j} y D_{2j}.

En este caso, durante una operación 100, el calculador 32 comparados valores D_{c1j} y D_{c2j}.

Si los valores D_{c1j} y D_{c2j} son idénticos, durante una operación 102, el valor D_{c1j} es grabado en los nuevos bloques de memorias, respectivamente, de las zonas 44 y 46. Esta operación es por ejemplo idéntica a la operación 88 salvo en que es el valor D_{c1j} el que es utilizado el lugar del valor D_{1j}.

A continuación, se procede a una operación 104 durante la cual los bloques B_{1j} y B_{2j} son marcados como defectuosos. Esta operación 104 es por ejemplo idéntica a la operación 90. El procedimiento vuelve a continuación a la operación 76.

Si los valores D_{c1j} y D_{c2j} son diferentes o si la corrección de error con el código CC_0 no ha tenido éxito, el calculador 32 procede a una operación 108 durante la cual trata de corregir los datos de las zonas 44 con ayuda del código CC_1. si la corrección tiene éxito, el calculador 32 obtiene un valor D_{c1j} corregido del valor D_{1j}. Se procede entonces a una operación 110 idéntica a la operación 102. La operación 110 continúa por la operación 104.

Si la operación 108 es infructuosa y no permite corregir el valor D_{1j} entonces el calculador ejecuta una operación 112 durante la cual intenta corregir el valor D_{2j} con ayuda del código CC_2. Si esta operación 112 tiene éxito, el calculador 32 obtiene un valor corregido D_{c2j}. Se procede entonces a una operación 114 idéntica a la operación 94 salvo en que es el valor D_{c2j} el que es utilizado en lugar del valor D_{2j}. La operación 114 continúa por la operación 104.

Si no ha sido posible corregir ni el valor D_{1j} ni el valor D_{2j} entonces, durante una operación 116, el calculador 32 establece que el dato se ha perdido ya que éste es erróneo y no puede ser corregido. Durante esta operación 116, el calculador 32 marca los bloques de memorias B_{1j} y B_{2j} como defectuosos. Esta operación es realizada como se ha descrito con relación a la operación 90. A continuación, el programa es capaz de gestionar la ausencia del valor para este dato y en este caso la ejecución del programa prosigue. Si la ejecución del programa no puede proseguir sin el valor del dato, entonces la ejecución del programa es detenida y el procesador de seguridad es por ejemplo reinicializado.

La fig. 5 representa un procedimiento de protección del procesador 16 idéntico al procedimiento de la fig. 4 salvo que los códigos de corrección de error no son utilizados. Así, en este procedimiento, las operaciones 98 a 114 son omitidas. Además, en el caso entre los valores D_{1j} y D_{2j} son detectados como siendo los dos erróneos, se procede directamente a la operación 116.

Son posibles numerosos modos distintos de realización. Por ejemplo las operaciones de transformación del valor grabado pueden ser omitidas. En este caso, las operaciones 56 y 72 son omitidas.

La asignación de memoria puede ser una asignación de memoria lógica o física.

El algoritmo utilizado para detectar un error en el valor D_{1j} puede ser diferente del algoritmo utilizado para detectar un error en el valor D_{2j}. En este caso, los valores de los códigos CD_{1j} y CD_{2j} son diferentes.

En una variante, el código de detección de un error es también utilizado después de la corrección del valor grabado con ayuda del código de corrección de error. Ello permite verificar, si fuera necesario, que el valor corregido es correcto.

En otra variante, el código CC_0 es omitido, o a la inversa, los códigos CC_1 y CC_2 son omitidos o no es utilizado ningún código corrector de error.

En otra variante, un código corrector de error es únicamente utilizado para una sola de las zonas de memorias.

El código corrector de error no es necesariamente común para toda una zona de memoria. En una variante, el código corrector de error está construido para un grupo restringido de varios bloques de una zona de memoria. Un código corrector de error puede igualmente estar construido para cada bloque de memoria y únicamente para este bloque de memoria. En este caso, de preferencia, el código corrector de error reemplaza al código de detección de error. En efecto, casi todos los códigos de corrección de error permiten igualmente la detección de un error.

El código corrector de error puede igualmente ser común a los valores D_{1j} y D_{2j}.

El código corrector de error puede ser construido según otros algoritmos tales como el algoritmo de Hamming o un

turbo-código.

Cuando un bloque de memoria B_{1j} es detectado como defectuoso, no es necesario que el bloque de memoria B_{2j} que le está emparejado sea igualmente marcado de manera sistemática como siendo igualmente defectuoso. En una variante, el bloque B_{2j} es marcado como defectuoso únicamente si el código CD_{2j} asociado a este bloque confirma que el dato que contiene es igualmente erróneo. En caso contrario, una tabla asocia a la dirección de cada bloque B_{1j} la dirección del bloque emparejado B_{2j}. En este caso, esta tabla es modificada para asociar un nuevo bloque B'_{1j} utilizado para reemplazar el bloque precedente B_{1j} en la dirección del bloque B_{2j}.

La zona 42 de control puede ser grabada en los bloques de la zona 44. En estas condiciones, como todos los bloques de esta zona 44 está duplicada en la zona 46. Ellos permite por tanto proteger la zona de control contra la corrupción de datos o los errores de lectura de la misma manera que de no importa qué otro bloque de estas zonas 44 y 46.

La transformación del valor D_{ij} en un valor D'_{ij} puede ser omitida o únicamente empleada para una de las zonas 44 ó 46.

El orden de ciertas operaciones o etapas de los procedimientos descritos aquí puede ser modificado. Por ejemplo, el cálculo del código de detección de error es realizado después de la transformación del valor D_{ij} en un valor transformado D'_{ij}. En este caso, durante la lectura, la verificación de que el dato leído es correcto o erróneo es realizada a partir del valor D'_{ij} y no del valor D_{ij}.

En una variante, antes de actualizar la marca MD_{ij} de un bloque defectuoso, el calculador verifica que el bloque B_{ij} es realmente defectuoso. Por ejemplo realiza las operaciones siguientes:

a) Escritura de un valor D_{ij} en el bloque B_{ij} en cuestión, luego

b) Lectura del valor D_{ij} grabado en este bloque B_{ij},

c) Comparación de los valores escrito y leído, y luego

d) Si estos valores son iguales, las operaciones a) a c) son reiteradas al menos N veces. Si no, la marca MD_{ij} es actualizada para indicar que el bloque B_{ij} es defectuoso.

Típicamente, el número N es superior a dos y, de preferencia superior a diez.

Si las operaciones precedentes a) a d) son a menudo puestas en práctica para un mismo bloque B_{ij} pero que la verificación precedente conduce cada vez a dejar del bloque B_{ij} como siendo utilizable, un valor particular puede ser afectado a la marca MD_{ij} indicando que este bloque B_{ij} es poco seguro. En estas condiciones, en tanto que sea posible, el bloque B_{ij} no es elegido para grabar en él nuevos datos. Por el contrario, si no existen otros bloques de memorias más seguros disponibles, este bloque B_{ij} será entonces utilizado para grabar un dato.

Cada una de las memorias descritas aquí puede ser realizada en forma de un solo componente electrónico o de una asociación de varios componentes electrónicos conectados independientemente unos de los otros al calculador 32. Por ejemplo, las zonas 44 y 46 pueden corresponder a dos memorias físicamente distintas unidas cada una por su propio bus de lectura al calculador 32.

Más de dos zonas de memorias redundantes pueden ser empleadas. En este caso, el valor del dato copiado en esta memoria es copiado en cada una de estas zonas de memorias. Los procedimientos descritos anteriormente pueden ser adaptados fácilmente al caso de W zonas de memorias donde W es un número entero estrictamente superior a dos.

Lo que se ha descrito precedentemente puede aplicarse igualmente a una memoria no volátil.

REIVINDICACIONES

1. Procedimiento de detección de un error de lectura de un dato, incluyendo este procedimiento:

a) la grabación (50) de una primera copia del dato en una primera zona de una memoria electrónica y la grabación de una segunda copia del dato en una segunda zona de una memoria electrónica,

en respuesta a una solicitud de lectura del dato;

b) la lectura (68) de los valores de la primera y segunda copias del dato respectivamente en la primera y segunda zonas,

c) la comparación (74) de los valores leídos de la primera y segunda copias del dato,

d) si los valores leídos de la primera y segunda copias son idénticos, entonces no es detectado ningún error de lectura de este dato,

caracterizado por que:

e) si los valores leídos de la primera y segunda copias son diferentes, entonces las operaciones b) y c) precedentes son reiteradas (78, 80), y luego

f) si los valores leídos durante la operación e) son idénticos, entonces un error de lectura de este dato es detectado (82) y, si no, ningún error de lectura de este dato es detectado.

2. Procedimiento según la reivindicación 1 en el que el procedimiento incluye:

- la asociación (54) al valor de la primera copia del dato de un código de detección de error que añade redundancia al dato y que permite detectar un error en el valor de este dato,

- si durante la operación f) los valores leídos de la primera y segunda copias son diferentes, entonces el procedimiento incluye la verificación (86) de si el valor leído de la primera copia es erróneo o al contrario correcto con ayuda del código de detección de error asociado a este valor, y

- si el valor leído de la primera copia es correcto, la grabación (88) del valor de la primera copia en un bloque de memoria de la segunda zona de memoria diferente del bloque de memoria donde se encontraba la segunda copia precedente de este dato de manera que forme una nueva segunda copia del dato, y luego

- la utilización de la nueva segunda copia en lugar y en vez de la segundo copia precedente del dato durante toda nueva iteración de las operaciones b) y c).

3. Procedimiento según una cualquiera de las reivindicaciones precedentes, en el que el procedimiento incluye:

- la asociación (54) a la primera y segunda copias del dato de un código de corrección de error que añade suficiente redundancia a los valores de la primera y segunda copias para permitir corregir uno o varios bits erróneos en el valor de cada una de estas copias del dato,

- si durante la operación f) los valores leídos de la primera y segunda copias son diferentes, entonces el procedimiento incluye:

  - la corrección (98) con ayuda del código de corrección de error de los valores de la primera y segunda copias del dato para obtener valores corregidos para estas primera y segunda copias, y luego
  - la comparación (100) de los valores corregidos de la primera y segunda copias, y
  - si los valores corregidos son idénticos, la grabación (102) del valor corregido de la primera copia del dato en nuevos bloques de memorias, respectivamente, de la primera y la segunda zonas de memorias diferentes de los bloques de memorias donde se encontraban las precedentes primera y segunda copias de este dato de manera que formen nuevas primera y segunda copias del dato, y luego
  - la utilización de las nuevas primera y segunda copias en lugar y en vez de las precedentes primera y segunda copias durante toda nueva iteración de las operaciones b) y c).

4. Procedimiento según la reivindicación 2 ó 3, en el que para cada bloque de memoria en que se encontraba una copia del dato precedente, el procedimiento incluye:

- la memorización (90, 104, 116) de este bloque de memoria como defectuoso, y

- durante la asignación de un nuevo bloque de memoria en una de las zonas de memoria para grabar en ella una información, la elección (52) de este bloque de memoria únicamente entre los bloques de memorias de esta zona no

memorizados como defectuosos.

5. Procedimiento según una cualquiera de las reivindicaciones precedentes, en el que:

- durante la grabación de la primera copia del dato, el valor del dato es transformado en primer lugar (56) con ayuda de una función de enmascaramiento que puede invertirse para obtener un valor transformado diferente del valor no transformado y diferente del valor grabado de la segunda copia, y luego es este valor transformado el que es grabado en la primera zona, y

- durante la lectura de la primera copia, la transformación inversa es aplicada (72) al valor transformado grabado para obtener el valor leído de la primera copia.

6. Procedimiento de protección de un procesador de seguridad contra un ataque en lectura, ejecutando este procesador de seguridad (76) una rutina de cifrado o de descifrado de una información con la ayuda de un dato confidencial grabado en una primera y segunda zonas de memoria de este procesador de seguridad, caracterizado por que:

- el procesador de seguridad detecta (82) un error de lectura del dato confidencial poniendo en práctica un procedimiento conforme a una cualquiera de las reivindicaciones precedentes,

- si no se ha detectado ningún error de lectura del dato confidencial, el procesador de seguridad procede al cifrado o al descifrado (76) de la información con ayuda del dato confidencial leído, si no

- en respuesta a la detección de un error de lectura, el procesador de seguridad dispara (86) automáticamente una contramedida que limita el cifrado o descifrado de la información con ayuda del dato confidencial leído, con relación al caso en que no se ha detectado ningún error de lectura de este dato confidencial.

7. Soporte (38) de grabación de informaciones, caracterizado por que incluye instrucciones para la puesta en práctica de un procedimiento conforme a una cualquiera de las reivindicaciones precedentes, cuando estas instrucciones son ejecutadas por un calculador electrónico.

8. Procesador de seguridad que incluye:

- una primera (44) y una segunda (46) zonas de una memoria electrónica,

- un calculador electrónico (32) programado para:

a) grabar una primera copia del dato en la primera zona y grabar una segunda copia del dato en la segunda zona,

en respuesta a una solicitud de lectura del dato:

b) leer los valores de la primera y segunda copias del dato, respectivamente, en la primera y segunda zonas,

c) comparar los valores leídos de la primera y segunda copias del dato, y

d) si los valores leídos de la primera y segunda copias son idénticos, entonces no detectar ningún error de lectura de este dato,

caracterizado por que el calculador electrónico está igualmente programado para:

e) si los valores leídos de la primera y segunda copias son diferentes, reiterar las operaciones b) y c) precedentes, y luego

f) si los valores leídos durante la operación e) son idénticos, entonces detectar un error de lectura de este dato y, si no, no indicar ningún error de lectura.

9. Procesador según la reivindicación 8, en el que el procesador de seguridad es el procesador de seguridad de una tarjeta con chip (30).

10. Procesador según una cualquiera de las reivindicaciones 8 a 9, en el que la memoria electrónica (38) llamada primera memoria, es una memoria no volátil y el procesador de seguridad incluye una segunda memoria (36) volátil en la que son sistemáticamente copiados los datos requeridos en la primera memoria (38) a fin de ser tratados por el calculador electrónico (32).

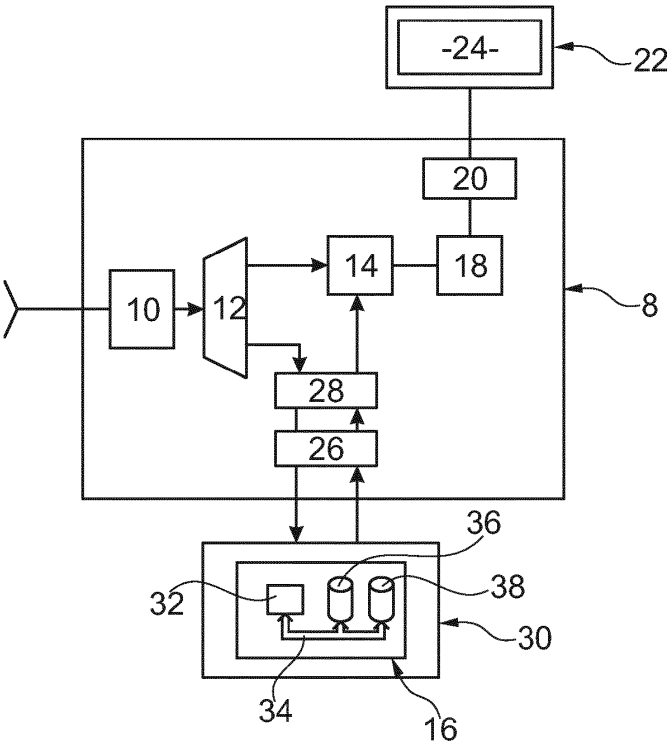

Fig. 1

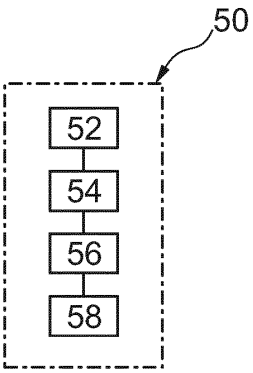

Fig. 3

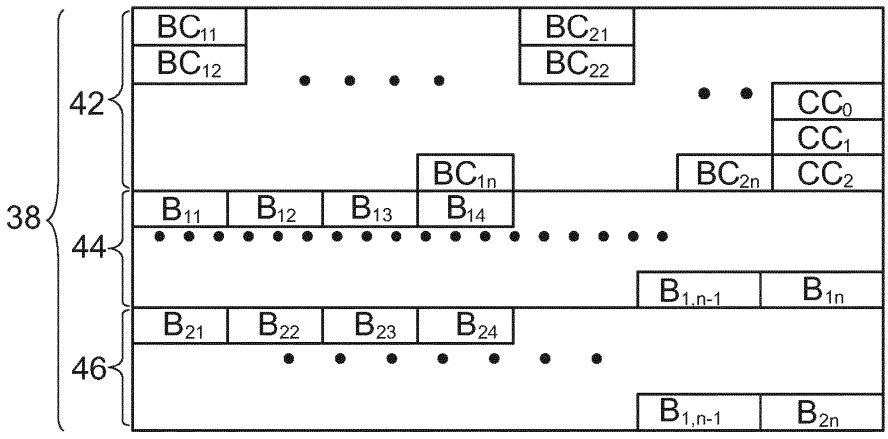

Fig. 2

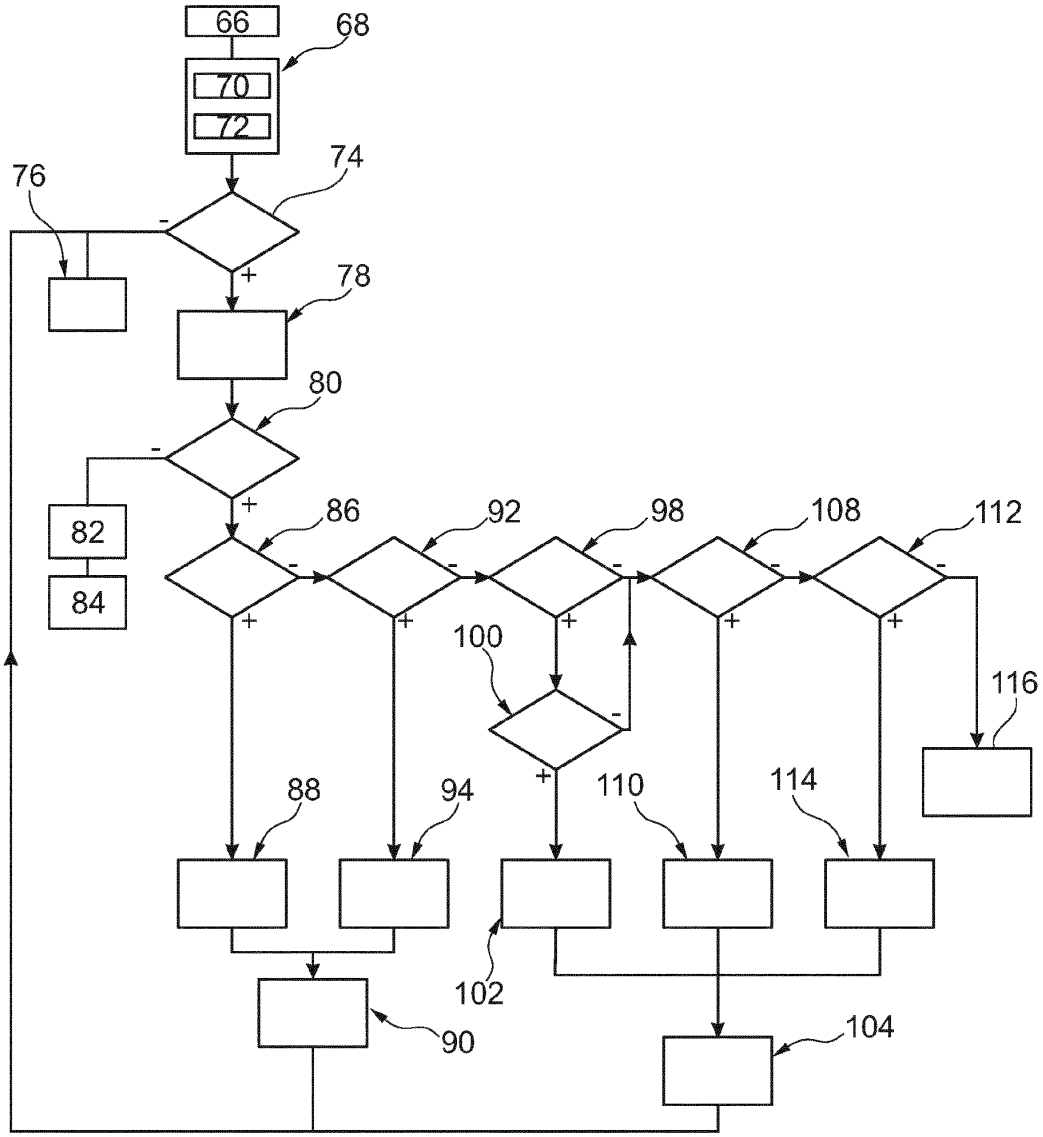
66
68
70
72
74
76
78
80
82
84
86
92
98
108
112
100
116
88
94
110
114
102
90
104

Fig. 4

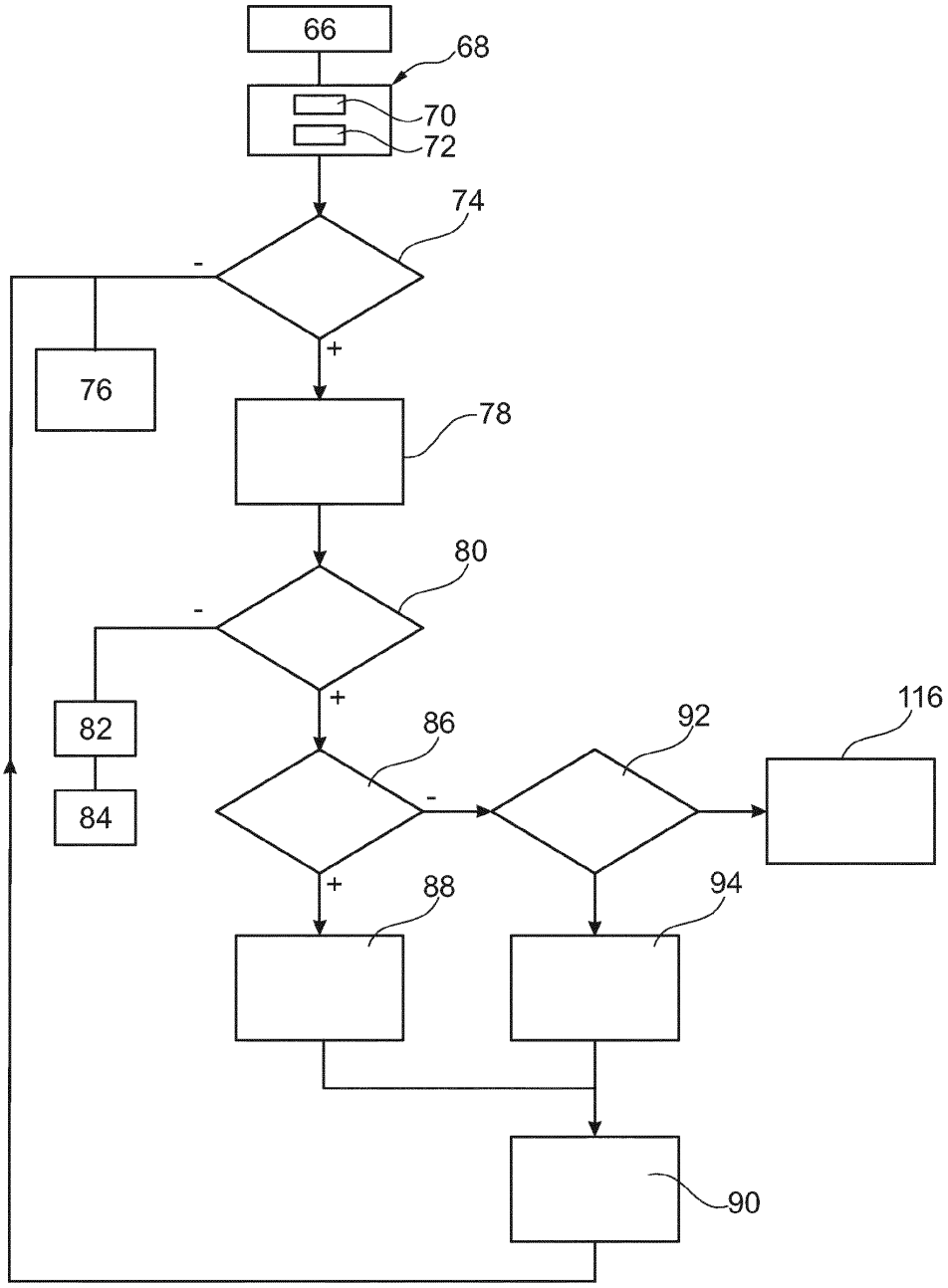

Fig. 5