(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 824 830**

(51) Int. CI.:

G06F 8/654 (2008.01)
G06F 11/14 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **02.11.2011 PCT/EP2011/069233**

(87) Fecha y número de publicación internacional: **18.05.2012 WO12062632**

(96) Fecha de presentación y número de la solicitud europea: **02.11.2011 E 11776462 (1)**

(97) Fecha y número de publicación de la concesión europea: **05.08.2020 EP 2638466**

(54) Título: **Procedimiento de actualización de software para un dispositivo integrado**

(30) Prioridad:

08.11.2010 EP 10306218

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
13.05.2021

(73) Titular/es:

THALES DIS FRANCE SA (100.0%)
6, rue de la Verrerie
92190 Meudon, FR

(72) Inventor/es:

DURAND, STÉPHANE

(74) Agente/Representante:

ELZABURU, S.L.P

DESCRIPCIÓN

Procedimiento de actualización de software para un dispositivo integrado

La presente invención se refiere generalmente a una actualización de software en un dispositivo integrado.

Un sistema integrado es un sistema informático especializado, que incluye tanto hardware como software, que forma parte de un sistema o máquina más grande. Además, el sistema o máquina más grande puede tener una pluralidad de sistemas integrados. Por lo común, cada uno de los sistemas integrados se aloja en una sola placa de microprocesador con firmware almacenado como código objeto dentro de un dispositivo de memoria no volátil.

El sistema o máquina más grande que generalmente utiliza el sistema integrado puede incluir una amplia variedad de sistemas, que van desde teléfonos hasta dispositivos de almacenamiento masivo, receptores de satélite digitales y sistemas similares. Los sistemas integrados comúnmente utilizan software, entre el que se encuentra el firmware que proporciona un sistema operativo que administra las funciones de bajo nivel del sistema integrado.

Por lo común, un sistema integrado proporciona un procedimiento de actualización de código para admitir nuevas funciones o solucionar problemas en el firmware (es decir, actualización del firmware). En muchos casos, la actualización del firmware dejará fuera de servicio el sistema o la máquina más grande durante un período de tiempo durante el cual se realiza la actualización del firmware.

Para proporcionar una característica de actualización de firmware, el sistema integrado generalmente incluye una memoria de tipo Flash para almacenar el firmware. El firmware habitualmente comprende tanto un cargador de arranque como un firmware principal almacenado en un sector de arranque de la memoria de tipo flash. El cargador de arranque se coloca en la primera dirección de memoria de la memoria de tipo Flash y se ejecuta en el momento del arranque. El cargador de arranque se utiliza como firmware a prueba de fallos en caso de que el firmware principal esté dañado. El cargador de arranque se ejecuta en el momento del arranque, de manera que comprueba, a su vez, la integridad del firmware principal (preferiblemente calculando una suma de comprobación y comparándola con un resultado almacenado o un valor predeterminado) antes de que se ejecute realmente el firmware principal. El cargador de arranque puede diagnosticar, además, si el firmware principal está dañado y habilitar una actualización de un firmware principal dañado. Si el cargador de arranque no detecta un error de integridad, bien bifurca el microprocesador en la primera dirección del firmware principal o bien realiza su carga. Si el cargador de arranque detecta un error de integridad, puede usar una interfaz de E/S para recibir un código de reemplazo de firmware principal. Alternativamente, se puede proporcionar un firmware principal de refuerzo para que una prueba de integridad fallida dé como resultado la ejecución del firmware principal de refuerzo.

Por ejemplo, el documento US 2005/0060528 proporciona un ejemplo de un método para actualizar un cargador de arranque de un dispositivo, que incluye una verificación de la completitud de la actualización antes de reiniciar el dispositivo.

En la práctica, algunos sistemas que ya están en uso pueden requerir un nivel de seguridad mejorado, como tarjetas de acceso o teléfonos inteligentes. De hecho, un ataque conocido para eludir las características de seguridad del firmware principal consiste en cambiar la dirección del firmware principal mencionado en el programa del cargador de arranque. De este modo, el sistema puede ejecutar un firmware principal falso. El cargador de arranque también puede ser atacado para descargar una actualización de firmware principal falsa y activar su ejecución.

El documento US 2010/0131694 proporciona un método para proteger dicho cargador de arranque contra los intentos de un atacante de reemplazarlo por un cargador de arranque dañado.

Sin embargo, tales métodos podrían dar como resultado que el dispositivo actualizado no se pueda utilizar en caso de fallo de energía en algunas etapas cruciales de los métodos.

El documento US 5701492 divulga un método de actualización de cargador de arranque protegido contra fallos de energía, pero requiere un área del cargador de arranque en la memoria al menos dos veces más grande que el tamaño del cargador de arranque.

En consecuencia, existe la necesidad de un método mejorado de actualización de cargador de arranque protegido frente a fallos de energía, con menores requisitos de tamaño de memoria.

De esta forma, la invención se refiere a un método de acuerdo con la reivindicación 1.

De acuerdo con otra realización, el método comprende una etapa b') entre las etapas a) y c), de tal manera que la etapa b') comprende un reinicio del dispositivo integrado.

De acuerdo con una realización adicional, la etapa d) comprende modificar la instrucción de salto para bifurcar la ejecución al software actualizado.

De acuerdo con otra realización, la modificación de la instrucción de salto se realiza mediante una operación de inscripción atómica.

De acuerdo con una realización adicional, el método comprende, además, una etapa e) en la que la parte del cargador de arranque actualizada borra el código del programa de gestión de actualización y escribe una parte de firmware principal actualizada en esta ubicación borrada.

De acuerdo con una realización adicional, la memoria EEPROM comprende modos de borrado y escritura tales, que la unidad de datos de memoria más pequeña borrada en el modo de borrado es de mayor tamaño que la unidad de datos de memoria más pequeña escrita en el modo de escritura.

De acuerdo con otra realización, la modificación de esta primera parte del cargador de arranque comprende establecer todas las palabras de datos almacenadas en el área del cargador de arranque en un mismo valor, comprendiendo el dispositivo integrado un microprocesador diseñado para manejar estas palabras de datos que tienen este mismo valor que instrucciones neutras.

La ventaja de la presente invención resultará evidente a partir de la siguiente descripción de varias realizaciones con referencia a los dibujos adjuntos, en los que:

- La Figura 1 es una vista esquemática de un dispositivo integrado para el que se puede llevar a cabo la invención;

- La Figura 2 es una vista esquemática de un área de memoria no volátil utilizada para almacenar un cargador de arranque y un firmware principal;

- La Figura 3 ilustra las etapas llevadas a cabo de acuerdo con una realización de la invención.

La Figura 1 es una vista esquemática de un ejemplo de dispositivo integrado 1 para el que se puede llevar a cabo la invención. En este ejemplo, el dispositivo integrado 1 es una tarjeta inteligente de acceso utilizada para autentificar a un usuario de forma distante. El dispositivo integrado 1 incluye un microcontrolador 11. Este microcontrolador 11 incluye una memoria RAM 14, una memoria EEPROM no volátil 12 y un microprocesador 15. El microcontrolador 11 también incluye una primera interfaz de entrada / salida 17 y una segunda interfaz de entrada / salida 16. La interfaz de E/S 17 está conectada a una interfaz de contacto 3 de la tarjeta inteligente. La interfaz de contacto 3 está configurada para un lector de tarjetas inteligentes, por ejemplo, un lector de tarjetas inteligentes con comunicación por USB. Por tanto, el microcontrolador 11 es capaz de comunicarse con otros dispositivos, por ejemplo, para llevar a cabo una transacción que incluye una autentificación. La interfaz de E/S 16 está conectada a una antena 4. La antena 4 está configurada para proporcionar una comunicación sin contacto con otros dispositivos, por ejemplo, para llevar a cabo una transacción que incluye una autentificación. La antena 4 puede adaptarse, en particular, para comunicarse usando un protocolo IEEE 802.11.

La Figura 2 es una vista esquemática de la memoria EEPROM 12, por ejemplo, una memoria de tipo Flash. Las memorias de tipo flash se encuentran comúnmente en productos electrónicos de consumo. La memoria de tipo flash se valora en muchas aplicaciones como medio de almacenamiento debido a sus rápidas velocidades de acceso, bajo consumo de energía y funcionamiento no volátil. La memoria EEPROM 12 comprende un área de almacenamiento 121 de cargador de arranque y un área de almacenamiento 122 de firmware principal. La memoria EEPROM 12 almacena el firmware a cargo de administrar las características de bajo nivel del microcontrolador 11. El firmware comprende una parte 123 de cargador de arranque, almacenada en el área 121, y una parte 124 de firmware principal, almacenada en el área 122. El tamaño de la memoria EEPROM 12 puede ser, por lo común, de aproximadamente 400 kB. Obviamente, también se pueden usar diferentes tamaños de memoria sin apartarse de las enseñanzas de la invención.

La parte 123 de cargador de arranque está destinada a ser el primer software ejecutado por el microprocesador 15 en el arranque. La parte 123 de cargador de arranque define en qué dirección se almacena el firmware principal en la EEPROM 12. La parte 123 de cargador de arranque también comprende características para verificar la integridad de la parte 124 de firmware principal. La parte 123 de cargador de arranque se encarga, en particular, de actualizar la parte 124 de firmware principal en caso de que se detecte un fallo de integridad. El área de almacenamiento 121 de cargador de arranque puede tener una capacidad de almacenamiento mayor que el tamaño de la parte 123 de cargador de arranque almacenada.

De acuerdo con la invención, el área de almacenamiento 121 de cargador de arranque puede almacenar una parte del cargador de arranque cargada durante el proceso de fabricación de la tarjeta inteligente 1, o una parte del cargador de arranque cargada en la tarjeta inteligente 1 durante su ciclo de vida.

La memoria EEPROM 12 incluye un conjunto ordenado de bloques de datos. Un bloque de datos se define como el área de almacenamiento más pequeña que se puede borrar de forma independiente. Una página define la unidad más pequeña que se puede escribir en la memoria de tipo Flash en una sola operación. El tamaño de la página define, por tanto, la granularidad de escritura de la memoria de tipo Flash. Estas características de acceso de la memoria de tipo Flash provocan dificultades de gestión. El área de almacenamiento 121 de cargador de arranque comprende uno o más bloques. El área de almacenamiento 122 de firmware principal comprende uno o más bloques.

Si la memoria EEPROM 12 es del tipo Flash, solo permite dos estados de almacenamiento: borrado y no borrado. En el estado borrado, un byte puede ser todos unos (OxFF) o todos ceros (OxOO), dependiendo del dispositivo de tipo flash. En el ejemplo, un byte borrado contiene OxFF. Durante una operación de escritura, uno o más bits se establecen en 0. Un bit de datos solo se puede escribir cuando está inicialmente en un estado de borrado. Una vez escrito, no se puede escribir en el bit antes de borrarlo. Para devolver el bit a su estado de borrado, se debe borrar un bloque de datos completo. La tecnología Flash no permite la alternancia de bits o bytes individuales de un estado no borrado a un estado borrado.

Entre los diversos tipos de memorias de tipo flash, las memorias Flash se encuentran comúnmente en sistemas integrados. Una memoria Flash se organiza en páginas de tamaño fijo (por ejemplo, 512 bytes por página), y un cierto número de páginas constituye un bloque (por ejemplo, 32 páginas por bloque). Con algunos tipos de memorias Flash como la memoria NOR-Flash, el tamaño de la página puede ser tan pequeño como una palabra.

El objetivo de la invención es reemplazar de forma segura el software previo por software actualizado en un dispositivo integrado. Un usuario puede realizar convenientemente dicho reemplazo de software para corregir errores del software anterior o para mejorar sus características de seguridad, a todo lo largo del ciclo de vida del dispositivo integrado.

En el caso específico de un procedimiento de actualización de un cargador de arranque, el método de actualización de acuerdo con la invención garantiza que una interrupción repentina de la energía no inutilizará un dispositivo integrado, incluso si la interrupción ocurre después de que se haya borrado el último cargador de arranque y antes de que el cargador de arranque actualizado se active con éxito. Un dispositivo integrado podría volverse inutilizable, en particular, si ocurriera una interrupción durante la escritura del cargador de arranque en la EEPROM y si este cargador de arranque incompleto no se pudiera ejecutar en el siguiente inicio.

La Figura 3 es un diagrama de bloques que enumera las etapas realizadas por un método de acuerdo con una realización de la invención.

En la etapa 201, el dispositivo integrado 1 se sitúa en un entorno en el que puede comunicarse con dispositivos externos. El dispositivo integrado 1 almacena un cargador de arranque previo en su memoria EEPROM 12. El usuario final solicita una actualización del cargador de arranque a través de un dispositivo externo, o bien la activa el propio dispositivo integrado 1. El dispositivo integrado 1 se reinicia y el microprocesador 15 ejecuta el cargador de arranque previo.

En la etapa 202, el cargador de arranque previo borra los bloques de datos que contienen la parte de firmware principal previa del dispositivo 1 integrado, esto es, los bloques de datos del área de almacenamiento 122 de firmware principal.

En la etapa 203, el cargador de arranque previo carga un código de programa de gestión de actualización en la memoria EEPROM 12. El código del programa de gestión de actualización se almacena, por ejemplo, en el área de almacenamiento 122 de firmware principal, en los bloques de datos borrados en la etapa 202. Ventajosamente, la dirección de ejecución del código de programa de gestión de actualización es idéntica a la dirección de ejecución de la parte de firmware principal previo. El comienzo del área de almacenamiento 122 de firmware principal es, ventajosamente, el bloque de datos próximo al área de almacenamiento 121 de cargador de arranque.

En la etapa 204, un cargador de arranque actualizado es cargado en la memoria EEPROM 12, por ejemplo, en el área de almacenamiento 122 de firmware principal, próxima al código del programa de gestión de actualización. El cargador de arranque actualizado se puede almacenar en los bloques de datos borrados, por ejemplo, contiguo al código de programa de gestión de actualización. Este cargador de arranque actualizado está en una forma no arrancable. La dirección de este cargador de arranque actualizado está escrita en el código de programa de gestión de actualización, a fin de que este código de programa de gestión de actualización pueda acceder a ella en un estadio posterior.

En la etapa 205, el dispositivo integrado 1 es, por ejemplo, reiniciado. El cargador de arranque previo todavía se ejecuta en un primer momento. Una vez que ha realizado una verificación de integridad satisfactoria del código de programa de gestión de actualización, activa la ejecución de este código de programa de gestión de actualización. La gestión del dispositivo integrado 1 se transfiere así al código del programa de gestión de actualización.

En la etapa 206, el código de programa de gestión de actualización borra todos los bloques de datos que lo preceden, es decir, todos los bloques de datos del área de almacenamiento 121 de cargador de arranque. El cargador de arranque previo se borra por tanto y, en consecuencia, no se puede utilizar.

En la etapa 207, el código de programa de gestión de actualización almacena una instrucción de actualización que se ejecutará al inicio en el área de cargador de arranque. Por ejemplo, el código de programa de gestión de actualización escribe una instrucción de salto en su propia dirección, en la primera posición de memoria del área de almacenamiento 121 de cargador de arranque. Cuando se lee, la instrucción de salto bifurca la ejecución al código de programa de gestión de actualización.

En la etapa 208, el código de programa de gestión de actualización copia el cargador de arranque actualizado en el área de almacenamiento 121 de cargador de arranque. El salto al código de programa de gestión de actualización permanece almacenado hasta el final de la copia del cargador de arranque actualizado.

En la etapa 209, el código de programa de gestión de actualización finaliza la copia del cargador de arranque actualizado en el área de almacenamiento 121 de cargador de arranque. Realiza una verificación de integridad en el cargador de arranque actualizado copiado para confirmar que este cargador de arranque actualizado se puede ejecutar de forma segura. El código de programa de gestión de actualización modifica entonces la instrucción de salto que incluye su propia dirección. Esta instrucción de salto se modifica, preferiblemente mediante una operación de inscripción atómica, convirtiéndose en una instrucción que bifurca la ejecución al cargador de arranque actualizado al principio del siguiente inicio. La instrucción de salto puede modificarse de manera que refiera la dirección del principio del cargador de arranque actualizado, o bien modificarse como una instrucción neutra, seguida por el propio cargador de arranque actualizado.

En la etapa 210, se reinicia el dispositivo integrado 1 y se ejecuta el cargador de arranque actualizado. El cargador de arranque actualizado instala, ventajosamente, una parte de firmware principal actualizada en el área de almacenamiento 122 de firmware principal. El cargador de arranque actualizado borra, en primer lugar, el (los) bloque(s) de datos que almacena(n) el código de programa de gestión de actualización. El cargador de arranque actualizado escribe entonces la parte de firmware principal actualizada en los bloques de datos borrados, y almacena la dirección de memoria de la parte de firmware principal actualizada.

Se mostrará, a continuación, la seguridad del método analizando la incidencia de una interrupción en cada etapa.

Si tiene lugar una interrupción en la etapa 202, el cargador de arranque previo permanece operativo. Al inicio, se ejecuta el cargador de arranque previo y este puede realizar de manera satisfactoria el borrado de la parte de firmware principal previa.

Si tiene lugar una interrupción en la etapa 203, el cargador de arranque previo detecta un error de integridad en el código de programa de gestión de actualización cargado. Al inicio, se ejecuta el cargador de arranque previo y este puede realizar de manera satisfactoria la carga del código de programa de gestión de actualización.

Si se produce una interrupción en la etapa 204, el cargador de arranque previo detecta un error de integridad en el cargador de arranque actualizado cargado. Al inicio, se ejecuta el cargador de arranque previo y este puede realizar de manera satisfactoria la carga del cargador de arranque actualizado en la memoria EEPROM 12. La operación en la que la dirección del cargador de arranque actualizado se escribe en el código de programa de gestión de actualización puede ser atómica, de modo que no hay riesgo de almacenar una dirección inconsistente después de una interrupción.

Si tiene lugar una interrupción en la etapa 205, se ejecuta primeramente el cargador de arranque previo en el siguiente inicio y luego este entrega la gestión del dispositivo 1 integrado al código de programa de gestión de actualización. Por tanto, se reanuda la actualización del cargador de arranque.

Si se produce una interrupción en la etapa 206, los bloques de datos del cargador de arranque previo se borran o permanecen sin cambios, debido a la naturaleza atómica de una etapa de borrado de bloques de datos. Si el borrado no se realizó correctamente, el cargador de arranque previo se ejecutará nuevamente en el próximo inicio y el código de programa de gestión de actualización realizará la etapa de borrado una vez más.

En caso de que los bloques de datos se borren con éxito, todas las palabras de datos almacenadas en ellos tienen el mismo valor. El microprocesador 15 maneja, ventajosamente, tales palabras de datos como instrucciones neutras: una vez ejecutada, una instrucción neutra no interrumpe la ejecución del microprocesador 15 o no modifica su flujo de ejecución. El microprocesador pasa por encima de esa instrucción y ejecuta la siguiente instrucción. Esta puede ser una instrucción indefinida que no desencadena una excepción del microprocesador, o una instrucción básica (por ejemplo, el copiado de un valor de registro en sí mismo) que lleva al microprocesador 15 a ejecutar automáticamente la siguiente instrucción almacenada. Si se produce una interrupción una vez que se han borrado los bloques de datos del cargador de arranque previo, en el siguiente inicio, se lee una instrucción neutra en la primera dirección de la parte de almacenamiento de cargador de arranque. A continuación, el microprocesador 15 ejecuta las siguientes instrucciones neutras de la parte de almacenamiento de cargador de arranque, hasta que llega a la instrucción de inicio del código de programa de gestión de actualización. Seguidamente, se ejecuta el código de programa de gestión de actualización, lo que lleva a la actualización de la parte de cargador de arranque. De esta forma, incluso si la dirección del código de programa de gestión de actualización no pudiera almacenarse en la parte de almacenamiento de cargador de arranque después de una interrupción, el microprocesador 15 aún puede acceder al código de programa de gestión de actualización y ejecutarlo.

En la etapa 207, la escritura de la instrucción de salto en la dirección del código de programa de gestión de actualización, en el área de cargador de arranque, es, ventajosamente, atómica, con el fin de proporcionar una garantía de seguridad. De esta forma, esta operación de escritura no se ve afectada por una interrupción. La instrucción de salto constituye una instrucción de actualización, que proporciona acceso al código de programa de gestión de actualización sin tener que solicitar la ejecución de todas las instrucciones neutras almacenadas en el

área de almacenamiento de cargador de arranque borrada. En lugar de una instrucción de salto, se puede utilizar otro tipo de instrucción de bifurcación como instrucción de actualización.

Si se produce una interrupción en el paso 208 hasta la modificación de la instrucción de salto, la instrucción de salto se lee primeramente y el código de programa de gestión de actualización se ejecuta en el siguiente inicio. A continuación, el código de programa de gestión de actualización reanuda la copia del cargador de arranque actualizado en el área de almacenamiento 121 de cargador de arranque.

La instrucción de actualización puede estar formada por un código de salto que comprende varias instrucciones y termina con la instrucción de salto. En tal caso, las primeras instrucciones del código de salto se escriben inicialmente en el área de almacenamiento 121 de cargador de arranque, y la instrucción de salto se escribe en una última operación de escritura atómica.

A diferencia de lo que podría haber concebido un experto en la técnica, las operaciones de borrado y escritura del área de almacenamiento 121 de cargador de arranque están desfasadas en el tiempo, lo que, de hecho, proporciona una seguridad de actualización del cargador de arranque.

El microcontrolador 11 puede incluir, en particular, un microprocesador 15 distribuido comercialmente bajo la referencia ARM7.

Las siguientes instrucciones pueden almacenarse en el área de almacenamiento 121 de cargador de arranque, en la etapa 207:

@OxOO: PC LDR, OxOOOAAAOOBB

(Esta instrucción almacena la dirección en el registro en curso en ese momento)

@ xO1: SALTO OxOOOAAAOOBB

(Esta instrucción es una instrucción de salto que bifurca la ejecución a la dirección OxOOOAAAOOBB)

Ventajosamente, la parte AAA de la dirección constituye un desplazamiento de direcciones entre el cargador de arranque actualizado y el código de programa de gestión de actualización. Esto significa que la dirección del cargador de arranque actualizado se puede establecer como OxOOOOOOOOBB. Usando tal desplazamiento, la dirección almacenada en OxYY se puede modificar fácilmente en la etapa 209 de manera que apunte al cargador de arranque actualizado en lugar de al código de programa de gestión de actualización. Los valores 1 incluidos en el desplazamiento de direcciones simplemente se establecen en valores 0 mediante una operación de escritura.

Aunque la invención se ha descrito en su aplicación a un procedimiento de actualización de cargador de arranque, la invención también se aplica a un procedimiento de actualización de cualquier otro tipo de software almacenado en una memoria EEPROM.

REIVINDICACIONES

1. Un método para actualizar una parte (123) de cargador de arranque previo de un firmware de un dispositivo integrado (1), de tal manera que dicho firmware comprende un conjunto ordenado de bloques de datos y dicho firmware comprende dicha parte (123) de cargador de arranque previo almacenada en un área de almacenamiento (121) de cargador de arranque de la memoria EEPROM del dispositivo integrado, para que se ejecute al inicio del dispositivo integrado, y una parte (124) de firmware principal previo, almacenada en un área de almacenamiento (122) de firmware principal de la memoria EEPROM, de tal modo que el método comprende las etapas, ejecutadas por el dispositivo integrado, de:

a) borrar (202) bloques de datos del área de almacenamiento de firmware principal,

b) cargar en el área de almacenamiento (122) de firmware principal borrada un código de programa de gestión de actualización (203), en una ubicación subsiguiente al área de almacenamiento (121) de cargador de arranque, y una parte (204) de cargador de arranque actualizado, en una ubicación a la que se accede durante la ejecución del código del programa de gestión de actualización,

c) ejecutar la parte (205) de cargador de arranque previo desencadenando una ejecución de dicho código de programa de gestión de actualización, comprendiendo esta etapa de ejecución del código de programa de gestión de actualización:

- borrar (206) bloques de datos del área de almacenamiento de cargador de arranque, de manera que se borre toda el área de memoria que precede al código de programa de gestión de actualización, y almacenar (207) al menos una instrucción de salto al comienzo de dicha área de almacenamiento de cargador de arranque borrada, de tal modo que la ejecución de esta instrucción de salto al inicio bifurca la ejecución del código de programa de gestión de actualización;

- escribir (208) de la parte de cargador de arranque actualizado en el área de almacenamiento de cargador de arranque borrada de la memoria EEPROM;

d) modificar (209) la instrucción de salto de tal modo que la parte de gestor de arranque actualizado del firmware se ejecute al inicio.

2. Un método de acuerdo con la reivindicación 1, que comprende, además, una etapa b') entre las etapas a) y c), de tal manera que la etapa b') comprende un reinicio del dispositivo integrado.

3. Un método de acuerdo con la reivindicación 1, en el que la etapa d) comprende modificar (209) la instrucción de salto para bifurcar la ejecución al software de cargador de arranque actualizado.

4. Un método de acuerdo con la reivindicación 3, en el que la modificación de la instrucción de salto se realiza mediante una operación de inscripción atómica.

5. Un método de acuerdo con la reivindicación 1, que comprende, además, una etapa e) en la que la parte de cargador de arranque actualizado borra el código de programa de gestión de actualización y escribe una parte de firmware principal actualizada en esta posición borrada.

6. Un método de acuerdo con la reivindicación 1, en el que la memoria EEPROM (12) comprende modos de borrado y escritura, de tal manera que la unidad de datos de memoria más pequeña borrada en modo de borrado es de mayor tamaño que la unidad de datos de memoria más pequeña escrita en modo de escritura.

7. Un método de acuerdo con las reivindicaciones 1 y 6, en el que la modificación de la parte de cargador de arranque previo comprende establecer todas las palabras de datos almacenadas en el área (121) de cargador de arranque en un mismo valor, de tal modo que el dispositivo integrado (1) comprende un microprocesador (15) diseñado para manejar estas palabras de datos que tienen este mismo valor que instrucciones neutras.

Fig. 1

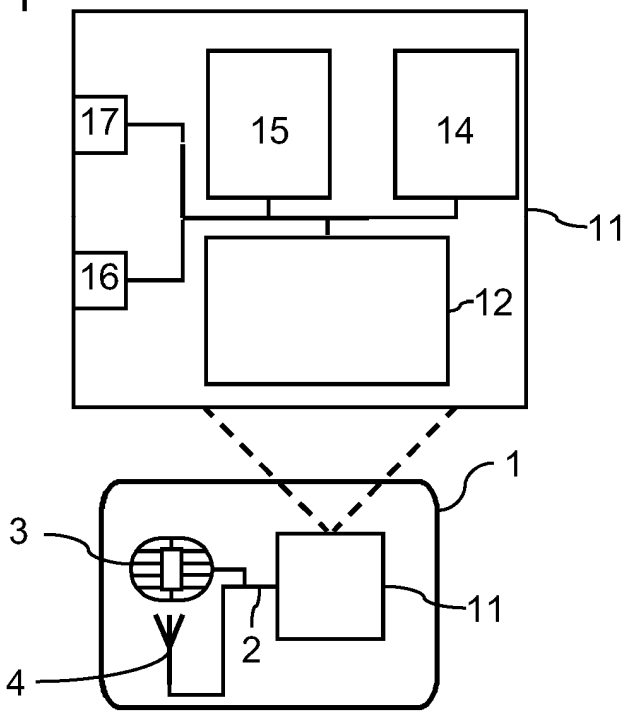

Fig. 2

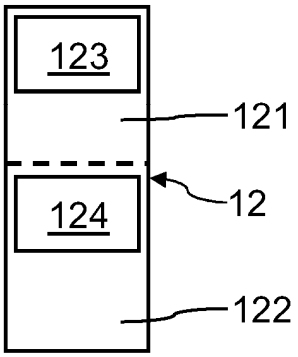

Fig. 3