(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 598 816**

(51) Int. CI.:

G06F 9/46 (2006.01)
G06F 11/14 (2006.01)
G06F 9/54 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **13.11.2012 PCT/IB2012/056370**

(87) Fecha y número de publicación internacional: **25.07.2013 WO13108090**

(96) Fecha de presentación y número de la solicitud europea: **13.11.2012 E 12866318 (4)**

(97) Fecha y número de publicación de la concesión europea: **14.09.2016 EP 2805236**

(54) Título: **Uso de una funcionalidad de interrupción con margen de advertencia por un programa**

(30) Prioridad:

18.01.2012 US 201213352514

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
30.01.2017

(73) Titular/es:

INTERNATIONAL BUSINESS MACHINES CORPORATION (100.0%)
New Orchard Road
Armonk, NY 10504, US

(72) Inventor/es:

GAINEY, CHARLES, JR.;
KUBALA, JEFFREY, PAUL;
FARRELL, MARK;
SCHMIDT, DONALD, WILLIAM;
MULDER, JAMES;
PIERCE, BERNARD y
ROGERS, ROBERT

(74) Agente/Representante:

DE ELZABURU MÁRQUEZ, Alberto

DESCRIPCIÓN

Uso de una funcionalidad de interrupción con margen de advertencia por un programa

Antecedentes

Un aspecto de esta invención se refiere, en general, al procesamiento dentro de un entorno informático y, en particular, a facilitar el procesamiento asociado a recursos compartidos.

Un entorno virtual, que incluye una unidad central de procesamiento (CPU, por sus siglas en inglés) anfitriona y una o varias unidades centrales de procesamiento invitadas, es un tipo de entorno en el cual se comparten recursos. Un programa anfitrión (por ejemplo, un sistema operativo anfitrión), que se ejecuta en una CPU anfitriona, provee una CPU invitada (también denominada CPU virtual). El programa anfitrión realiza acciones para asignar recursos procedentes de una configuración de anfitrión subyacente y para adjudicar esos recursos a la CPU invitada.

En una realización particular, existe una CPU invitada cuando una CPU anfitriona entra en el modo de ejecución interpretativa. En este momento, el sistema operativo invitado (también denominado en la presente memoria "programa invitado") comienza su ejecución en la CPU virtualizada, mientras que el programa anfitrión suspende su ejecución en la CPU anfitriona. El programa anfitrión reanuda su ejecución en la CPU cuando termina el modo de ejecución interpretativa. Existen técnicas de vinculación entre el anfitrión y el invitado mediante las cuales se guardan y se restauran los estados de anfitrión e invitado. Típicamente, cuando un programa anfitrión inicia un programa invitado, el programa anfitrión entra en suspensión hasta que el programa invitado retorna. Tanto la CPU invitada como la CPU anfitriona son modos distintos de la única CPU anfitriona.

La configuración de anfitrión incluye habitualmente todos los recursos del sistema informático. Estos recursos incluyen, pero sin limitación, unidades centrales de procesamiento (CPU), memoria principal y dispositivos de entrada/salida (E/S). En un sistema de este tipo, una sola CPU anfitriona puede dar soporte a múltiples CPU invitadas. Esto se logra asignando a cada CPU invitada un período de tiempo, denominado "porción de tiempo", para utilizar durante el mismo la CPU anfitriona, y trasladando después la CPU anfitriona a otra CPU invitada durante una porción de tiempo, y así sucesivamente. El número de CPU invitadas a las que puede dar soporte una CPU anfitriona varía dependiendo de las capacidades de la CPU anfitriona y de la capacidad deseada que se asigne a cada CPU invitada.

Una configuración con invitados se forma típicamente a partir de dos o más CPU invitadas, y se la conoce como una configuración de multiprocesamiento (MP) con invitados. Se puede proveer cada una de las CPU invitadas por compartición de una CPU anfitriona independiente o incluso por compartición de una única CPU anfitriona. Un atributo de tal compartición es que una CPU invitada puede funcionar durante un período de tiempo, denominado porción de tiempo y, a continuación, queda inactiva durante un cierto período arbitrario de tiempo. El período de inactividad varía dependiendo de las políticas de prioridad establecidas por el sistema, del número total de CPU invitadas que deban compartirse las CPU anfitrionas, y de la técnica de compartición particular que se utilice.

En un sistema de multiprocesamiento con invitados, de este tipo, el sistema operativo invitado puede despachar un programa, al que a veces se le denomina unidad despachable (DU, por sus siglas en inglés), a una CPU invitada y después, durante la ejecución de esa unidad despachable, la porción de tiempo de anfitrión de esa unidad despachable expira. Esto podría dejar a la unidad despachable en un estado tal que no se puede continuar en ninguna otra CPU invitada de la configuración de multiprocesamiento con invitados, independientemente de la disponibilidad de cualquier otra CPU invitada. En lugar de ello, para continuar debe esperar a que esa única CPU invitada reciba su siguiente porción de tiempo. Dependiendo de la técnica de compartición y de la prioridad relativa de la configuración con invitados, la siguiente porción de tiempo puede tardar un período sustancial de tiempo. Incluso aunque la configuración con invitados tenga otras CPU invitadas que puedan ejecutar la unidad despachable, no es posible continuar la unidad despachable debido al estado de la CPU invitada de la unidad despachable que se guardó cuando expiró la anterior porción de tiempo. Mientras no se pueda utilizar exactamente ese estado para continuar la CPU invitada, la unidad despachable permanece inactiva.

La patente de EE.UU. n.º 7,536,690 B2 (Alverson, G.A. *et al.*, "Deferred Task Swapping in a Multithreaded Environment", de 19 de mayo de 2009) describe un método y un sistema que preparan una tarea para que sea intercambiada y quede fuera de la utilización del procesador que está en ejecución en un ordenador con múltiples procesadores que soportan, cada uno, múltiples flujos.

La patente de EE.UU. n.º 5,594,893 (Byers, R.F. et *al.*, " System for Monitoring and Controlling Operation of Multiple Processing Units", de 14 de enero de 1997) describe un controlador de bastidor (en inglés, "shelf controller") de procesador que supervisa el estado del bastidor. Cuando surge una petición de apagado o de reinicio, bien sea iniciada por un controlador central o bien por el bastidor del procesador, el controlador de bastidor del procesador proporciona al menos un primer temporizador que concede al bastidor del procesador en funcionamiento tiempo suficiente para apagar procesos del sistema operativo.

Breve compendio

Al proporcionar un producto de programa informático para facilitar el procesamiento en un entorno informático se superan deficiencias de la técnica anterior y se confieren ventajas. El producto de programa informático incluye un medio de almacenamiento legible por ordenador que puede ser leído por un circuito de procesamiento y almacenar instrucciones para ser ejecutadas por el circuito de procesamiento, al objeto de poner en práctica un método. El método incluye, por ejemplo, que un programa obtenga una indicación de una funcionalidad de margen de advertencia instalada dentro del entorno informático, que la funcionalidad de margen de advertencia proporcione al programa un período de gracia de margen de advertencia para realizar una función; que el programa reciba una notificación de margen de advertencia que indica que el período de gracia de margen de advertencia ha comenzado; y que el programa, basándose en la notificación de margen de advertencia, al menos inicie la función dentro del período de gracia de margen de advertencia.

También se describen y reivindican en la presente memoria métodos y sistemas referentes a uno o varios aspectos de la presente invención. Además, también se describen y pueden reivindicarse en la presente memoria servicios relacionados con uno o varios aspectos de la presente invención.

Mediante las técnicas de la presente invención se obran características y ventajas adicionales. En la presente memoria se describen con detalle otras realizaciones y aspectos de la invención, y se consideran parte de la invención reivindicada.

Breve descripción de las diversas vistas de los dibujos

En las reivindicaciones que se encuentran al final de la memoria descriptiva se señalan en particular, y se reivindican específicamente, uno o varios aspectos de la presente invención. Lo que antecede, así como objetos, características y ventajas de la invención, resultan evidentes a partir de la descripción detallada que sigue, tomada conjuntamente con los dibujos adjuntos, en los cuales:

la Figura 1 representa una realización de un entorno informático que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 2 representa otra realización de un entorno informático que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 3 representa otra realización más de un entorno informático que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 4 representa una realización de la lógica asociada con la observación, por el invitado, de la funcionalidad de interrupción con margen de advertencia, según un aspecto de la presente invención;

la Figura 5 representa una realización de la lógica asociada con la observación, por el anfitrión, de la funcionalidad de interrupción con margen de advertencia, según un aspecto de la presente invención;

la Figura 6 representa una realización de la lógica asociada con un anfitrión que gestiona un abandono de invitado, según un aspecto de la presente invención;

la Figura 7 representa una realización de una panorámica de la lógica de la funcionalidad de interrupción con margen de advertencia, según un aspecto de la presente invención;

las Figuras 8A-8C representan realizaciones de la lógica asociada con el procesamiento de la funcionalidad de interrupción con margen de advertencia, según un aspecto de la presente invención;

la Figura 9 representa una realización de la lógica asociada con la recepción de una interrupción con margen de advertencia, según un aspecto de la presente invención;

la Figura 10 representa una realización de un formato de una instrucción "Diagnose" (diagnosticar) utilizada según un aspecto de la presente invención;

la Figura 11 representa una realización de un producto de programa informático que incorpora uno o varios aspectos de la presente invención;

la Figura 12 representa una realización de un sistema informático anfitrión que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 13 representa un ejemplo adicional de un sistema informático que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 14 representa otro ejemplo de un sistema informático que comprende una red informática que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 15 representa una realización de diversos elementos de un sistema informático que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 16A representa una realización de la unidad de ejecución del sistema informático de la Figura 15 que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 16B representa una realización de la unidad de bifurcación del sistema informático de la Figura 15 que incorpora y utiliza uno o varios aspectos de la presente invención;

la Figura 16C representa una realización de la unidad de carga/almacenamiento del sistema informático de la Figura 15 que incorpora y utiliza uno o varios aspectos de la presente invención; y

la Figura 17 representa una realización de un sistema informático anfitrión emulado que incorpora y utiliza uno o varios aspectos de la presente invención.

Descripción detallada

Según un aspecto de la presente invención, se proporciona una capacidad de advertir a un programa (por ejemplo, un sistema operativo) de que tiene un período de gracia durante el cual realizar una función. Por ejemplo, se concede un período de gracia a un programa para realizar una limpieza (por ejemplo, completar, detener y/o transferir una unidad despachable).

Según un aspecto adicional de la presente invención, se advierte a un programa y/o a un procesador que está a punto de perder acceso a recursos (por ejemplo, recursos compartidos). Por ejemplo, se emite una advertencia a un procesador que comparte recursos con otros procesadores, acerca de que el procesador está a punto de perder acceso a recursos. Como ejemplo adicional, a un programa, por ejemplo un sistema operativo, que se ejecuta en un procesador compartido (es decir, el programa comparte el procesador con otros programas) se le advierte de que está a punto de perder sus recursos de procesador.

En una realización particular, se proporciona una capacidad en la cual se emite, a un programa invitado que se ejecuta en una CPU invitada proporcionada por una CPU anfitriona, una advertencia de expiración de una porción de tiempo concedida a la CPU invitada desde la CPU anfitriona, o de apropiación por el anfitrión de la porción de tiempo del invitado. La advertencia provee un período de gracia que la CPU invitada puede utilizar para realizar una función determinada, por ejemplo completar la ejecución de una unidad despachable, detener la unidad despachable en un punto en el cual la unidad despachable sea re-despachable y/o transferir la unidad despachable a otra CPU invitada.

En la presente memoria, un período de gracia incluye como ejemplos una cantidad de tiempo, un número de instrucciones, un número de ciclos, etc. Tiene una duración predeterminada, dentro de la cual se pueden realizar una o más funciones.

Se describe, con referencia a la Figura 1, una realización de un entorno informático que incorpora y utiliza uno o varios aspectos de la presente invención. En esta realización particular, un entorno informático 100 incluye una pluralidad de procesadores 102 que comparten recursos 104. A cada procesador (y/o a un programa, por ejemplo un sistema operativo, que se ejecute en el procesador) se le concede una cierta cantidad de tiempo, denominada porción de tiempo, para compartir los recursos. A modo de ejemplo, los recursos incluyen recursos de la unidad central de procesamiento, memoria, dispositivos de entrada/salida o interfaces, y/u otros recursos. Al procesador (o a un programa que se ejecute en el mismo), que tiene acceso a los recursos, se le advierte de que su acceso está a punto de terminar y, por tanto, el procesador (o el programa) debe realizar una acción particular, por ejemplo limpiar, completar una unidad de trabajo, detener una unidad de trabajo, transferir una unidad de trabajo, etc.

Se describe, con referencia a la Figura 2, otra realización de un entorno informático 200 que incorpora y utiliza uno o varios aspectos de la presente invención. El entorno informático 200 se basa, por ejemplo, en la z/Architecture® ofrecida por International Business Machines Corporation (IBM®), Armonk, Nueva York. La z/Architecture® está descrita en una publicación de IBM® titulada "z/Architecture Principles of Operation", IBM® Publication n.º SA22-7832-08, novena edición, agosto de 2010. En un ejemplo, un entorno informático basado en la z/Architecture® incluye un servidor System z®, ofrecido por International Business Machines Corporation, Armonk, Nueva York. IBM®, z/Architecture® y zSeries®, así como z/VM® y z/OS® que se mencionan más adelante, son marcas comerciales registradas de International Business Machines Corporation, Armonk, Nueva York, EE.UU. Otros nombres utilizados en la presente memoria pueden ser marcas comerciales registradas, marcas comerciales o nombres de productos de International Business Machines Corporation o de otras compañías.

A modo de ejemplo, el entorno informático 200 incluye un complejo central de procesamiento (CPC, por sus siglas en inglés) 202 que proporciona soporte de máquina virtual. El CPC 202 incluye, por ejemplo, una o varias máquinas virtuales 204 (o en otra realización, particiones lógicas), uno o varios procesadores centrales 206, al menos un anfitrión 208 (por ejemplo, un programa de control, tal como un hipervisor), y un subsistema 210 de entrada/salida, cada uno de los cuales se describe a continuación. En este ejemplo, las máquinas virtuales y el anfitrión están incluidos en memoria.

El soporte de máquina virtual del CPC provee la capacidad de dirigir un gran número de máquinas virtuales, cada una de las cuales es capaz de albergar un sistema operativo invitado 212, por ejemplo z/VM®, z/OS® o Linux, a modo de ejemplo. Cada máquina virtual 204 puede funcionar como un sistema independiente. Es decir, cada máquina virtual puede ser reiniciada, albergar un sistema operativo invitado, y trabajar con distintos programas, de forma independiente. Un sistema operativo o programa de aplicación que se ejecute en una máquina virtual aparenta tener acceso a un sistema total y completo, pero, en realidad, solamente está disponible una parte del mismo.

Los recursos físicos del CPC (por ejemplo, unidades CPU, memoria, dispositivos E/S, etc.) son propiedad del anfitrión 208, y el anfitrión despacha los recursos físicos compartidos a los sistemas operativos invitados, según se requiera, a fin de cubrir sus demandas de procesamiento. El anfitrión controla las interacciones entre los sistemas operativos invitados y los recursos físicos de máquina compartidos, ya que el gran número de invitados normalmente impide que el anfitrión simplemente realice el reparto y asigne los recursos de hardware a los invitados configurados.

Los procesadores centrales 206 son recursos de procesador físico que se pueden asignar a una máquina virtual. Por ejemplo, la máquina virtual 204 incluye uno o varios procesadores lógicos, cada uno de los cuales representa la totalidad o una parte de un recurso 206 de procesador físico que puede ser asignado dinámicamente a la máquina virtual. El anfitrión 208 gestiona las máquinas virtuales 204. A modo de ejemplo, el anfitrión puede estar implementado en microcódigo que se ejecuta en procesadores 206, o bien puede ser una parte de un sistema operativo anfitrión que se ejecuta en la máquina. En un ejemplo, el anfitrión 208 es el Processor Resource/System Manager (PR/SM) ofrecido por International Business Machines Corporation, Armonk, Nueva York.

El subsistema 210 de entrada/salida dirige el flujo de información entre dispositivos y almacenamiento principal. Está acoplado al complejo central de procesamiento, de manera que puede ser parte del complejo central de procesamiento o bien ser independiente del mismo. El subsistema de E/S alivia a los procesadores centrales de la tarea de comunicarse directamente con los dispositivos de E/S acoplados al CPC y permite que el procesamiento de datos discurra de forma concurrente con el procesamiento de E/S.

En una realización, el *hardware*/*firmware* del anfitrión (por ejemplo, PR/SM) y del procesador (por ejemplo, System z®) interactúan entre sí de una manera cooperativa controlada, con el fin de procesar las operaciones del sistema operativo invitado sin requerir la transferencia de control desde/hacia el sistema operativo invitado y el anfitrión. Las operaciones de invitado se pueden ejecutar directamente sin intervención del anfitrión a través de una funcionalidad que permite ejecutar las instrucciones de forma interpretativa para un invitado. Esta funcionalidad proporciona una instrucción, "Start Interpretive Execution" (SIE, iniciar ejecución interpretativa), que puede ser emitida por el anfitrión, y que designa un bloque de control denominado "descripción de estado" que alberga el estado y los controles de invitado (máquina virtual). La instrucción coloca la CPU en un modo de ejecución interpretativa en el cual se procesan directamente las instrucciones e interrupciones de invitado, hasta que surge una situación que requiere la atención del anfitrión. Cuando se produce una situación de este tipo, se finaliza la ejecución interpretativa y, o bien se presenta una interrupción de anfitrión, o bien la instrucción SIE completa el almacenamiento de detalles de la situación encontrada; esta última acción se denomina intercepción. En "System/370 Extended Architecture/Interpretive Execution", IBM Publication n.º SA22-7095-01, de septiembre de 1985, se describe un ejemplo de ejecución interpretativa.

En la presente memoria, *firmware* incluye, por ejemplo, el microcódigo, el milicódigo y/o el macrocódigo del procesador. Incluye, por ejemplo, las instrucciones a nivel de *hardware* y/o estructuras de datos utilizadas en la implementación de código máquina de nivel superior. En una realización incluye, por ejemplo, código propietario que normalmente se entrega como microcódigo, que incluye *software* de confianza o microcódigo específico para el hardware subyacente y controla el acceso del sistema operativo al hardware del sistema.

En la Figura 3 se representa otro ejemplo de un entorno informático que incorpora uno o varios aspectos de la presente invención. En este ejemplo, se proporciona un sistema informático de anfitrión emulado 300 que emula un ordenador central 302 de una arquitectura de anfitrión. En el sistema informático de anfitrión emulado 300, un procesador (CPU) anfitrión 304 es un procesador anfitrión emulado (o procesador anfitrión virtual) y se realiza a través de un procesador 306 de emulación que tiene una arquitectura nativa de conjunto de instrucciones distinta de la utilizada por los procesadores del ordenador principal 302. El sistema informático de anfitrión emulado 300 tiene una memoria 308 a la que puede acceder el procesador 306 de emulación. En la realización de ejemplo, la memoria 308 está dividida en una parte 310 de memoria de ordenador anfitrión y una parte 312 de rutinas de emulación. La memoria 310 del ordenador anfitrión está disponible para programas del ordenador anfitrión emulado 302, conforme a la arquitectura del ordenador anfitrión, y puede incluir tanto un anfitrión o hipervisor 314 como una o varias máquinas virtuales 316 que ejecuten sistemas operativos invitados 318, análogos a los elementos de la Figura 2 que tienen nombres similares.

El procesador 306 de emulación ejecuta instrucciones nativas de un conjunto de instrucciones arquitecturado, con una arquitectura distinta de la del procesador emulado 304. Las instrucciones nativas se obtienen, por ejemplo, de la memoria 312 de rutinas de emulación. El procesador de emulación 306 puede acceder a una instrucción de anfitrión para ejecución desde un programa en la memoria 310 del ordenador anfitrión, mediante el empleo de una o varias instrucciones obtenidas de una rutina de secuencia y acceso/descodificación que pueden descodificar la o las

instrucciones de anfitrión a las que se ha accedido, para determinar una rutina de ejecución de instrucciones nativa con el fin de emular la función de la instrucción de anfitrión a la que se ha accedido. Una de estas instrucciones de anfitrión puede ser, por ejemplo, una instrucción "Start Interpretive Execution" (SIE), mediante la cual el anfitrión busca ejecutar un programa invitado en una máquina virtual. Las rutinas 312 de emulación pueden incluir soporte para esta instrucción, y para ejecutar una secuencia de instrucciones de invitado en una máquina virtual 316 conforme a la definición de esta instrucción SIE.

Las rutinas de funcionalidades arquitecturadas pueden emular otras funcionalidades que estén definidas para la arquitectura del sistema informático anfitrión 302, entre ellas funcionalidades tales como registros de uso general, registros de control, traducción dinámica de direcciones y soporte para subsistema de E/S y memoria caché de procesador, a modo de ejemplos. Las rutinas de emulación también pueden aprovechar funciones disponibles en el procesador 306 de emulación (por ejemplo, registros generales y la traducción dinámica de direcciones virtuales) con el fin de mejorar el rendimiento de las rutinas de emulación. También se pueden proporcionar *hardware* especial y motores de descarga para ayudar al procesador 306 a emular las funciones del ordenador anfitrión 302.

Según un aspecto de la presente invención, se proporciona una funcionalidad de interrupción con margen de advertencia que se puede utilizar en muchos tipos de entornos informáticos. Aunque se puede utilizar en muchos tipos de entornos, en la presente memoria se describen aspectos de la funcionalidad que hacen referencia a un sistema de multiprocesamiento con invitados. Tal como se ha descrito más arriba, en los sistemas de multiprocesamiento con invitados, sistemas operativos invitados despachan unidades (por ejemplo, programas, código, etc.) despachables a unidades centrales de procesamiento invitadas que se alojan en al menos una unidad central de procesamiento invitada. La CPU anfitriona proporciona una porción de tiempo (por ejemplo, una cantidad de tiempo u otro plazo, por ejemplo un número de instrucciones, un número de ciclos, etc.) a la CPU invitada, tiempo durante el cual la unidad despachable se ejecuta. Si, durante la ejecución de la unidad despachable, expira la porción de tiempo, la unidad despachable puede quedar en un estado en el que no pueda continuarse en ninguna otra CPU invitada de la configuración de multiprocesamiento con invitados, independientemente de la disponibilidad de cualesquiera otras CPU invitadas. En lugar de ello, debe esperar a que esa única CPU invitada reciba su siguiente porción de tiempo, para continuar. Dependiendo de la técnica de compartición particular empleada y de la prioridad relativa de la configuración con invitados, la siguiente porción de tiempo puede tardar un período sustancial de tiempo. Incluso aunque la configuración con invitados tenga otras CPU invitadas que puedan ejecutar la unidad despachable, no es posible continuar la unidad despachable debido del estado de la CPU invitada de la unidad despachable que se guardó cuando expiró la anterior porción de tiempo. Mientras no se pueda utilizar exactamente ese estado para continuar la CPU invitada, la unidad despachable permanece inactiva.

Es posible prolongar la porción de tiempo mediante la concesión de tiempo adicional (u otro plazo adicional, por ejemplo, instrucciones adicionales, ciclos adicionales, etc.), pero incluso con este tiempo extra, la CPU invitada podría prolongar la ejecución de la unidad despachable y seguir dejándola en el mismo estado indespachable en el que estaría al expirar la porción de tiempo normal.

Dado que un programa anfitrión no conoce los controles ni el estado utilizados por un programa invitado arbitrario que ejecuta una unidad despachable arbitraria, no se puede conceder siempre tiempo adicional para que el sistema operativo invitado limpie su unidad despachable sin establecer un protocolo entre el programa anfitrión y el programa invitado. Sin un protocolo, cualquier tiempo extra concedido a la CPU invitada sería consumido en el procesamiento principal y posiblemente seguiría concluyendo con la misma situación de atasco de unidad despachable. Por lo tanto, según un aspecto de la presente invención, se proporciona un protocolo de este tipo.

Según un aspecto de la presente invención, se proporciona un período o prolongación de gracia que incluye una advertencia que indica al programa invitado qué acción particular se debe tomar (por ejemplo, completar una unidad despachable o hacer re-despachable a la unidad despachable). A modo de ejemplo, se proporciona el período de gracia en respuesta a la expiración de la porción de tiempo, o en respuesta a la apropiación del invitado por el anfitrión, antes de la expiración de la porción de tiempo de aquél, con el fin de reclamar el procesador para alguna otra prioridad superior, desde el punto de vista del anfitrión, tal como se describe con mayor detalle en la presente memoria.

A modo de ejemplo, se proporciona el período de gracia en lugar de prolongar incondicionalmente la porción de tiempo. Si la porción de tiempo normal ha expirado por completo, se concede un período de gracia, pero con cargo a la siguiente porción normal de tiempo para seguir siendo justo con los demás invitados virtualizados que tienen cada uno una expectativa de porción de tiempo. Si no ha expirado la porción de tiempo normal, el período de gracia se toma del tiempo normal restante. En cualquier caso, el período de gracia limita el tiempo (u otro período) restante concedido a la CPU invitada y no es en sí mismo extensible. De este modo, la CPU invitada no puede continuar funcionando durante un período arbitrario y desconocido.

Al comienzo del período de gracia, se efectúa una notificación al programa invitado al objeto de que limpie la unidad despachable (por ejemplo, completándola, deteniéndola y/o transfiriéndola). La imposición del período de gracia asegura que la CPU invitada no exceda el plazo adicional concedido. El protocolo mediante el cual se concede un período de gracia al programa invitado y se le notifica que el tiempo (u otro plazo) casi ha vencido (es decir, que el período de gracia ha comenzado) es un acuerdo entre el programa invitado y el programa anfitrión de forma que el

programa invitado entiende el protocolo, con lo que hace que dicha notificación sea valiosa. Es decir, el programa invitado normalmente atenderá la notificación haciendo, cuando sea necesario, que la unidad despachable corriente en ese momento sea despachable en otra CPU invitada de la configuración con invitados (por ejemplo, transfiriéndola).

A continuación se describen detalles adicionales acerca del protocolo y de la funcionalidad de interrupción con margen de advertencia (también denominada "margen de advertencia" o "funcionalidad de margen de advertencia"), haciendo referencia a las Figuras 4-10. La realización descrita con referencia a estas figuras se refiere a un entorno virtual que tiene uno o varios invitados creados por uno o varios anfitriones. Sin embargo, uno o varios aspectos de la presente invención se refieren también a otros entornos, entre ellos entornos no virtuales en los que múltiples procesadores y/o múltiples programas comparten recursos.

Haciendo referencia a la Figura 4, se describen detalles acerca del protocolo de la funcionalidad de interrupción con margen de advertencia tal como lo observa un invitado. Tal como lo observa el invitado, el protocolo de margen de advertencia incluye, por ejemplo, una indicación de funcionalidad instalada, inscripción de invitados en registro, notificación y abandono voluntario, cada uno de los cuales se describe a continuación.

Haciendo referencia a la Figura 4, el programa invitado entiende el protocolo de margen de advertencia y busca una indicación de funcionalidad instalada, PASO 400. En un ejemplo, esta indicación es un bit almacenado en un bloque de control (por ejemplo, un Service Call Control Block (SCCB, bloque de control de llamada de servicio), que se observa utilizando un comando de lectura, por ejemplo un comando Read SCP Information (leer información de SCP). Después de determinar que la funcionalidad está instalada, el programa invitado realiza la inscripción en el registro, PASO 402. La inscripción en el registro es un mecanismo en el cual el programa invitado comunica al programa anfitrión que el programa invitado entiende el protocolo de funcionalidad de interrupción con margen de advertencia. En un ejemplo, la inscripción en el registro se lleva a cabo utilizando una instrucción Diagnose, de lo cual se describe un ejemplo más adelante.

En una realización, la inscripción en el registro iniciada desde cualquier unidad central de procesamiento invitada cubre todas las unidades centrales de procesamiento invitadas de la configuración de multiprocesamiento, ya que se desea un comportamiento consistente en todas las CPU invitadas de la configuración de multiprocesamiento con invitados. En una configuración de multiprocesamiento con invitados, las CPU invitadas utilizan la misma memoria principal, y se asume que las CPU invitadas funcionan en lo que a veces se denomina "imagen única". Por lo tanto, la inscripción de una CPU invitada en el registro es aplicable a las demás CPU invitadas del entorno de multiprocesamiento. En una realización, la inscripción en el registro es irrevocable, y ello ayuda a evitar ventanas de temporización, simplifica el desarrollo y permite una la facilidad de comprobación mejorada. Aunque la inscripción sea irrevocable, el programa invitado puede determinar si va a continuar participando en el protocolo. Si decide hacerlo, no necesita participar mediante un reinicio o teniendo que reiniciar uno o más indicadores de habilitación descritos más adelante.

Después de inscribir programa invitado en el registro del protocolo de margen de advertencia, se puede notificar un período de gracia a una CPU invitada, PASO 404. Por ejemplo, una CPU anfitriona puede advertirla de la expiración de una porción de tiempo (o, en otro ejemplo, de una expiración inminente) o de la apropiación de su porción de tiempo. En un ejemplo particular, se notifica a una CPU de una configuración con invitados registrados de, por ejemplo, la expiración de su porción de tiempo normal y del comienzo de un período de gracia que concede un plazo adicional para, por ejemplo, realizar una limpieza.

Después de la notificación, el invitado tiene una cantidad limitada de tiempo u otro plazo, el período de gracia (en una realización particular 50 microsegundos, por ejemplo), para hacer re-despachable a la unidad despachable, o para realizar otros ajustes apropiados. Si la porción normal de tiempo ya ha terminado, entonces antes de devolver el control al anfitrión, por ejemplo, se utiliza el período de gracia para hacer re-despachable a la unidad despachable o para realizar otros ajustes apropiados. Si la porción de tiempo no ha terminado, entonces se utiliza el período de gracia y se renuncia a cualquier porción restante de la porción de tiempo. Se produce una contabilidad normal del tiempo real utilizado por una CPU invitada.

Después de realizada la notificación, la CPU invitada se encuentra en un período restringido (por ejemplo, una cantidad limitada de tiempo) tras del cual se hace terminar involuntariamente el funcionamiento de la CPU invitada. Sólo se realiza una notificación por cada período de porción de tiempo normal. De este modo, la CPU invitada sigue estando limitada por un control de tiempo final que asegura que se pueda compartir en otro sitio la CPU anfitriona compartida subyacente, preservando así el buen orden y la disciplina en la virtualización global proporcionada por el programa anfitrión.

La notificación puede realizarse mediante cualquier mecanismo que provoque que el programa invitado detecte un estatus único. Los ejemplos incluyen una interrupción única de invitado, una posición de la memoria principal definida por arquitectura que se pueda fijar, o bien un dispositivo de memoria externa E/S a disposición tanto del anfitrión como del invitado. Lo primero requiere una habilitación de invitado adecuada, para permitir la interrupción. Los dos últimos requieren exámenes periódicos con una frecuencia suficiente para que no se desperdicie el período de gracia. En un ejemplo particular de la z/Architecture®, se utiliza como notificación una interrupción de invitado,

denominada interrupción con margen de advertencia (WTI, por sus siglas en inglés).

Después de ser advertido, el invitado abandona voluntariamente su porción de tiempo/período de gracia dado, PASO 406. El invitado termina su porción de tiempo/período de gracia corriente en ese momento después de haber sido notificado y haber hecho re-despachable a la unidad despachable (por ejemplo, haber detenido y transferido la unidad despachable, o haberla completado). Este abandono indica al programa anfitrión que el invitado está siguiendo efectivamente el protocolo. Pueden presentarse otros motivos para que un invitado renuncie al control, devolviéndolo así al programa anfitrión. Normalmente, en ninguno de dichos abandonos inusuales se presentarían condiciones para que el procesamiento restringido hiciese re-despachable a una unidad despachable. Si la CPU invitada abandona dentro del período de gracia a través del protocolo de interrupción con margen de advertencia, en la siguiente porción de tiempo, cuandoquiera que suceda, se envía una indicación de retroalimentación. De este modo, el programa invitado sabe que ha cumplido la limitación de tiempo impuesta por el período de gracia.

Si el invitado ha llegado tarde en su abandono voluntario, se le retira la ejecución al expirar el período de gracia. La siguiente vez que se arranca la CPU invitada, con una porción de tiempo normal, se le envía una indicación de retroalimentación para que el invitado sepa que ha llegado tarde. Por lo general, esto se puede utilizar para detectar problemas en el programa invitado, ya que habitualmente el período de gracia permite un tiempo suficiente para realizar la limpieza y salir voluntariamente.

Si se produce un abandono inusual, se espera que se produzca rápidamente el abandono voluntario en la siguiente vez que se arranque la CPU invitada dentro de una porción de tiempo normal. El mismo mecanismo de retroalimentación informaría al programa invitado de que se ha producido un abandono inusual y, así, proporcionaría diversa información para contribuir a determinar el problema.

El abandono voluntario se realiza mediante cualquier mecanismo que provoque que el control pase del programa invitado al programa anfitrión, y que incluya el mecanismo de retroalimentación antes mencionado. Para que el programa anfitrión reconozca la petición del invitado, el mecanismo utilizado debe estar definido en el protocolo de interrupción con margen de advertencia de esa arquitectura particular. En un ejemplo, este mecanismo incluye la instrucción Diagnose que se describe más adelante.

En una realización, además de que el invitado acate el protocolo de interrupción con margen de advertencia, el anfitrión también acata la funcionalidad, como se describe a continuación con mayor detalle, haciendo referencia a Figura 5.

Haciendo referencia a la Figura 5, el anfitrión reconoce la indicación de funcionalidad instalada y la refleja a sus invitados, PASO 500. Por ejemplo, el anfitrión comprueba el bit de "instalado" en el bloque de control (por ejemplo, SCCB) y reconoce el estatus de instalado del protocolo de margen de advertencia (es decir, que está establecido), y sabe cómo el programa anfitrión puede utilizarlo para beneficio del anfitrión. Por lo tanto, refleja a su invitado la indicación de la funcionalidad. Por ejemplo, para reflejar la funcionalidad a un invitado, el anfitrión establece un bit de instalado en un bloque de control de invitado (por ejemplo, SCCB de invitado) o en un área de memoria accesible al invitado. Si, por cualquier razón, el programa anfitrión no desea que un invitado observe el estatus de instalado del protocolo de funcionalidad de interrupción con margen de advertencia y no permite que un invitado observe su estatus de instalado y lo utilice, el programa anfitrión envía una indicación de "no instalado" al invitado (por ejemplo, pone a cero el bit visto por el invitado). Además, en una realización, el programa anfitrión establece los controles de las CPU invitadas de manera que se desactiva el protocolo de margen de seguridad (por ejemplo, apaga uno o más bits específicos en la descripción de estado de la CPU invitada).

Cuando un invitado inicia la inscripción en el registro, el programa anfitrión recibe la petición no solicitada de inscripción en el registro y recuerda que el invitado se ha registrado, PASO 502. Una petición de inscripción en el registro iniciada por una única CPU invitada es suficiente para inscribir en el registro todas las CPU invitadas de una configuración de multiprocesamiento con invitados. Así, el programa anfitrión habilita el protocolo de margen de advertencia para todas las CPU de la configuración con invitados, PASO 504. Por ejemplo, el programa anfitrión fija uno o más bits específicos de la descripción de estado de la CPU invitada, a fin de habilitar para los invitados la funcionalidad de interrupción con margen de advertencia. No es obligatorio que vuelva al invitado una retroalimentación del registro. Si una CPU invitada intentase inscribirse en el registro, incluso cuando la funcionalidad no está instalada, el anfitrión ignoraría la solicitud y no habilitaría a la CPU invitada para el protocolo de la funcionalidad de interrupción con margen de advertencia.

Después del registro y la habilitación de los invitados respecto a la funcionalidad de margen de advertencia, un invitado puede recibir una notificación de invocación del protocolo, PASO 506. Esto se puede efectuar en diversos escenarios, como se describe a continuación.

A modo de ejemplo, cuando se habilita el protocolo de interrupción con margen de advertencia para una CPU invitada que trabaja en el modo de ejecución interpretativa sobre una CPU anfitriona, por ejemplo la CPU anfitriona X, el programa anfitrión puede iniciar el protocolo desde la CPU anfitriona Y. Es decir, se ha creado una CPU invitada con una CPU anfitriona X y esa CPU anfitriona X no está disponible en ese momento para el programa anfitrión. Si el programa anfitrión tiene motivos para readquirir la CPU X, primeramente hace que la CPU X salga del

modo de ejecución interpretativa. Es decir, se detiene la CPU invitada, que sale así del modo de ejecución interpretativa de la CPU X. El hecho de detener la CPU invitada en cualquier punto arbitrario sin permitir que la CPU invitada se detenga voluntariamente crea el riesgo de un problema, que el protocolo de interrupción con margen de advertencia pretende resolver. El protocolo de interrupción con margen de advertencia permite a la CPU anfitriona Y pedir una notificación al permitir que una acción del programa anfitrión se transforme en una notificación en la CPU invitada X, PASO 506. Dado que el programa invitado se ha inscrito con anterioridad en el registro, el programa anfitrión espera que el programa invitado reconozca la notificación y apoye la gestión adecuada de la notificación, incluido el último paso de terminar voluntariamente la ejecución, devolviendo así el control de la CPU anfitriona X al programa anfitrión. Cuando esto ocurre, el programa anfitrión puede proceder a realizar en la CPU anfitriona X el uso, cualquiera que fuese, que ha originado la puesta en marcha del proceso.

El programa anfitrión notifica a un invitado, por ejemplo, estableciendo un estatus cualquiera, estableciendo un indicador (por ejemplo, un bit) o provocando el envío de una señal no solicitada, asíncrona (por ejemplo, una interrupción con margen de advertencia) al invitado. Aunque el invitado esté registrado, el momento de la recepción de una señal de notificación de este tipo sigue siendo desconocido para él. Al inscribirse en el registro, el invitado simplemente se ha comprometido a adherirse al protocolo, en caso de que así se le indique, y cuando llegue el momento.

En un sistema de uniprocesador anfitrión, si la única CPU anfitriona se encuentra en el modo de ejecución interpretativa, de manera que la CPU invitada está en funcionamiento, no existe otra CPU anfitriona para invocar el protocolo de interrupción con margen de advertencia. Sin embargo, incluso en este caso la propia CPU puede invocar el protocolo de interrupción con margen de advertencia cuando la CPU anfitriona, aun estando en el modo de ejecución interpretativa, reconoce la expiración de la porción de tiempo y puede conceder entonces un período de gracia y realizar la notificación.

En un ejemplo adicional de notificación, la notificación se produce cuando, debido a un cambio interno de estatus reconocido por la CPU anfitriona cuando se encuentra en el modo de ejecución interpretativa, la CPU anfitriona hace que se envíe a la CPU invitada la notificación definida en el protocolo de interrupción con margen de advertencia. Se da un ejemplo de ello cuando la CPU invitada está habilitada para el protocolo de interrupción con margen de advertencia y la CPU anfitriona reconoce el final de la porción de tiempo. Antes de indicar a la CPU invitada que renuncie al control, la CPU otorga internamente un período de gracia para conceder a la CPU invitada tiempo suficiente para recibir la señal, tomar las medidas apropiadas (por ejemplo, completar la unidad despachable corriente en ese momento o hacer re-despachable la unidad despachable corriente en ese momento) y terminar voluntariamente. Internamente, la CPU anfitriona retiene el estado que indica que la CPU invitada ha sido notificada. Si el invitado no termina voluntariamente dentro del período de gracia, la CPU lo reconoce, y da fin a la ejecución del invitado, devolviendo así el control al programa anfitrión al terminar la modalidad de ejecución interpretativa. En una realización, el invitado no tiene manera alguna de determinar por qué se ha invocado el protocolo, sino solamente que se le ha notificado que limpie y termine. Pueden existir otras lógicas de anfitrión para provocar el término de un modo de ejecución interpretativa y, así, poner fin a la ejecución invitada. Por ejemplo, existen escenarios en los que se debe detener la ejecución de todas las CPU invitadas, para realizar algún cambio coordinado en toda la configuración de invitados. Una configuración de multiprocesamiento con invitados no debe tener unidades CPU que trabajen con reglas o suposiciones distintas de otras CPU de la configuración invitada. Tal asimetría podría crear resultados impredecibles en los invitados.

La CPU anfitriona recibe el efecto de la CPU invitada que ha realizado el abandono voluntario o ha salido por cualquier otra razón (por ejemplo, por que se ha devuelto al anfitrión el recurso de la CPU), PASO 508. Si el abandono se ha debido a lo que define el protocolo de interrupción con margen de advertencia, entonces el programa anfitrión recuerda proporcionar una retroalimentación a la CPU invitada la siguiente vez que arranque, aunque transcurra mucho tiempo. Esta retroalimentación es una indicación positiva ("buena"), que asume que el invitado ha salido voluntariamente antes de la expiración del período de gracia. Si la salida se ha debido a cualquier otra razón, no se produce retroalimentación por el protocolo de interrupción con margen de advertencia en el siguiente arranque de la CPU invitada.

Si el invitado ha llegado tarde en el abandono voluntario; es decir, el invitado realiza una acción para salir voluntariamente, pero el período de gracia ha expirado, la ejecución de la CPU invitada queda apropiada al expirar el período de gracia. La siguiente vez que se arranca la CPU invitada con una porción de tiempo normal, se genera una indicación de retroalimentación por excepción, de manera que el invitado sepa que llegó tarde en su abandono voluntario. Generalmente, esto se puede utilizar para detectar problemas en el programa invitado, dado que el período de gracia usual concede tiempo suficiente para realizar la limpieza y salir voluntariamente.

Si se produce una salida que no es el abandono voluntario del protocolo de interrupción con margen de advertencia, la siguiente vez que se arranca la CPU invitada con una porción de tiempo normal, no se incluye ninguna retroalimentación conforme al protocolo de margen de advertencia.

Los medios para el abandono voluntario se implantan mediante cualquier mecanismo que provoque que el control pase desde el programa invitado al programa anfitrión, que sea reconocido por el programa anfitrión como el abandono voluntario previsto en el protocolo, y que incluya el mecanismo de retroalimentación antes mencionado.

En un ejemplo, se utiliza para el abandono voluntario una instrucción Diagnose. Es decir, para indicar el término de la porción de tiempo se utiliza la instrucción Diagnose con un parámetro particular. Después de que el programa de invitados emite y ejecuta la instrucción Diagnose, el programa anfitrión determina si la salida se realizó a tiempo. Entonces, cuando se arranca nuevamente el invitado, lo que se sitúa en la siguiente instrucción secuencial después de Diagnose, se provee un código de situación que indica si se realizó a tiempo. El código de situación se pone, por ejemplo, en la PSW de invitado que se utiliza para arrancar el invitado en la siguiente instrucción secuencial. El invitado puede entonces probar el código de condición.

Se describe con mayor detalle, haciendo referencia a la Figura 6, cómo gestiona el anfitrión el abandono voluntario del invitado. Inicialmente, cuando la CPU invitada se detiene el control vuelve a la CPU anfitriona, PASO 600. Se efectúa una determinación acerca de si el control fue devuelto dentro del período de gracia, PREGUNTA 602. Si el control fue devuelto dentro del período de gracia, entonces el programa anfitrión observa un abandono voluntario de un invitado, conforme al protocolo de interrupción con margen de advertencia, y recuerda una realimentación "buena" para el siguiente arranque de la CPU invitada, sin importar cuál sea la CPU anfitriona que provea en ese momento la CPU invitada, PASO 604. Esto es así, suponiendo que la funcionalidad de interrupción con margen de advertencia está instalada. En caso contrario, el estatus de realimentación no se recuerda. Sin embargo, si el invitado realiza una acción para abandonar voluntariamente, pero se encuentra fuera del período de gracia, PREGUNTA 602, entonces el programa anfitrión de la CPU anfitriona observa un abandono voluntario de invitado conforme al protocolo de interrupción con margen de advertencia (aunque ha sido fuera de plazo y se ha tenido que expulsar de manera no voluntaria) y recuerda una realimentación "mala" para el siguiente arranque de la CPU invitada, sin importar cuál sea la CPU anfitriona que provea en ese momento la CPU invitada, PASO 606. Una vez más, esto es así suponiendo que la funcionalidad de interrupción con margen de advertencia está instalada. En caso contrario, el estatus de realimentación no se recuerda.

Después de esto, tanto si el anfitrión está recordando una retroalimentación buena como si está recordando una retroalimentación mala, el programa anfitrión redirige la CPU anfitriona a una asignación de apropiación, PASO 608. Es decir, el anfitrión se redirige a realizar una o más funciones ahora que ha recuperado sus recursos (CPU).

Por otra parte, en el siguiente arranque secuencial de la CPU invitada, con independencia de cuál sea la CPU anfitriona que provea la CPU invitada, si se recuerda el estatus de retroalimentación, se establece la indicación de estatus de retroalimentación antes de arrancar la CPU invitada, PASO 610. En un ejemplo, esto se establece en la descripción de estado SIE, por ejemplo en la PSW de la descripción de estado que indica el comienzo de la siguiente instrucción secuencial.

Haciendo referencia a las Figuras 7-9 se describen detalles adicionales sobre el procesamiento asociado con la funcionalidad de interrupción con margen de advertencia. En particular, la Figura 7 representa una realización de la lógica asociada con una panorámica del procesamiento de la funcionalidad de interrupción con margen de advertencia; Las Figuras 8A-8C proporcionan detalles del procesamiento de la funcionalidad de interrupción con margen de advertencia, según un aspecto de la presente invención; y la Figura 9 representa una realización de la lógica asociada con la recepción de una interrupción con margen de advertencia.

Haciendo referencia a la Figura 7, inicialmente un programa invitado (por ejemplo, un sistema operativo invitado) reconoce que la funcionalidad de interrupción con margen de advertencia está instalada, PASO 700. En una realización, esto se consigue haciendo que el programa invitado observe un indicador de funcionalidad instalada (por ejemplo, un bit) que está situado, por ejemplo, en un bloque de control específico. Si el sistema operativo invitado tiene el soporte para participar en la funcionalidad de interrupción con margen de advertencia, reconoce el indicador de funcionalidad de interrupción con margen de advertencia instalada y luego indica su capacidad para participar en el protocolo. En un ejemplo, esto incluye registrar su intención de participar en el procesamiento con margen de advertencia, PASO 702. Tal como se describe en la presente memoria, en un ejemplo la inscripción en el registro se realiza a través de una instrucción Diagnose. Una vez registrado, el sistema operativo invitado indica tanto a la CPU anfitriona como al programa anfitrión que sabe cómo manejar una interrupción con margen de advertencia (WTI), que es una interrupción no ambigua que proporciona una advertencia al invitado de que, por ejemplo, está a punto de perder el acceso a su recurso compartido (por ejemplo, la CPU invitada) y que, por ejemplo, debe tomar medidas con respecto a su unidad despachable que está ejecutando en ese momento. En una realización, la inscripción en el registro es un requisito previo para recibir la WTI. Si el invitado no se ha inscrito en relación con la funcionalidad de interrupción con margen de advertencia, cuando expira la porción de tiempo del invitado no se le ofrece ningún período de gracia, y la CPU invitada es expulsada del modo de ejecución interpretativa.

En una realización, aunque está inscrito en el registro, el programa invitado cuenta con dos mecanismos para deshabilitar la presentación de la WTI. Por ejemplo, se puede poner a cero un bit seleccionado dentro de, por ejemplo, la palabra de estado de programa (PSW), lo que deshabilita la presentación de todas las interrupciones externas, entre ellas la WTI; o bien se puede poner a cero un bit dentro de un registro de control determinado (por ejemplo, CR0), para deshabilitar sólo la WTI. Si los dos bits son uno, la presentación de la WTI está habilitada. Si la presentación de una WTI permanece deshabilitada durante todo el período de gracia de la WTI, la ejecución del invitado termina sin el beneficio de la WTI, lo que constituye un abandono involuntario.

Durante la ejecución interpretativa de la CPU invitada, si la CPU invitada reconoce internamente, o bien una

situación de interrupción externa por temporizador de CPU anfitriona (por ejemplo, una porción de tiempo que ha expirado), o bien una apropiación solicitada por el programa anfitrión, PREGUNTA 704, el procesamiento interno de la CPU determina, antes de que el anfitrión reciba el control, si se debe realizar el procesamiento de la interrupción con margen de advertencia, PREGUNTA 706. Es decir, el procesamiento interno de la CPU comprueba que el invitado está habilitado para el procesamiento con margen de advertencia y, por lo tanto, determina si se debe incluir el procesamiento con margen de advertencia en el procesamiento a realizar. Si no se debe realizar el procesamiento de interrupción con margen de advertencia, entonces la ejecución interpretativa del invitado termina, PASO 708, y el control vuelve al programa anfitrión, PASO 710. Volviendo a la PREGUNTA 706, si se debe realizar, no obstante, el procesamiento de interrupción con margen de advertencia, entonces se realiza ese procesamiento, como se describe con mayor detalle más adelante, PASO 712.

Se describen, haciendo referencia a las Figuras 8A-8C, realizaciones de detalles adicionales del procesamiento de interrupción con margen de advertencia. En este procesamiento se utilizan varios indicadores de control, entre ellos los siguientes:

un control interno (por ejemplo, bit G) activo durante el período de gracia de la funcionalidad de interrupción con margen de advertencia, que no es visible arquitectónicamente, pero que es utilizado por la lógica interna de la CPU;

un control interno (por ejemplo, bit P), presentado durante una interrupción con margen de advertencia (WTI), que indica, cuando vale uno, que se ha presentado la WTI al invitado, y cuando vale cero indica que no se ha presentado. Al igual que el control interno activo durante el período de gracia de la funcionalidad de interrupción con margen de advertencia, el control interno de WTI presentada no es arquitectónicamente visible, pero es utilizado por la lógica interna de la CPU;

una apropiación del control, por el programa anfitrión, en detrimento del invitado (por ejemplo, un bit T), que es, por ejemplo, el indicador de petición de intervención de margen de advertencia en la descripción de estado de la CPU invitada; y

existe habilitación de interrupciones externas, cuando un indicador E vale uno. En un ejemplo, el indicador E es un bit dentro de la palabra actual de estado del programa (PSW).

Haciendo referencia a la Figura 8A, en un ejemplo, o bien se reconoce una situación de interrupción por temporizador de la CPU anfitriona (por ejemplo, ha expirado la porción de tiempo) o bien se reconoce una petición de intervención con margen de advertencia (por ejemplo, el anfitrión desea una devolución anticipada de recursos de CPU, es decir, antes de que acabe la porción de tiempo). Si se reconoce una situación de interrupción por temporizador de la CPU anfitriona, PREGUNTA 800, se realiza una determinación acerca de si se ha fijado el indicador de control de período de gracia activo (por ejemplo, ¿es G igual a 1?), PREGUNTA 802. Si no se ha fijado G, entonces se fija el indicador G en, por ejemplo, 1, PASO 804, y el período de gracia de la funcionalidad de interrupción con margen de advertencia está a punto de comenzar. Después se guarda el valor actual en ese momento del temporizador de la CPU anfitriona (en la presente memoria, al valor guardado se le denomina "valor original"), PASO 806, y se ajusta el temporizador de la CPU anfitriona para el período de gracia del margen de advertencia (por ejemplo, 50 microsegundos), PASO 808.

Después de esto, se realiza una determinación de si el invitado está habilitado para una interrupción con margen de advertencia, PREGUNTA 810. En una realización, si está activo el nivel 2 de invitados, que indica que un invitado ha dado inicio a otro invitado, entonces el invitado 2 sale del modo de ejecución interpretativa como en el caso de una interrupción de invitado 1, y se anula la instrucción Start Interpretive Execution del invitado 1. Por lo tanto, en este punto el procesamiento se realiza como invitado 1. Si no está activo el invitado 2, entonces el procesamiento simplemente continúa con el invitado 1. Si el invitado está habilitado para una WTI, entonces se presenta al invitado la interrupción externa con margen de advertencia (WTI), PASO 812. En un ejemplo, esta interrupción incluye un código particular de interrupción que se presenta, lo que indica que dispone de un período de gracia para realizar una o más funciones (por ejemplo, limpieza), si lo desea.

Además, se da a P el valor 1, lo que indica que se ha presentado la WTI, PASO 814. También se da el valor 1 al bit T, utilizando una función de actualización enclavada (puede ocurrir que ya valga 1 si originalmente se utilizó una petición de intervención), PASO 816. El período de gracia en el temporizador de la CPU anfitriona sigue su cuenta atrás, con independencia de que se haya presentado la WTI, PASO 818. Después se sale de este procesamiento, PASO 820. En un ejemplo, la consignación de haber salido de este proceso indica que la CPU ha completado el procesamiento, corriente en ese momento, de la funcionalidad de interrupción con margen de advertencia y está volviendo a otro procesamiento, según lo dictado por el estado de la CPU en ese momento.

Volviendo a la PREGUNTA 810, si el invitado no está habilitado para la interrupción con margen de advertencia, el procesamiento continúa en el paso 816. En este caso, el invitado no está habilitado para la WTI, por lo que ésta no se puede presentar al invitado. Sin embargo, se pone en espera el bit T, de manera que se pueda detectar más tarde, cuando se habilite el invitado para la WTI.

Volviendo a la PREGUNTA 800, si no se trata de una situación de interrupción por temporizador de la CPU anfitriona, entonces se reconoce una petición de intervención con margen de advertencia (es decir, apropiación por

el anfitrión). Es decir, el bit T dentro del campo de petición de intervención de la descripción de estado de invitado vale 1. Por tanto, se efectúa una determinación acerca de si se ha dado valor al indicador G, PREGUNTA 850. Si no se ha dado valor (por ejemplo, es 0), entonces el procesamiento continúa en el PASO 804. En este caso, el hecho de que T sea igual a 1 es la razón inicial para comenzar el proceso WTI. Sin embargo, si se ha dado valor al bit G, entonces se efectúa una determinación acerca de si se ha dado valor a P, PREGUNTA 852. Si no se ha dado valor a P (por ejemplo, es igual a 0), entonces el procesamiento continúa en el PASO 810, con un intento de presentar la WTI. Si, por el contrario, se ha dado valor a P (por ejemplo, no es igual a 0), entonces el descubrimiento de que T es igual a 1 después de que haya comenzado el período de gracia de la funcionalidad de margen de advertencia no tiene efecto alguno, y se abandona el proceso, PASO 854.

Volviendo a la PREGUNTA 802, si se ha dado valor a G (por ejemplo, es igual a 1), la CPU invitada ya ha estado funcionando dentro del período de gracia, y la expiración del temporizador de la CPU anfitriona indica que el período de gracia ha expirado. Así pues, se había iniciado previamente un ciclo de WTI y el período de gracia ha expirado. Por lo tanto, haciendo referencia a la Figura 8B, el valor original del temporizador de la CPU anfitriona guardado anteriormente ha disminuido en la cantidad de tiempo realmente utilizado durante el período de gracia, y después se carga en el temporizador de la CPU anfitriona, PASO 860. Se abandona el modo de ejecución de interpretativa, PASO 862, y se presenta al anfitrión la interrupción externa por temporizador de la CPU anfitriona, PASO 864 (esto es una forma de salida involuntaria de invitado).

Además de lo anterior, se puede iniciar el análisis de la WTI a través de ciertas instrucciones que pueden habilitar la CPU para la WTI. Por ejemplo, haciendo referencia a la Figura 8C, en principio una serie de instrucciones que puedan habilitar la CPU para la WTI, entre ellas, por ejemplo, una instrucción "Load PSW (Extended)" (cargar PSW (extendida)) y una instrucción "Store Then" (almacenar después) o "System Mask" (máscara de sistema) que pueden dar valor al bit designado de la PSW, y "Load Control" (control de carga) que pueden dar valor al bit seleccionado del registro de control, realizar seguimiento, como se describe en la presente memoria. Por ejemplo, una instrucción que puede habilitar para interrupciones comprueba el bit T en lo referente a un potencial procesamiento con margen de advertencia. Si T = 0, PREGUNTA 880, entonces no existe WTI, y se abandona este proceso, PASO 884. Sin embargo, si T = 1, entonces el proceso continúa a la PREGUNTA 822.

En la PREGUNTA 882, se efectúa una determinación de si se ha dado valor a P (por ejemplo, igual a 1). Si es así, entonces se abandona este proceso, PASO 884, ya que anteriormente se había detectado la habilitación. Sin embargo, si no se ha dado valor a P (por ejemplo, no es igual a 1), entonces se efectúa una determinación adicional en cuanto a si se ha dado valor a G (por ejemplo, igual a 1), PREGUNTA 886. Si no es así, entonces el procesamiento continúa al PASO 804 (Figura 8A). Si, por el contrario, se ha dado valor a G (por ejemplo, igual a 1), PASO 886 (Figura 8C), entonces el procesamiento continúa a la PREGUNTA 810 de la Figura 8A, PASO 888, y se abandona el procesamiento.

Se describen, con referencia a la Figura 9, detalles adicionales del procesamiento de la interrupción con margen de advertencia. Cuando el programa invitado recibe la interrupción con margen de advertencia, realiza todas las funciones (por ejemplo, funciones del sistema operativo (S.O.)) que deba realizar para, por ejemplo, hacer re-despachable a la unidad de trabajo despachable, PASO 900. Por ejemplo, el invitado detiene la unidad despachable en un punto determinado, guarda su estado y, o bien la traslada a otra CPU invitada o bien permite que sea trasladada al proporcionar información de estado, etc. El sistema operativo invitado indica que ha terminado mediante la emisión de una señal de "limpieza completa dentro del margen de advertencia" al programa anfitrión (es decir, un abandono voluntario), PASO 902. Esta señal puede ser cualquier mecanismo que haga que la operación invitada renuncie a la porción de tiempo restante. Sin embargo, debe ser reconocida por el programa anfitrión como la parte de limpieza del protocolo. En un ejemplo, se utiliza una función de "limpieza completada" de la instrucción Diagnose.

Si el programa de invitados emite la señal de "limpieza completada" antes de que expire el período de gracia, PREGUNTA 904, el programa anfitrión recuerda que la CPU invitada salió a tiempo, PASO 906. Se trata de un abandono voluntario. En la siguiente ocasión en que se arranca la CPU invitada, se indica de vuelta a la CPU invitada la naturaleza de "a tiempo" de la señal, PASO 908. En un ejemplo, se establece una PSW de reanudación de invitado para indicar un código de situación satisfactoria (por ejemplo, código de situación 0).

Volviendo a la PREGUNTA 904, si, por la razón que sea, el programa invitado tarda demasiado tiempo, el período de gracia expira por el hecho de que el temporizador de la CPU anfitriona ha ido reduciendo el período de gracia hasta cero, presentando así a la CPU una situación de interrupción externa por temporizador de CPU anfitriona. En este caso, la CPU reconoce que el invitado ya estaba dentro del período de gracia y no otorga otro período de gracia. En lugar de ello, se detiene la ejecución del invitado y el control vuelve al programa anfitrión, por recepción de la interrupción externa. Un programa anfitrión reconoce que esta conclusión de la CPU invitada es una salida de invitado involuntaria.

En el siguiente arranque de la CPU invitada, el sistema operativo invitado puede emitir entonces una señal de "limpieza completada", aunque ahora ya es demasiado tarde. El programa anfitrión ya no tiene expectativas de aguardar la recepción de la señal de limpieza completa. Así, en la siguiente ocasión en que se arranca la CPU invitada, se indica de vuelta a la CPU invitada la naturaleza de "demasiado tarde" de la señal, PASO 912. En un

ejemplo, se marca la PSW de reanudación de invitado para que indique una situación de retraso, que el invitado verá en el siguiente arranque. A la emisión de una instrucción Diagnose de "demasiado tarde" se la denomina a veces una instrucción Diagnose "prescrita", ya que anteriormente no consiguió salir dentro del período de gracia y luego salió sin motivo.

En un ejemplo, en un nuevo arranque de la CPU invitada, el programa invitado puede comprobar la parte del protocolo relativa a la señal de reanudación, para ver si la señal había sido emitida o no dentro del período de gracia. El programa invitado puede utilizar esta información para investigar por qué causa él mismo incurrió en retraso y realizar mejoras para mejorar las estadísticas de ser más puntual en el futuro.

En una realización, cuando el invitado se encuentra deshabilitado para todas las interrupciones externas, la supervisión la realizan una serie de instrucciones que pueden permitir interrupciones externas. Cuando el invitado está habilitado para interrupciones externas, se examina la habilitación para WTI. En este punto, si la WTI está habilitada y el bit P es 0, se presenta la WTI a la CPU invitada.

Como se ha mencionado más arriba, en una realización se utiliza una función Diagnose para indicar que se ha completado la limpieza o para inscribirse en el registro de la funcionalidad de interrupción con margen de advertencia. En cuanto a la indicación de limpieza completa, la función Diagnose, cuando se emite con un parámetro de limpieza y se ejecuta, indica que la CPU emisora ha realizado todo el procesamiento deseado asociado con la recepción de una interrupción externa con margen de advertencia. Cuando finaliza la ejecución, se establece un código de situación que indica si la conclusión se comunicó o no dentro del intervalo de tiempo, dependiente del modelo, permitido para la limpieza después de la interrupción con margen de advertencia.

En cuanto a la función de registro, la función Diagnose, cuando se emite con el parámetro de registro y se ejecuta, indica que la configuración de emisión entiende la interrupción con margen de advertencia. Cuando finaliza la ejecución, se establece un código de situación satisfactoria. El estatus de inscripción en el registro se borra mediante un reinicio del sistema.

Se describe, con referencia a la Figura 10, una realización de un formato de instrucción Diagnose. En una realización, una instrucción Diagnose 1000 incluye un código operativo 1002 que indica la función Diagnose; un primer campo de registro 1004 (R_1); un segundo campo de registro 1006 (R_3); un campo de registro general 1008 (B_2); y un campo de desplazamiento 1010 (D_2). En un ejemplo, se suma el contenido del campo D_2 al contenido de registro general B_2. No se utiliza el resultado para direccionar datos sino que, en lugar de ello, se utilizan ciertos bits (por ejemplo, los bits 48 a 63) como una extensión de código de operación. Cuando la extensión de código de operación es un valor predeterminado, se especifica que se ha completado la limpieza dentro del margen de advertencia y se renuncia a una porción de tiempo.

En un ejemplo, el campo R_3 está sin utilizar y contiene ceros. Además, los bits especificados de registro general R_1 están sin utilizar y deben contener ceros, y un bit particular de registro general R_1 (por ejemplo, el bit 63) especifica, cuando es cero, que la función de limpieza se ha completado y, cuando es 1, la función de registro.

En una partición lógica que utilice CPU físicas compartidas, esta función puede mejorar el rendimiento del sistema al permitir que la CPU física en la cual esté funcionando la CPU lógica sea asignada a otra CPU lógica.

Más allá de la Diagnose, cualquier otra salida de SIE durante el intervalo de gracia de la WTI, cualquiera que sea la causa, restaura de manera similar el valor original del temporizador de la CPU anfitriona disminuido en la cantidad de tiempo gastada en el período de gracia.

Se describe con detalle en la presente memoria una funcionalidad de interrupción con margen de advertencia que proporciona, en una realización, un mecanismo mediante el cual se puede presentar una interrupción externa con margen de advertencia a una CPU de una configuración con recursos de CPU compartidos, por ejemplo una partición lógica. El programa de control puede utilizar la interrupción externa con margen de advertencia como señal para hacer, a la unidad despachable que se esté ejecutando en ese momento, re-despachable sobre una CPU distinta dentro de la configuración.

En una realización, un procesador lógico (invitado) que se está ejecutando en una porción de tiempo en un procesador físico recibe una señal de advertencia que indica un período de gracia, por ejemplo, una cantidad de tiempo antes de que el procesador lógico sea interrumpido (desasignado del procesador físico que puede ser compartido), que permite completar o trasladar a otro procesador lógico el trabajo que el procesador lógico está realizando. A modo de ejemplo, se indica a la CPU invitada que su porción de tiempo ha expirado y que debe dar prioridad a la unidad de trabajo despachable (DU, por sus siglas en inglés) corriente en ese momento para hacerla re-despachable en otra CPU invitada. En un ejemplo, la señal de advertencia es una interrupción que tiene un código de interrupción que indica que se trata de una WTI. En un ejemplo adicional, el código de interrupción incluye información sobre la cantidad de tiempo u otro plazo concedido al período de gracia.

En una realización, la funcionalidad de interrupción con margen de advertencia se puede utilizar tanto en entornos no virtuales como en entornos virtuales, en los que un programa y/o procesador comparte recursos (por ejemplo, recursos de CPU u otros recursos) con uno o varios programas y/o procesadores distintos.

En una realización, en la cual el entorno es un entorno virtual, desde la perspectiva de un invitado:

1. El programa invitado observa la situación de "instalada" de la funcionalidad de protocolo de interrupción con margen de advertencia.

2. El programa invitado se inscribe en el registro del protocolo de interrupción con margen de advertencia.

3. La CPU invitada recibe una notificación de margen de advertencia, dependiendo de la arquitectura particular (por ejemplo, una indicación por memoria compartida, una indicación por dispositivo E/S compartido, una interrupción).

4. El programa invitado que se ejecuta en la CPU invitada realiza el procesamiento aplicable, conforme a la naturaleza del programa invitado que ha recibido la notificación (se espera que el procesamiento de la notificación sea único para cada sistema operativo).

5. La CPU invitada renuncia al control, conforme a la técnica voluntaria del protocolo de margen de advertencia protocolo.

6. En el siguiente arranque de la CPU invitada, el programa invitado puede observar la retroalimentación, conforme al protocolo de margen de advertencia.

Además, en una realización, desde la perspectiva del anfitrión:

A. El programa anfitrión observa la situación de "instalada" de la funcionalidad de protocolo de interrupción con margen de advertencia.

1. El programa anfitrión capta la indicación de "instalada" de la funcionalidad de protocolo de interrupción con margen de advertencia.

2. El programa anfitrión recuerda persistentemente el estatus de "instalado" del protocolo de interrupción con margen de advertencia.

3. El programa anfitrión indica el estatus de "instalado" del protocolo de interrupción con margen de advertencia a todas las configuraciones con invitados.

4. El programa anfitrión deshabilita el protocolo de margen de advertencia en todas las CPU invitadas no inscritas en el registro.

5. El programa anfitrión se prepara para reconocer una petición de inscripción en el registro de margen de advertencia para invitados, procedente de cada configuración con invitados.

B. El programa anfitrión reconoce una petición de inscripción en el registro para margen de advertencia, procedente de un invitado.

1. El programa anfitrión recuerda persistentemente que la configuración con invitados entiende el protocolo de margen de advertencia.

2. El programa anfitrión habilita al invitado para el protocolo de margen de advertencia.

C. Durante el funcionamiento normal de la CPU invitada X, se utiliza la apropiación de la CPU invitada X para recuperar la CPU anfitriona X correspondiente.

1. El programa anfitrión de la CPU Y indica la notificación a la CPU invitada X.

a. La CPU X propaga la notificación a la CPU invitada X por medio de actualización de la ubicación de memoria compartida, actualización de dispositivo E/S compartido o interrupción a la CPU invitada X, conforme al protocolo de margen de advertencia.

D. La CPU invitada X se detiene, devolviendo el control a la CPU anfitriona X.

1. Si se está dentro del período de gracia, el programa anfitrión de la CPU X observa el abandono voluntario del invitado conforme al protocolo de margen de advertencia y recuerda una retroalimentación "buena" para el próximo arranque de la CPU invitada X, con independencia de cuál sea la CPU anfitriona que pueda estar proveyendo la CPU invitada X en ese momento.

a. Si se está dentro del período de gracia, pero la salida de la CPU invitada X no es conforme al protocolo de margen de advertencia, no se recuerda estatus de retroalimentación.

2. Si no se está dentro del período de gracia, el programa anfitrión de la CPU X observa el abandono voluntario del invitado conforme al protocolo de margen de advertencia y recuerda una retroalimentación "mala" para el próximo arranque de la CPU invitada X, con independencia de cuál sea la CPU anfitriona que pueda estar proveyendo la

CPU invitada X en ese momento.

a. Si no se está dentro de período de gracia, pero la salida de la CPU invitada X no es conforme al protocolo de margen de advertencia, no se recuerda estatus de retroalimentación.

3. El programa anfitrión de la CPU X redirige la CPU X para apropiarse de la asignación.

E. En el siguiente arranque secuencial de la CPU invitada X, con independencia de cuál sea la CPU anfitriona que provea la CPU invitada X, si se recuerda el estatus de retroalimentación, establecer la indicación de retroalimentación, conforme al protocolo de margen de advertencia, antes de arrancar la CPU invitada X.

En una realización, un procesador invitado de la configuración con invitados recibe una interrupción única, estando definida esa interrupción para una arquitectura informática, y siendo el significado de la interrupción una interrupción con margen de advertencia. La interrupción indica un código específico que identifica a la interrupción como una interrupción con margen de advertencia. La interrupción implica un intervalo de tiempo relativamente corto, denominado período de gracia, que conduce a finalizar la ejecución de un procesador invitado.

En un ejemplo, durante el período de gracia se espera nominalmente que el programa invitado haga re-despachable en otro procesador invitado la unidad despachable de trabajo en ese momento, evitando así que quede atascada en el procesador invitado en ese momento, a la espera de su siguiente arranque con porción de tiempo normal desde el anfitrión.

En un ejemplo, el intervalo de tiempo relativamente corto se concede una sola vez por cada arranque del procesador invitado originado por el programa anfitrión. El intervalo de tiempo se concede, por ejemplo, tomándolo del intervalo de tiempo existente dentro del cual se está ejecutando procesador invitado. Puesto que el intervalo de tiempo concedido se asigna tomándolo de la porción de tiempo normal restante, no se trata en sí de un préstamo de tiempo, sino que se está utilizando una cantidad limitada de tiempo tomada del intervalo corriente de tiempo, para asegurar la apropiación efectiva del procesador invitado en un período de tiempo relativamente corto.

En un ejemplo adicional en el cual ha expirado la porción de tiempo corriente, el intervalo de tiempo se concede como tiempo extra añadido al intervalo de tiempo existente dentro del cual se está ejecutando el procesador invitado. El programa anfitrión contabiliza el intervalo de tiempo concedido con cargo al siguiente intervalo de tiempo normal secuencial esperado, que será consumido por el procesador invitado dentro del cual se espera que se continúe ejecutando el procesador invitado. Sigue teniendo por objeto asegurar la apropiación efectiva del procesador invitado en un período de tiempo relativamente corto.

En un ejemplo, se puede generar una petición de interrupción para un suceso de margen de advertencia con el fin de informar al programa de que se está acercando el final del intervalo de ejecución corriente en ese momento, en una CPU compartida. La petición de interrupción es un tipo de situación pendiente que se genera cuando se inscribe la configuración en el registro y se la habilita para la funcionalidad de interrupción con margen de advertencia.

El procesamiento cooperativo entre los programas (por ejemplo, anfitrión e invitado) optimiza la distribución de recursos (por ejemplo, la CPU) entre los programas (por ejemplo, los sistemas operativos invitados). Por ejemplo, uno o varios aspectos proporcionan un mejor tiempo de respuesta con la misma utilización de CPU. Además, se libera la serialización del sistema antes de que quede sin despachar por el hipervisor.

En una realización adicional, se pueden utilizar uno o varios aspectos de la invención con peticiones de un sistema operativo para permitir que un hilo de ejecución individual continúe, a fin de mejorar el tiempo transcurrido de trabajo con sensibilidad temporal. Es decir, un hilo puede solicitar tiempo adicional, o puede concedérsele, para realizar una función.

Como un experto en la técnica apreciará, uno o varios aspectos de la presente invención pueden ponerse en práctica en forma de un sistema, método o producto de programa informático. En consecuencia, uno o varios aspectos de la presente invención pueden tomar la forma de una realización completamente en *hardware*, una realización totalmente en *software* (incluidos *firmware*, *software* residente, microcódigo, etc.) o una realización que combine *software* y *hardware,* aspectos todos a los que se puede hacer referencia en la presente memoria, de manera general, como "circuito", "módulo" o "sistema". Además, uno o varios aspectos de la presente invención pueden tomar la forma de un producto de programa informático incorporado en uno o varios medios legibles por ordenador que tengan incorporado en los mismos código de programa legible por ordenador.

Se puede utilizar cualquier combinación de uno o varios medios legibles por ordenador. El medio legible por ordenador puede ser un medio de almacenamiento legible por ordenador. Un medio de almacenamiento legible por ordenador puede ser, por ejemplo, pero sin limitación, un sistema, aparato o dispositivo electrónico, magnético, óptico, electromagnético, infrarrojo o semiconductor, o cualquier combinación adecuada de los anteriores. Los ejemplos más específicos (lista que no es exhaustiva) de medio de almacenamiento legible por ordenador incluyen los siguientes: una conexión eléctrica que tiene uno o más cables, un disquete portátil de ordenador, un disco duro, una memoria de acceso aleatorio (RAM, por sus siglas en inglés), una memoria de sólo lectura (ROM), una memoria de sólo lectura, programable y borrable (EPROM o memoria *flash*), una fibra óptica, una memoria de sólo lectura en

disco compacto portátil (CD-ROM), un dispositivo de almacenamiento óptico, un dispositivo de almacenamiento magnético, o cualquier combinación adecuada de lo anterior. En el contexto de la presente memoria, un medio de almacenamiento legible por ordenador puede ser cualquier medio tangible que puede contener o almacenar un programa para uso por, o en conexión con, un sistema, aparato o dispositivo de ejecución de instrucciones.

Haciendo referencia a la Figura 11, en un ejemplo un producto 1100 de programa informático incluye, por ejemplo, uno o varios medios 1102 de almacenamiento legibles por ordenador, no transitorios, para almacenar en los mismos medios o lógica 1104 de código de programa legible, con el fin de proporcionar y facilitar uno o varios aspectos de la presente invención.

Utilizando un medio apropiado que incluye, pero sin limitación, la tecnología inalámbrica, cableada, por cable de fibra óptica, por RF, etc., o cualquier combinación adecuada de lo anterior, se puede transmitir código de programa incorporado en un medio legible por ordenador.

El código de programa informático para llevar a cabo operaciones con respecto a uno o varios aspectos de la presente invención puede estar escrito en cualquier combinación de uno o varios lenguajes de programación, entre ellos un lenguaje de programación orientado a objetos, por ejemplo Java, Smalltalk, C++ o similares, y lenguajes de programación procedimentales convencionales, tales como el lenguaje de programación "C", ensamblador o lenguajes de programación similares. El código de programa se puede ejecutar en su totalidad en el ordenador del usuario, parcialmente en el ordenador del usuario, como un paquete de software independiente, parcialmente en el ordenador del usuario y parcialmente en un equipo remoto, o bien en su totalidad en el ordenador o servidor remoto. En este último escenario, el ordenador remoto puede estar conectado al ordenador del usuario a través de cualquier tipo de red, entre ellas una red de área local (LAN) o una red de área extensa (WAN), o bien se puede efectuar una conexión a un ordenador externo (por ejemplo, a través de la Internet utilizando un proveedor de servicios de Internet).

En la presente memoria se describen uno o varios aspectos de la presente invención haciendo referencia a ilustraciones de diagrama de flujo y/o diagramas de bloques de métodos, aparatos (sistemas) y productos de programa informático según realizaciones de la invención. Se entenderá que cada uno de los bloques de las ilustraciones de diagramas de flujo y/o diagramas de bloques, y las combinaciones de bloques en las ilustraciones de diagramas de flujo y/o diagramas de bloques, puede ser implementado mediante instrucciones de programa informático. Se pueden proporcionar estas instrucciones de programa informático a un procesador de un ordenador de uso general, un ordenador de propósito especial u otro aparato de procesamiento de datos programable para producir una máquina, de manera que las instrucciones que se ejecutan a través del procesador del ordenador u otro aparato de procesamiento de datos programable crean medios para implementar las funciones o acciones especificadas en el bloque o los bloques del diagrama de flujo y/o del diagrama de bloques.

Estas instrucciones de programa informático también se pueden almacenar en un medio legible por ordenador que puede dirigir un ordenador, otro aparato de procesamiento de datos programable, u otros dispositivos, para que funcionen de una manera particular, de manera que las instrucciones almacenadas en el medio legible por ordenador produzcan un artículo de fabricación que incluya instrucciones que implementan la función o la acción especificadas en el bloque o los bloques del diagrama de flujo y/o del diagrama de bloques.

Las instrucciones de programa informático también pueden estar cargadas en un ordenador, otro aparato de procesamiento de datos programable, u otros dispositivos, para hacer que una serie de pasos operativos que han de realizarse en el ordenador, otro aparato programable u otros dispositivos, produzcan un proceso implementado por ordenador de manera que las instrucciones que se ejecutan en el ordenador u otro aparato programable proporcionen procesos para implementar las funciones o acciones especificadas en el bloque o los bloques del diagrama de flujo y/o del diagrama de bloques.

Los diagramas de flujo y de bloques de las Figuras ilustran la arquitectura, operatividad y funcionamiento de posibles implementaciones de sistemas, métodos y productos de programa informático según diversas realizaciones de uno o varios aspectos de la presente invención. En este sentido, cada bloque de los diagramas de flujo o de bloques puede representar un módulo, segmento o porción de código, que comprenda una o varias instrucciones ejecutables para la implementación de la función o funciones lógicas especificadas. También hay que señalar que, en algunas implementaciones alternativas, las funciones escritas en el bloque pueden producirse fuera del orden escrito en las Figuras. Por ejemplo, dos bloques mostrados en sucesión pueden ser ejecutados, de hecho, de manera sustancialmente simultánea, o bien a veces se pueden ejecutar los bloques en orden inverso, dependiendo de la operatividad en cuestión. También se señalará que cada bloque de la ilustración de diagrama de bloques y/o diagrama de flujo, y las combinaciones de bloques en la ilustración de diagrama de bloques y/o diagrama de flujo, puede ser implementado por sistemas basados en *hardware* de propósito especial que realicen las funciones o acciones especificadas, o por combinaciones de hardware de propósito especial e instrucciones informáticas.

Además de lo anterior, un proveedor de servicios que ofrezca gestión de entornos de cliente puede proporcionar, ofrecer, desplegar, administrar, atender, etc., uno o varios aspectos de la presente invención. Por ejemplo, el proveedor de servicio puede crear, mantener, dar servicio, etc. a código de ordenador y/a o una infraestructura informática que lleva a cabo uno o varios aspectos de la presente invención para uno o varios clientes. A cambio, el

proveedor de servicios puede recibir pago por parte del cliente en virtud de un contrato de abono y/o según tarifa, a modo de ejemplos. De manera adicional o como alternativa, el proveedor de servicios puede recibir pago por la venta de contenido publicitario a uno o varios terceros.

En un aspecto de la presente invención, se puede desplegar una aplicación para llevar a cabo uno o varios aspectos de la presente invención. A modo de ejemplo, el despliegue de una aplicación comprende proporcionar infraestructura informática que se pueda hacer funcionar para llevar a cabo uno o varios aspectos de la presente invención.

Como aspecto adicional de la presente invención, se puede desplegar una infraestructura informática que comprenda la integración de código legible por ordenador en un sistema informático, en donde el código en combinación con el sistema informático es capaz de llevar a cabo uno o varios aspectos de la presente invención.

Como otro aspecto adicional más de la presente invención, se puede proporcionar un procedimiento para integrar infraestructura informática, que comprende integrar código legible por ordenador en un sistema informático. El sistema informático comprende un medio legible por ordenador, en el cual el medio informático comprende uno o varios aspectos de la presente invención. El código, en combinación con el sistema informático, es capaz de llevar a cabo uno o varios aspectos de la presente invención.

Aunque se han descrito en lo que antecede diversas realizaciones, estas constituyen solamente ejemplos. Por ejemplo, entornos informáticos de otras arquitecturas pueden incorporar y utilizar uno o varios aspectos de la presente invención. Además, el período de gracia puede ser distinto de una cantidad de tiempo, por ejemplo un número de instrucciones o ciclos, o cualquier otro valor cuantificable. Se pueden efectuar muchos cambios y/o adiciones sin salir del alcance de la presente invención.

Además, otros tipos de entornos informáticos pueden beneficiarse de uno o varios aspectos de la presente invención. A modo de ejemplo, se puede utilizar un sistema de procesamiento de datos adecuado para almacenar y/o ejecutar de código de programa, que incluya al menos dos procesadores acoplados directa o indirectamente a elementos de memoria a través de un bus de sistema. Los elementos de memoria incluyen, por ejemplo, memoria local empleada durante la ejecución real del código de programa, almacenamiento masivo, y memoria caché que proporcione almacenamiento temporal de al menos cierto código de programa con el fin de reducir el número de veces que, durante la ejecución, deba recuperarse código desde almacenamiento masivo.

Se pueden acoplar al sistema, ya sea directamente o a través de controladores de E/S intermedios, dispositivos de entrada/salida (E/S) que incluyen, pero sin limitación, teclados, pantallas de visualización, dispositivos señaladores, DASD, cinta, CD, DVD, unidades extraíbles y otros medios de memoria, etc.). También se pueden acoplar al sistema adaptadores de red para permitir que el sistema de procesamiento de datos se acople a otros sistemas de procesamiento de datos o impresoras remotas o dispositivos de almacenamiento a través de redes privadas o públicas intermedias. Módems, módems de cable, y tarjetas Ethernet son sólo algunos de los tipos de adaptadores de red disponibles.

Se describen a continuación otros ejemplos de entornos informáticos que pueden incorporar y/o utilizar uno o varios aspectos de la presente invención.

Haciendo referencia a la Figura 12, se delinean componentes representativos de un sistema informático anfitrión 5000 que implementa uno o varios aspectos de la presente invención. El ordenador anfitrión representativo 5000 comprende una o varias CPU 5001 en comunicación con la memoria del ordenador (es decir, el almacenamiento central) 5002, así como interfaces de E/S hacia dispositivos 5011 de medios de almacenamiento y redes 5010 para comunicar con otros ordenadores o redes SAN, y similares. La CPU 5001 es compatible con una arquitectura que tiene un conjunto de instrucciones dotadas de arquitectura y operatividad dotada de arquitectura. La CPU 5001 puede poseer traducción dinámica de direcciones (DAT, por sus siglas en inglés) 5003 para transformar direcciones de programa (direcciones virtuales) en direcciones reales de memoria. Una DAT incluye típicamente un tampón de traducción lateral (TLB) 5007 para almacenar en memoria caché traducciones, de modo que el acceso posterior al bloque 5002 de memoria de ordenador no origine el retraso de la traducción de direcciones. Se emplea típicamente una memoria caché 5009 entre la memoria 5002 de ordenador y el procesador 5001. La memoria caché 5009 puede ser jerárquica, con una gran memoria caché disponible para más de una CPU y memorias caché más pequeñas, más rápidas (de nivel inferior) entre la memoria caché grande y cada CPU. En algunas implementaciones, las memorias caché de nivel inferior se dividen, a fin de proporcionar cachés de bajo nivel separadas para extraer instrucciones y acceder a datos. En una realización, una unidad 5004 de extracción de instrucciones extrae de la memoria 5002, a través de una memoria caché 5009, una instrucción. En una unidad 5006 de descodificación de instrucciones se descodifica la instrucción y se despacha (en algunas realizaciones, junto con otras instrucciones) a la unidad o unidades 5008 de ejecución de instrucciones. Típicamente se emplean varias unidades de ejecución 5008, por ejemplo una unidad de ejecución aritmética, una unidad de ejecución de coma flotante y una unidad de ejecución de instrucción de bifurcación. La unidad de ejecución ejecuta la instrucción, accediendo a operandos de registros de instrucción especificados o a la memoria, según se requiera. Si se precisa acceder a un operando (cargado o almacenado) desde la memoria 5002, normalmente una unidad 5005 de carga/almacenamiento maneja el acceso bajo el control de la instrucción que se está ejecutando. Las instrucciones pueden ser ejecutadas en

circuitos de *hardware* o en microcódigo interno (*firmware*) o mediante una combinación de ambas cosas.

Como se ha indicado, un sistema informático incluye información en el almacenamiento local (o principal), así como direccionamiento, protección y grabación de referencia y de cambios. Algunos aspectos del direccionamiento incluyen el formato de las direcciones, el concepto de espacios de direcciones, los diversos tipos de direcciones y la manera en que un tipo de dirección se traduce a otro tipo de dirección. Algunos de los almacenamientos principales incluyen ubicaciones de almacenamiento asignadas permanentemente. El almacenamiento principal proporciona al sistema el almacenamiento rápido de datos, con acceso directamente direccionable. Se deben cargar en el almacenamiento principal (desde dispositivos de entrada) tanto los datos como los programas, antes de que puedan ser procesados.

El almacenamiento principal puede incluir uno o varios almacenamientos intermedios más pequeños, con acceso más rápido, denominados a veces memorias caché. Típicamente, una memoria caché está asociada físicamente con una CPU o un procesador de E/S. Los efectos, salvo sobre el rendimiento, de la construcción física y del uso de distintos medios de almacenamiento no son observables generalmente por el programa.

Se pueden mantener memorias cachés separadas para instrucciones y operandos de datos. La información dentro de una memoria caché se mantiene en bytes contiguos dentro de una frontera integral denominada bloque de caché o línea de caché (o línea, para abreviar). Un modelo puede proporcionar una instrucción EXTRACT CACHE ATRIBUTE (extraer atributo de caché) que devuelve el tamaño en bytes de una línea de caché. Un modelo puede proporcionar también instrucciones PREFETCH DATA (pre-extraer datos) y PREFETCH DATA RELATIVE LONG (pre-extraer datos relativo largo), que realizan la pre-extracción de almacenamiento a la memoria caché de datos o de instrucciones o bien la liberación de los datos desde la memoria caché.

El almacenamiento se visualiza como una larga cadena horizontal de bits. Para la mayoría de las operaciones, los accesos al almacenamiento discurren en una secuencia de izquierda a derecha. La cadena de bits se subdivide en unidades de ocho bits. A una unidad de ocho bits se la denomina *byte*, que es el elemento básico con el que se construyen todos los formatos de información. Cada ubicación de *byte* en el almacenamiento está identificada por un número entero no negativo único, que es la dirección de esa ubicación de *byte* o, llanamente, la dirección de *byte*. Las ubicaciones de *byte* adyacentes tienen direcciones consecutivas, comenzando por 0 a la izquierda y continuando en una secuencia de izquierda a derecha. Las direcciones son números enteros binarios sin signo y tienen 24, 31 o 64 bits.

La información entre el almacenamiento y una CPU o un subsistema de canal se transmite a razón de un *byte,* o un grupo de *bytes*, a la vez. Salvo que se especifique otra cosa, en la z/Architecture®, por ejemplo, a un grupo de *bytes* del almacenamiento se le asigna una dirección por el byte más a la izquierda del grupo. El número de *bytes* en el grupo, o bien es implícito, o bien lo especifica explícitamente la operación a realizar. Cuando se utiliza en una operación de la CPU, a un grupo de *bytes* se le denomina "campo". Dentro de cada grupo de *bytes* en la z/Architecture®, por ejemplo, los bits se numeran en una secuencia de izquierda a derecha. En la z/Architecture®, a los bits más a la izquierda se les denomina a veces bits "de orden alto" y a los bits a la derecha se les denomina bits "de orden bajo". Sin embargo, los números de bit no son direcciones de almacenamiento. Solamente se pueden asignar direcciones a los *bytes*. Para operar sobre bits individuales de un *byte* en el almacenamiento, se accede a todo el *byte*. Los bits de un *byte* están numerados de 0 a 7, de izquierda a derecha (por ejemplo, en la z/Architecture®). A los bits de una dirección se les puede numerar 8-31 o 40-63 en direcciones de 24 bits, o 1-31 o 33-63 en direcciones de 31 bits; se numeran 0-63 en direcciones de 64 bits. Dentro de cualquier otro formato de longitud fija de múltiples *bytes*, los bits que componen el formato son numerados consecutivamente comenzando por 0. Con el objeto de detectar errores, y en particular para corregirlos, se pueden transmitir uno o más bits de comprobación con cada *byte* o con un grupo de *bytes*. Dichos bits de control son generados automáticamente por la máquina y no pueden ser controlados directamente por el programa. Las capacidades de almacenamiento se expresan en número de bytes. Cuando la longitud de un campo de operando de almacenamiento está implícita en el código de operación de una instrucción, se dice que el campo tiene una longitud fija, que puede ser uno, dos, cuatro, ocho o dieciséis *bytes*. Pueden estar implícitos campos más grandes para algunas instrucciones. Cuando la longitud de un campo de operando de almacenamiento no está implícita, sino que se indica explícitamente, se dice que el campo tiene una longitud variable. La longitud de los operandos de longitud variable puede variar en incrementos de un *byte* (o, en el caso de algunas instrucciones, en múltiplos de dos bytes u otros múltiplos). Cuando se almacena información, solamente se reemplaza el contenido de aquellas ubicaciones de *byte* que están incluidas en el campo especificado, aunque el ancho de la ruta física al almacenamiento puede ser mayor que la longitud del campo que se está almacenando.

Ciertas unidades de información deben estar en el almacenamiento dentro una frontera integral. Se dice que una frontera es integral para una unidad de información cuando su dirección de almacenamiento es un múltiplo de la longitud de la unidad en *bytes*. Se dan nombres especiales a los campos de 2, 4, 8 y 16 *bytes* en una frontera integral. Una media palabra es un grupo de dos *bytes* consecutivos en una frontera de dos *bytes* y es el elemento básico de construcción de instrucciones. Una palabra es un grupo de cuatro *bytes* consecutivos en una frontera de cuatro *bytes*. Una palabra doble es un grupo de ocho *bytes* consecutivos en una frontera de ocho *bytes*. Una palabra cuádruple es un grupo de 16 bytes consecutivos en una frontera de 16 bytes. Cuando las direcciones de almacenamiento designan medias palabras, palabras, palabras dobles, y palabras cuádruples, la representación

binaria de la dirección contiene, respectivamente, uno, dos, tres o cuatro bits cero en el extremo derecho. Las instrucciones deben encontrarse en fronteras integrales de dos *bytes*. Los operandos de almacenamiento de la mayoría de las instrucciones no tienen requisitos de alineación de frontera.

En dispositivos que implementan memorias caché separadas para operandos de instrucciones y operandos de datos, se puede sufrir un retraso significativo si el programa almacena en una línea de caché instrucciones que son posteriormente extraídas, con independencia de si el almacenamiento altera las instrucciones que son posteriormente extraídas.

En una realización, la invención puede ponerse en práctica mediante *software* (a veces denominado código interno con licencia, *firmware*, microcódigo, milicódigo, picocódigo y similares, cualquiera de los cuales sería consistente con uno o varios aspectos de la presente invención). Haciendo referencia a la Figura 12, el procesador 5001 del sistema anfitrión 5000 puede acceder a código de programa de *software* que realiza uno o varios aspectos de la presente invención, desde dispositivos 5011 de medios de almacenamiento a largo plazo, tales como una unidad de CD-ROM, una unidad de cinta o un disco duro. El código de programa de *software* puede estar realizado en cualquiera de una diversidad de medios conocidos para su uso con un sistema de procesamiento de datos, tales como un disquete, un disco duro o un CD-ROM. El código puede estar distribuido en dichos medios, o bien puede estar distribuido a usuarios desde la memoria 5002 de ordenador o desde el almacenamiento de un sistema informático, a través de una red 5010, a otros sistemas informáticos para su uso por usuarios de esos otros sistemas.

El código de programa de *software* incluye un sistema operativo que controla la función y la interacción de los diversos componentes del ordenador y uno o varios programas de aplicación. Normalmente se transfiere por páginas el código de programa desde el dispositivo 5011 de medios de almacenamiento hacia el almacenamiento 5002 de ordenador, que tiene una velocidad relativamente más alta, donde está disponible para su procesamiento por el procesador 5001. Las técnicas y métodos para incorporar el código de programa de *software* en la memoria, sobre medios físicos y/o para distribuir *software* de código a través de redes son bien conocidos y no se discutirán aquí con mayor detalle. Al código del programa, cuando se crea y se almacena en un medio tangible (que incluye, pero sin limitación, módulos electrónicos de memoria (RAM), memoria *flash*, discos compactos (CD), DVD, cinta magnética y similares), se le denomina a menudo "producto de programa informático". Típicamente, el medio para producto de programa informático es legible por un circuito de procesamiento, preferiblemente ubicado en un sistema informático, para ser ejecutado por el circuito de procesamiento.

La Figura 13 ilustra un sistema representativo de *hardware* de estación de trabajo o de servidor, en el cual se pueden poner en práctica uno o varios aspectos de la presente invención. El sistema 5020 de la Figura 13 comprende un sistema informático de base representativo 5021, por ejemplo un ordenador personal, una estación de trabajo o un servidor, incluidos dispositivos periféricos opcionales. El sistema informático de base 5021 incluye uno o varios procesadores 5026 y un bus empleado para conectar y permitir la comunicación entre los uno o varios procesadores 5026 y los demás componentes del sistema 5021, conforme a técnicas conocidas. El bus conecta el procesador 5026 a la memoria 5025 y al almacenamiento a largo plazo 5027, que puede incluir una unidad de disco duro (incluidos cualquiera de los soportes magnéticos, CD, DVD y memoria *flash*, por ejemplo) o una unidad de cinta, por ejemplo. El sistema 5021 también podría incluir un adaptador de interfaz de usuario, que conecta el microprocesador 5026, a través del bus, con uno o varios dispositivos de interfaz, tales como un teclado 5024, un ratón 5023, una impresora/escáner 5030 y/u otros dispositivos de interfaz, que pueden ser cualquier dispositivo de interfaz de usuario, por ejemplo una pantalla sensible al tacto, una tableta de entrada digitalizada, etc. El bus también conecta un dispositivo 5022 de visualización, tal como una pantalla LCD o un monitor, al microprocesador 5026 a través de un adaptador de visualización.

El sistema 5021 puede comunicarse con otros ordenadores o redes de ordenadores a través de un adaptador de red capaz de comunicar 5028 con una red 5029. Son ejemplos de adaptadores de red los canales de comunicación, el anillo "token ring", Ethernet o módems. Como alternativa, el sistema 5021 puede comunicarse utilizando una interfaz inalámbrica, por ejemplo una tarjeta CDPD (siglas inglesas de "datos de paquetes digitales celulares"). El sistema 5021 puede estar asociado con esos otros ordenadores en una red de área local (LAN, por sus siglas en inglés) o una red de área extensa (WAN), o bien el sistema 5021 puede ser un cliente en una disposición cliente/servidor con otro ordenador, etc. Todas estas configuraciones, así como los correspondientes *hardware* y *software* de comunicaciones, son conocidos en la técnica.

La Figura 14 ilustra una red 5040 de procesamiento de datos en la que se pueden poner en práctica uno o varios aspectos de la presente invención. La red 5040 de procesamiento de datos puede incluir una pluralidad de redes individuales, tales como una red inalámbrica y una red cableada, cada una de las cuales puede incluir una pluralidad de estaciones de trabajo individuales 5041, 5042, 5043, 5044. Además, como apreciarán los expertos en la técnica, se pueden incluir una o varias redes LAN, donde una red LAN puede comprender una pluralidad de estaciones de trabajo inteligentes acopladas a un procesador anfitrión.

Todavía con referencia a la Figura 14, las redes pueden incluir también ordenadores centrales o servidores, tales como un ordenador de pasarela (servidor de cliente 5046) o servidor de aplicaciones (servidor remoto 5048 que puede acceder a un repositorio de datos y al que también se puede acceder directamente desde una estación de

trabajo 5045). Un ordenador de pasarela 5046 sirve como punto de entrada en cada red individual. Se necesita una pasarela cuando se conectan entre sí dos protocolos de red. Preferiblemente, la pasarela 5046 puede estar acoplada a otra red (por ejemplo, la Internet 5047) por medio de un enlace de comunicaciones. La pasarela 5046 también puede estar acoplada directamente a una o varias estaciones de trabajo 5041, 5042, 5043, 5044 utilizando un enlace de comunicaciones. El ordenador de pasarela se puede implementar utilizando un servidor IBM eServer™ System z® disponible de International Business Machines Corporation.

Haciendo referencia simultáneamente a las Figuras 13 y 14, el procesador 5026 del sistema 5020 puede acceder, desde medios de almacenamiento a largo plazo 5027, tales como una unidad de CD-ROM o un disco duro, a código de programación de *software* que puede incorporar uno o varios aspectos de la presente invención. El código de programación de *software* puede estar realizado en cualquiera de una diversidad de medios conocidos para su uso con un sistema de procesamiento de datos, tales como un disquete, un disco duro o un CD-ROM. El código puede distribuirse en dichos medios, o bien puede distribuirse a los usuarios 5050, 5051 desde la memoria o almacenamiento de un sistema informático, a través de una red, a otros sistemas informáticos para su uso por usuarios de esos otros sistemas.

Como alternativa, el código de programación puede estar realizado en la memoria 5025, y accede a él el procesador 5026 utilizando el bus del procesador. Tal código de programación incluye un sistema operativo que controla la función y la interacción de los diversos componentes informáticos y uno o varios programas de aplicación 5032. Normalmente, el programa de código es transferido por páginas desde medios de almacenamiento 5027 a la memoria de alta velocidad 5025, donde está disponible para su procesamiento por el procesador 5026. Las técnicas y métodos para realizar el código de programación de *software* en la memoria, en medios físicos, y/o para distribuir código de software a través de redes, son bien conocidos y no se describirán con mayor detalle en la presente memoria. Al código de programa, cuando ha sido creado y almacenado en un soporte material (incluidos, pero sin limitación, módulos electrónicos de memoria (RAM), memoria *flash*, discos compactos (CD), DVD, cinta magnética y similares, se le denomina a menudo "producto de programa informático". Típicamente, el medio para producto de programa informático es legible por un circuito de procesamiento, preferiblemente ubicado en un sistema informático, para ser ejecutado por el circuito de procesamiento.

La memoria caché que está más fácilmente disponible para el procesador (normalmente más rápida y más pequeña que otras memorias caché del procesador) es la memoria caché más baja (L1 o nivel uno), y el almacenamiento principal (memoria principal) es la memoria caché de nivel más alto (L3 si hay 3 niveles). Frecuentemente se divide la memoria caché de nivel más bajo en una memoria caché para instrucciones (memoria caché I) que contiene instrucciones de máquina a ejecutar, y una memoria caché para datos (memoria caché D) que contiene operandos de datos.

Haciendo referencia a la Figura 15, se representa una realización ilustrativa de procesador para el procesador 5026. Típicamente se emplean uno o más niveles de memoria caché 5053 para configurar bloques de memoria intermedia, al objeto de mejorar el rendimiento del procesador. La memoria caché 5053 es una memoria intermedia de alta velocidad que guarda líneas de caché de datos de memoria que son susceptibles de ser utilizados. Las líneas de caché típicas tienen 64, 128 o 256 *bytes* de datos de memoria. Se emplean más a menudo memorias caché separadas para guardar instrucciones de almacenamiento que para guardar datos. Frecuentemente, diversos algoritmos "fisgones" (en inglés, "snoop"), bien conocidos en la técnica, proporcionan coherencia de memoria caché (sincronización de copias de líneas en la memoria y en las memorias caché). A menudo, al almacenamiento de la memoria principal 5025 de un sistema de procesador se le denomina memoria caché. En un sistema de procesador que tiene 4 niveles de caché 5053, a la memoria principal 5025 se la denomina a veces memoria caché de nivel 5 (L5), ya que es típicamente más rápida y sólo guarda una parte de la memoria no volátil (DASD, cinta, etc.) que está disponible para un sistema informático. El almacenamiento principal 5025 "guarda" páginas de datos transferidas como páginas hacia dentro y hacia fuera de la memoria principal 5025 por el sistema operativo.

Un contador de programa (contador de instrucciones) 5061 realiza un seguimiento de la dirección de la instrucción actual que ha de ejecutarse. Un contador de programa en un procesador z/Architecture® tiene 64 bits y se puede truncar a 31 o 24 bits para dar soporte a límites de direccionamiento anteriores. Típicamente, un contador de programa está realizado en una PSW (palabra de estado de programa) de un equipo, de manera que persiste durante la conmutación de contexto. Por tanto, un programa en curso, que tiene un valor de contador de programa, puede ser interrumpido, por ejemplo, por el sistema operativo (conmutación de contexto desde el entorno de programa hacia el entorno del sistema operativo). La PSW del programa mantiene el valor del contador del programa, mientras el programa no está activo, y se utiliza el contador de programa (de la PSW) del sistema operativo mientras se está ejecutando el sistema operativo. Típicamente, el contador de programa se incrementa en una cantidad igual al número de bytes de la instrucción actual en ese momento. Las instrucciones RISC (cálculo con conjunto de instrucciones reducidas, por sus siglas en inglés) tienen normalmente longitud fija, mientras que las instrucciones CISC (cálculo con conjunto de instrucciones complejas) tienen normalmente longitud variable. Las instrucciones de la z/Architecture® de IBM son instrucciones CISC que tienen una longitud de 2, 4 o 6 bytes. El contador 5061 de programa resulta modificado tanto por una operación de conmutación de contexto como por una operación de toma de bifurcación de una instrucción de bifurcación, por ejemplo. En una operación de conmutación de contexto, el valor de contador de programa en ese momento se guarda en la palabra de estado del programa junto con otra información de estado acerca del programa en ejecución (por ejemplo, códigos de situación), y se

carga un nuevo valor de contador de programa que apunta a una instrucción de un nuevo módulo de programa que debe ejecutarse. Se realiza una operación de toma de bifurcación para permitir que el programa tome decisiones o realice bucles dentro del programa mediante la carga del resultado de la instrucción de bifurcación en el contador de programa 5061.

Típicamente, se emplea una unidad 5055 de extracción de instrucciones para extraer instrucciones por cuenta del procesador 5026. La unidad de extracción extrae, ya sea "las siguientes instrucciones secuenciales", instrucciones de destino de instrucciones de toma de bifurcación, o primeras instrucciones de un programa después de una conmutación de contexto. Las unidades modernas de extracción de instrucciones emplean a menudo técnicas de pre-extracción para pre-extraer especulativamente instrucciones basándose en la probabilidad de que se puedan utilizar las instrucciones pre-extraídas. Por ejemplo, una unidad de extracción puede extraer 16 bytes de instrucción que incluya la siguiente instrucción secuencial y *bytes* adicionales de más instrucciones secuenciales.

A continuación, el procesador 5026 ejecuta las instrucciones extraídas. En una realización, la o las instrucciones extraídas son transferidas a una unidad 5056 de despacho de la unidad de extracción. La unidad de despacho descodifica la o las instrucciones y reenvía la información acerca de la instrucción o instrucciones decodificadas a las unidades apropiadas 5057, 5058, 5060. Una unidad 5057 de ejecución recibirá típicamente información acerca de instrucciones aritméticas descodificados procedente de la unidad 5055 de extracción de instrucciones, y llevará a cabo operaciones aritméticas sobre operandos conforme al código operativo de la instrucción. Preferiblemente, los operandos se proporcionan a la unidad 5057 de ejecución, ya sea desde la memoria 5025, desde registros arquitecturados 5059, o desde un campo inmediato de la instrucción que se está ejecutando. Los resultados de la ejecución, cuando se almacenan, lo hacen sea en la memoria 5025, en los registros 5059 o en otro *hardware* de la máquina (por ejemplo, registros de control, registros de PSW y similares).

Un procesador 5026 tiene típicamente una o varias unidades 5057, 5058, 5060 para ejecutar la función de la instrucción. Haciendo referencia a la Figura 16A, una unidad de ejecución 5057 puede comunicarse con registros generales arquitecturados 5059, una unidad de descodificación/despacho 5056, una unidad de almacenamiento de carga 5060, y otras unidades 5065 de procesador por medio de la lógica de interfaz 5071. Una unidad de ejecución 5057 puede utilizar varios circuitos de registro 5067, 5068, 5069 para conservar información con la cual va a operar la unidad aritmético-lógica (ALU, por sus siglas en inglés) 5066. La ALU realiza operaciones aritméticas tales como sumar, restar, multiplicar y dividir, así como funciones lógicas tales como AND, OR y OR-exclusiva (XOR), rotar y desplazar. Preferiblemente, la ALU soporta operaciones especializadas que son dependientes del diseño. Otros circuitos pueden proporcionar otras funcionalidades arquitecturadas 5072, entre ellas, por ejemplo, códigos de situación y lógica de apoyo a la recuperación. Típicamente, el resultado de una operación ALU se conserva en un circuito de registro de salida 5070, que puede remitir el resultado a una diversidad de otras funciones de procesamiento. Existen muchas disposiciones de unidades de procesador, y la presente descripción sólo pretende proporcionar una comprensión representativa de una realización.

Por ejemplo, una instrucción ADD (sumar) sería ejecutada en una unidad de ejecución 5057 que tuviese operatividad aritmética y lógica, mientras que una instrucción de coma flotante, por ejemplo, sería ejecutada en una ejecución de coma flotante, que tiene capacidad especializada de coma flotante. Preferiblemente, una unidad de ejecución opera sobre operandos identificados por una instrucción, realizando una función definida por código operativo sobre los operandos. Por ejemplo, una instrucción ADD puede ser ejecutada por una unidad de ejecución 5057 sobre operandos que se encuentren en dos registros 5059 identificados por campos de registro de la instrucción.

La unidad de ejecución 5057 realiza la suma aritmética de dos operandos y almacena el resultado en un tercer operando, donde el tercer operando puede ser un tercer registro o bien uno de los dos registros de origen. La unidad de ejecución utiliza preferentemente una unidad aritmético-lógica (ALU) 5066 que es capaz de realizar diversas funciones lógicas tales como desplazar, rotar, AND, OR y XOR, así como diversas funciones algebraicas, entre ellas cualquiera de sumar, restar, multiplicar y dividir. Algunas ALU 5066 están diseñadas para operaciones escalares y algunas para coma flotante. Los datos pueden ser de tipo "por el extremo grande" (en inglés, "big endian") (en donde el *byte* menos significativo está en la dirección de *byte* más alta) o de tipo "por el extremo pequeño" (en inglés, "little endian") (donde el *byte* menos significativo está en la dirección de *byte* más baja) dependiendo de la arquitectura. La z/Architecture® de IBM es "big endian". Los campos con firma pueden ser signo y magnitud, complemento a 1 o complemento a 2, dependiendo de la arquitectura. Resulta ventajoso un número de complemento a 2 porque la ALU no necesita diseñar una capacidad de resta, ya que tanto un valor negativo como un valor positivo de complemento a 2 sólo requiere una adición dentro de la ALU. Los números se describen comúnmente en forma abreviada, en donde campo de 12 bits define una dirección de un bloque de 4.096 *bytes* y se describe comúnmente como un bloque de 4 Kbytes (*kilobytes*), por ejemplo.

Haciendo referencia a la Figura 16B, típicamente se envía la información de instrucción de bifurcación para ejecutar una instrucción de bifurcación a una unidad de bifurcación 5058 que a menudo emplea un algoritmo de predicción de bifurcaciones, por ejemplo una tabla 5082 de historial de bifurcaciones, para predecir el resultado de la bifurcación antes de que se hayan completado otras operaciones condicionales. Se extraerá el objetivo de la instrucción de bifurcación actual en ese momento y se ejecutará especulativamente antes de que se hayan completado las operaciones condicionales. Cuando se completan las operaciones condicionales, o bien se completan las

instrucciones de bifurcación ejecutadas especulativamente, o bien se descartan sobre la base de las condiciones de la operación condicional y el resultado especulado. Una instrucción de bifurcación típica puede probar códigos de situación y bifurcar a una dirección de destino si los códigos de situación cumplen el requisito de bifurcación de la instrucción de bifurcación, se puede calcular una dirección de destino sobre la base de diversos números, entre ellos los que se encuentran en los campos de registro o un campo inmediato de la instrucción, por ejemplo. La unidad de bifurcación 5058 puede emplear una ALU 5074 que tenga una pluralidad de circuitos de registro de entrada 5075, 5076, 5077 y un circuito de registro de salida 5080. La unidad de bifurcación 5058 puede comunicarse con registros generales 5059, la unidad de despacho de descodificación 5056 u otros circuitos 5073, por ejemplo.

La ejecución de un grupo de instrucciones puede verse interrumpida por diversas razones, entre ellas una conmutación de contexto iniciada por un sistema operativo, una excepción de programa o un error que provoca una conmutación de contexto, una señal de interrupción E/S que provoca una conmutación de contexto o la actividad de múltiples hilos de una pluralidad de programas (en un entorno de múltiples hilos), por ejemplo. Preferiblemente, una acción de conmutación de contexto guarda información de estado acerca de un programa que se está ejecutando en ese momento y luego carga información de estado acerca de otro programa que se invoca. La información de estado se puede guardar en los registros de hardware o en la memoria, por ejemplo. Preferiblemente, la información de estado comprende un valor de contador de programa que apunta a una siguiente instrucción a ejecutar, códigos de situación, información de traducción de memoria y contenido de registros arquitecturados. Una actividad de conmutación de contexto la pueden ejercer circuitos de hardware, programas de aplicación, programas de sistema operativo o código de *firmware* (microcódigo, picocódigo o código interno bajo licencia (LIC)), solos o en combinación.

Un procesador accede a operandos en función de métodos definidos por instrucción. La instrucción puede proporcionar un operando inmediato utilizando el valor de una parte de la instrucción, puede proporcionar uno o varios campos de registro que apunten explícitamente, ya sea a registros de propósito general o a registros de propósito especial (registros de coma flotante, por ejemplo). La instrucción puede utilizar como operandos registros implícitos identificados por un campo de código operativo. La instrucción puede utilizar como operandos posiciones de memoria. Una ubicación de memoria de un operando la puede proporcionar un registro, un campo inmediato o una combinación de registros y campo inmediato, como lo ilustra la funcionalidad de desplazamiento largo de la z/Architecture®, en donde la instrucción define un registro de base, un registro de índice y un campo inmediato (campo de desplazamiento) que se suman para proporcionar la dirección del operando en la memoria, por ejemplo. Localización en la presente memoria implica típicamente una ubicación en la memoria principal (almacenamiento principal), salvo que se indique otra cosa.

Haciendo referencia a la Figura 16C, un procesador accede al almacenamiento utilizando una unidad de carga/almacenamiento 5060. La unidad de carga/almacenamiento 5060 puede llevar a cabo una operación de carga obteniendo la dirección del operando de destino en la memoria 5053 y cargando el operando en un registro 5059 u otra ubicación 5053 de memoria, o bien puede llevar a cabo una operación de almacenamiento obteniendo la dirección del operando de destino en la memoria 5053 y almacenando datos obtenidos de un registro 5059 u otra ubicación 5053 de memoria en la ubicación del operando de destino en la memoria 5053. La unidad de carga/almacenamiento 5060 puede ser especulativa, y puede acceder a la memoria en una secuencia que no sigue el orden en cuanto a la secuencia de instrucciones, pero la unidad de carga/almacenamiento 5060 debe mantener frente a los programas la apariencia de que las instrucciones se ejecutan en orden. Una unidad de carga/almacenamiento 5060 puede comunicar con registros generales 5059, la unidad de descodificación/despacho 5056, la interfaz 5053 de memoria caché/memoria, u otros elementos 5083, y puede comprender diversos circuitos de registro, unidades ALU 5085 y lógica de control 5090 para calcular direcciones de almacenamiento y para proporcionar secuenciación de tubería con el fin de mantener las operaciones en orden. Algunas operaciones pueden no seguir el orden, pero la unidad de carga/almacenamiento proporciona una funcionalidad para hacer que el resultado de las operaciones aparente para el programa que se han realizado en orden, como es bien conocido en la técnica.

Preferiblemente, a las direcciones que "ve" un programa de aplicación se las denomina con frecuencia direcciones virtuales. A las direcciones virtuales se las denomina a veces "direcciones lógicas" y "direcciones efectivas". Estas direcciones virtuales son virtuales porque son redirigidas a la ubicación física de memoria por alguna de diversas tecnologías de traducción dinámica de direcciones (DAT), entre ellas, pero sin limitación, simplemente añadiendo como prefijo a una dirección virtual un valor de desplazamiento, traduciendo la dirección virtual a través de una o varias tablas de traducción, en donde las tablas de traducción comprenden preferiblemente al menos una tabla de segmentos y una tabla de páginas, solas o en combinación, en donde, preferiblemente, la tabla de segmentos tiene un señalador que apunta a la tabla de páginas. En la z/Architecture®, se proporciona una jerarquía de traducción, que incluye una primera tabla de regiones, una segunda tabla de regiones, una tercera tabla de regiones, una tabla de segmentos y una tabla de página opcional. El rendimiento de la traducción de direcciones se mejora frecuentemente mediante la utilización de una memoria intermedia de traducción lateral (TLB) que comprende el cartografiado de entradas desde una dirección virtual a una ubicación de memoria física asociada. Las entradas se crean cuando la DAT traduce una dirección virtual utilizando las tablas de traducción. El uso posterior de la dirección virtual puede utilizar después la entrada de la rápida TLB en lugar de los lentos accesos a las tablas de traducción secuencial. El contenido de la TLB puede ser administrado por una diversidad de algoritmos de sustitución, incluido el LRU ("menos recientemente utilizado", en sus siglas inglesas).

En caso de que el procesador sea un procesador de un sistema multiprocesador, cada procesador tiene la responsabilidad de mantener enclavados por coherencia los recursos compartidos, tales como E/S, memorias caché, memorias TLB y memoria. Por lo general, se utilizarán tecnologías "fisgonas" para mantener la coherencia de la memoria caché. En un entorno fisgón, se puede marcar cada línea de caché para que esté, o bien en un estado compartido, en un estado exclusivo, en un estado modificado o en un estado inválido, y similares, con el fin de facilitar el intercambio.

Las unidades de E/S 5054 (Figura 15) proporcionan al procesador medios para conectarse a dispositivos periféricos, entre ellos cinta, disco, impresoras, pantallas de visualización y redes, por ejemplo. Con frecuencia, controladores de *software* presentan las unidades de E/S al programa informático. En ordenadores centrales, tales como el System z® de IBM®, los adaptadores de canal y los adaptadores de sistemas abiertos son unidades de E/S del ordenador central que se ocupan de las comunicaciones entre el sistema operativo y los dispositivos periféricos.

Además, otros tipos de entornos informáticos se pueden beneficiar de uno o varios aspectos de la presente invención. A modo de ejemplo, tal como se ha mencionado en la presente memoria, un entorno puede incluir un emulador (por ejemplo, *software* u otros mecanismos de emulación), en donde se emula una arquitectura particular (incluidas, por ejemplo, la ejecución de instrucciones, funciones arquitecturadas, tales como traducción de direcciones, y registros arquitecturados) o un subconjunto de los mismos (por ejemplo, en un sistema informático nativo que tiene un procesador y memoria). En un entorno semejante, una o más funciones de emulación del emulador pueden implementar uno o varios aspectos de la presente invención, aunque un ordenador que ejecute el emulador pueda tener una arquitectura distinta de las capacidades que se emulan. A modo de ejemplo, en modo de emulación se descodifica la instrucción u operación específica que se está emulando, y se construye una función de emulación adecuada para implementar la instrucción u operación individuales.

En un entorno de emulación, un ordenador anfitrión incluye, por ejemplo, una memoria para almacenar instrucciones y datos; una unidad de extracción de instrucciones para extraer instrucciones de la memoria y, opcionalmente, proporcionar almacenamiento intermedio local para la instrucción extraída; una unidad de descodificación de instrucciones para recibir las instrucciones extraídas y para determinar el tipo de instrucciones que han sido extraídas; y una unidad de ejecución de instrucciones para ejecutar las instrucciones. La ejecución puede incluir cargar datos en un registro desde la memoria; almacenar datos de vuelta a la memoria desde un registro; o realizar algún tipo de operación aritmética o lógica, según determine la unidad de descodificación. En un ejemplo, cada unidad se implementa en *software*. Por ejemplo, las operaciones que las unidades llevan a cabo se implementan como una o varias subrutinas dentro del *software* del emulador.

Más particularmente, en un ordenador central, las instrucciones de máquina arquitecturadas son utilizadas por los programadores, normalmente hoy en día programadores "C", a menudo por medio de una aplicación compiladora. Estas instrucciones almacenadas en el medio de almacenamiento pueden ser ejecutadas de forma nativa en un z/Architecture® IBM® Server o, como alternativa, en máquinas que ejecuten otras arquitecturas. Pueden ser emuladas en los servidores de ordenador central presentes y futuros de IBM® y en otras máquinas de IBM® (por ejemplo, servidores Power Systems y System x® Servers). Se pueden ejecutar en máquinas que ejecutan Linux sobre una amplia variedad de máquinas que utilizan hardware fabricado por IBM®, Intel®, AMD™ y otros. Además de la ejecución en ese *hardware* bajo z/Architecture®, se puede utilizar Linux, así como máquinas que utilizan emulación por Hercules, UMX o FSI (Fundamental Software, Inc.), en donde generalmente la ejecución se realiza en modo de emulación. En modo de emulación, el *software* de emulación es ejecutado por un procesador nativo con el fin de emular la arquitectura de un procesador emulado.

Típicamente, el procesador nativo ejecuta el *software* de emulación que comprende, o bien *firmware* o bien un sistema operativo nativo, para realizar la emulación del procesador emulado. El *software* de emulación es responsable de extraer y ejecutar instrucciones de la arquitectura del procesador emulado. *El* software de emulación mantiene un contador de programa emulado para realizar un seguimiento de los límites de instrucción. El *software* de emulación puede extraer cada vez una o varias instrucciones de la máquina emulada y convertir la una o varias instrucciones de máquina emuladas en un grupo correspondiente de instrucciones de máquina nativa para que las ejecute el procesador nativo. Estas instrucciones convertidas se pueden almacenar en memoria caché, de manera que se pueda conseguir una conversión más rápida. No obstante, el *software* de emulación debe mantener las normas de arquitectura de la arquitectura del procesador emulado con el fin de asegurar que los sistemas operativos y las aplicaciones escritas para el procesador emulado funcionen correctamente. Además, el software de emulación debe proporcionar recursos identificados por la arquitectura del procesador emulado que incluyen, pero sin limitación, registros de control, registros de uso general, registros de coma flotante, función de traducción dinámica de direcciones, incluidas, por ejemplo, tablas de segmentos y tablas de páginas, mecanismos de interrupción, mecanismos de conmutación de contexto, relojes de hora del día (TOD, por sus siglas en inglés) e interfaces arquitecturadas hacia subsistemas de E/S de manera que un sistema operativo o un programa de aplicación, diseñados para ejecutarse en el procesador emulado, se puedan ejecutar en el procesador nativo que posee el *software* de emulación.

Se descodifica una instrucción específica que se está emulando, y se llama a una subrutina para realizar la función de la instrucción individual. Una función de *software* de emulación que emula una función de un procesador emulado se implementa, por ejemplo, en una subrutina o controlador en "C", o algún otro método para proporcionar un

controlador para el *hardware* específico, que entrará en la pericia de los técnicos después de comprender la descripción de la realización preferida. Diversas patentes de *software* y *hardware* de emulación, que incluyen, pero sin limitación, el documento de patente de EE.UU. n.º 5,551,013, titulado "Multiprocessor for Hardware Emulation", de Beausoleil *et al.*; y el documento de patente de EE.UU. n.º 6,009,261, titulado "Preprocessing of Stored Target Routines for Emulating Incompatible Instructions on a Target Processor", de Scalzi *et al.*; y el documento de patente de EE.UU. n.º 5,574,873, titulado "Decoding Guest Instruction to Directly Access Emulation Routines that Emulate the Guest Instructions", de Davidian *et al.*; y el documento de patente de EE.UU. n.º 6,308,255, titulado "Symmetrical Multiprocessing Bus and Chipset Used for Coprocessor Support Allowing Non-Native Code to Run in a System", de Gorishek *et al.*; y el documento de patente de EE.UU. n.º 6,463,582, titulado "Dynamic Optimizing Object Code Translator for Architecture Emulation and Dynamic Optimizing Object Code Translation Method", de Lethin *et al.*; y el documento de patente de EE.UU. n.º 5,790,825, titulado "Method for Emulating Guest Instructions on a Host Computer Through Dynamic Recompilation of Host Instructions", por Eric Traut, y muchos otros, ilustran una variedad de formas conocidas para conseguir la emulación de un formato de instrucción arquitecturado para una máquina distinta, destinadas a una máquina de destino disponible para los expertos en la técnica.

En Figura 17 se presenta un ejemplo de un sistema informático anfitrión emulado 5092 que emula un sistema informático anfitrión 5000' de una arquitectura anfitriona. En el sistema informático anfitrión emulado 5092, el procesador (CPU) anfitrión 5091 es un procesador anfitrión emulado (o procesador anfitrión virtual) y comprende un procesador de emulación 5093 que tiene una arquitectura de conjunto de instrucciones nativas diferente de la del procesador 5091 del ordenador anfitrión 5000'. El sistema informático anfitrión emulado 5092 tiene memoria 5094 accesible para el procesador de emulación 5093. En el ejemplo de realización, la memoria 5094 está divida en una parte de memoria de ordenador anfitrión 5096 y una parte de rutinas de emulación 5097. La memoria del ordenador anfitrión 5096 está disponible para programas del ordenador anfitrión emulado 5092 conformes a la arquitectura del ordenador anfitrión. El procesador de emulación 5093 ejecuta instrucciones nativas de un conjunto de instrucciones arquitecturadas de una arquitectura distinta de la del procesador emulado 5091, obteniéndose las instrucciones nativas de la memoria de rutinas de emulación 5097, y puede acceder a una instrucción de anfitrión para ejecutar un programa en la memoria del ordenador anfitrión 5096 mediante el empleo de una o varias instrucciones obtenidas, en una secuencia y rutina de acceso/descodificación que puede descodificar la o las instrucciones de anfitrión a las que se ha tenido acceso, con el fin de determinar una rutina de ejecución de instrucciones nativas para emular la función de la instrucción de anfitrión a la que se ha tenido acceso. Otras funcionalidades que están definidas para la arquitectura del ordenador anfitrión 5000' pueden ser emuladas por rutinas de funcionalidades arquitecturadas, entre ellas funcionalidades tales como registros de uso general, registros de control, traducción dinámica de direcciones y apoyo a subsistemas de E/S y de memoria caché del procesador, por ejemplo. Las rutinas de emulación también pueden aprovechar funciones disponibles en el procesador de emulación 5093 (como registros generales y traducción dinámica de direcciones virtuales) para mejorar el rendimiento de las rutinas de emulación. También se pueden proporcionar motores de *hardware* especial y motores de descarga para ayudar a que el procesador 5093 emule la función del ordenador anfitrión 5000'.

La terminología usada en la presente memoria tiene por objetivo solamente describir realizaciones particulares, y no se pretende que sea limitativa de la invención. En la presente memoria, las formas singulares "un", "una", "el" y "la" pretenden incluir también las formas plurales, a menos que el contexto indique claramente lo contrario. Se entenderá además que los términos "comprende" y/o "que comprende", cuando se utilizan en esta memoria descriptiva, especifican la presencia de las características, números enteros, pasos, operaciones, elementos y/o componentes nombrados, pero no excluye la presencia o adición de una o más características, números enteros, pasos, operaciones, elementos, componentes y/o grupos de los mismos.

En su caso, las correspondientes estructuras, materiales, acciones y equivalentes de todos los medios o elementos de paso más función de las reivindicaciones que siguen, pretenden incluir cualquier estructura, material o acción para realizar la función en combinación con otros elementos reivindicados tal como se reivindican específicamente. Se ha presentado la descripción de uno o varios aspectos de la presente invención con fines ilustrativos y de descripción, pero no se pretende que sea exhaustiva ni limitada al invento en la forma descrita. Para los expertos ordinarios en la técnica resultarán evidentes muchas modificaciones y variaciones sin salir del alcance de la invención. Se ha elegido y descrito la realización con el fin de explicar de la mejor manera los principios de la invención y la aplicación práctica, y para permitir a otros con pericia ordinaria en la técnica comprender la invención acerca de diversas realizaciones con diversas modificaciones, según sea adecuado para el uso particular contemplado.

REIVINDICACIONES

1. Un sistema informático (200) para facilitar el procesamiento en un entorno informático, comprendiendo dicho sistema informático:

una memoria; y

un procesador en comunicación con la memoria, en donde el sistema informático está configurado para realizar un método, comprendiendo dicho método:

que un programa obtenga una indicación de una funcionalidad de margen de advertencia instalada (700) dentro del entorno informático, donde la funcionalidad de margen de advertencia proporciona al programa un período de gracia de margen de advertencia para realizar una función;

que el programa inicie, basándose en la obtención de la indicación de que la funcionalidad de margen de advertencia está instalada, la inscripción del programa en el registro de la funcionalidad de margen de advertencia (702),

comprendiendo la inscripción en el registro una petición no solicitada de inscripción en el registro que indica que el programa entiende un protocolo de la funcionalidad de margen de advertencia y pretende participar en la funcionalidad de margen de advertencia;

que el programa reciba una notificación de margen de advertencia que indica que ha comenzado el período de gracia de margen de advertencia, siendo la inscripción del programa en el registro de la funcionalidad de margen de advertencia un requisito previo para recibir la notificación de margen de advertencia, en donde si el programa no está registrado, entonces no se ofrece período de gracia; y

que el programa, basándose en la notificación de margen de advertencia, al menos inicie la función dentro del período de gracia de margen de advertencia.

2. El sistema informático según la reivindicación 1, en donde el método comprende además que el programa se inscriba en el registro de la funcionalidad de margen de advertencia.

3. El sistema informático según la reivindicación 2, en donde, basándose en el registro, se habilita al programa para la funcionalidad de margen de advertencia.

4. El sistema informático según la reivindicación 1, en donde la función comprende uno de:

completar una unidad despachable en ejecución en un procesador en el cual se ejecuta el programa; o

hacer re-despachable en otro procesador del entorno informático a la unidad despachable.

5. El sistema informático según la reivindicación 1, en donde el programa es un programa invitado que tiene acceso a recursos compartidos del entorno informático durante una porción de tiempo proporcionada a una unidad central de procesamiento invitada en la que se ejecuta el programa invitado, y en donde el período de gracia de margen de advertencia es distinguible de la porción de tiempo.

6. El sistema informático según la reivindicación 5, en donde el período de gracia de margen de advertencia termina prematuramente la porción de tiempo.

7. El sistema informático según la reivindicación 5, en donde el período de gracia de margen de advertencia proporciona un plazo adicional a la porción de tiempo para realizar la función.

8. El sistema informático según la reivindicación 1, en donde el período de gracia es adicional a una porción de tiempo proporcionada al programa, y en donde si la porción de tiempo ha expirado por completo, el período de gracia es con cargo a una siguiente porción de tiempo para el programa y, si la porción de tiempo no ha expirado, el período de gracia se toma del tiempo normal restante de la porción de tiempo.

9. Un método para facilitar el procesamiento en un entorno informático (200), comprendiendo dicho método:

que un programa obtenga una indicación de una funcionalidad de margen de advertencia instalada (700) dentro del entorno informático, donde la funcionalidad de margen de advertencia proporciona al programa un período de gracia de margen de advertencia para realizar una función;

que el programa inicie, basándose en la obtención de la indicación de que la funcionalidad de margen de advertencia está instalada, la inscripción del programa en el registro de la funcionalidad de margen de advertencia (702),

comprendiendo la inscripción en el registro una petición no solicitada de inscripción en el registro que indica que el programa entiende un protocolo de la funcionalidad de margen de advertencia y pretende participar en la

funcionalidad de margen de advertencia;

que el programa reciba una notificación de margen de advertencia que indica ha comenzado el período de gracia de margen de advertencia, siendo la inscripción del programa en el registro de la funcionalidad de margen de advertencia un requisito previo para recibir la notificación de margen de advertencia, en donde si el programa no está registrado, entonces no se ofrece período de gracia; y

que el programa, basándose en la notificación de margen de advertencia, al menos inicie la función dentro del período de gracia de margen de advertencia.

10. El método según la reivindicación 9, en donde la función comprende uno de:

completar una unidad despachable en ejecución en un procesador en el cual se ejecuta el programa; o

hacer re-despachable en otro procesador del entorno informático a la unidad despachable.

11. El método según la reivindicación 9, en donde el programa es un programa invitado que tiene acceso a recursos compartidos del entorno informático durante una porción de tiempo proporcionada a una unidad central de procesamiento invitada en la que se ejecuta el programa invitado, y en donde el período de gracia de margen de advertencia es distinguible de la porción de tiempo.

12. El método según la reivindicación 9, en donde la inscripción en el registro se realiza a través de una instrucción Diagnose, en donde la instrucción Diagnose comprende un parámetro de registro para indicar que el segundo programa entiende una interrupción por margen de advertencia de la funcionalidad de margen de advertencia.

13. El método según la reivindicación 9, en donde el período de gracia es distinguible de una porción de tiempo proporcionada a un procesador en el cual se ejecuta el programa, teniendo acceso el programa a recursos compartidos del entorno informático durante la porción de tiempo, y en donde el período de gracia proporciona un plazo adicional a la porción de tiempo, en donde cualquier parte del plazo utilizado por el programa para realizar la función es con cargo a una siguiente porción de tiempo para el segundo programa.

14. El método según la reivindicación 9, en donde el método comprende además:

que el programa obtenga, la vez siguiente que se ejecute el programa, basada en la determinación de que el programa ha salido voluntariamente antes de la expiración del período de gracia, retroalimentación positiva que indica la salida voluntaria antes de la expiración del período de gracia; y

que el programa obtenga, la vez siguiente que se ejecute el programa, basada en la determinación de que el programa no ha llegado a tiempo para salir voluntariamente antes de la expiración del período de gracia, una indicación de retroalimentación de excepción que indica tardanza en salir voluntariamente.

15. Un programa informático que comprende código de programa informático para, cuando está cargado en un sistema informático y se ejecuta en el mismo, hacer que dicho sistema informático realice todos los pasos del método según cualquiera de las reivindicaciones 9 a 14.

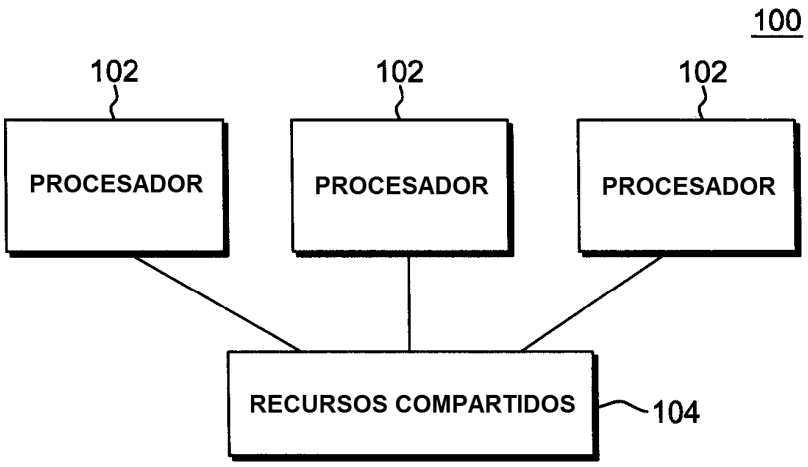
100
102
102
102
PROCESADOR
PROCESADOR
PROCESADOR
RECURSOS COMPARTIDOS
104

FIG. 1

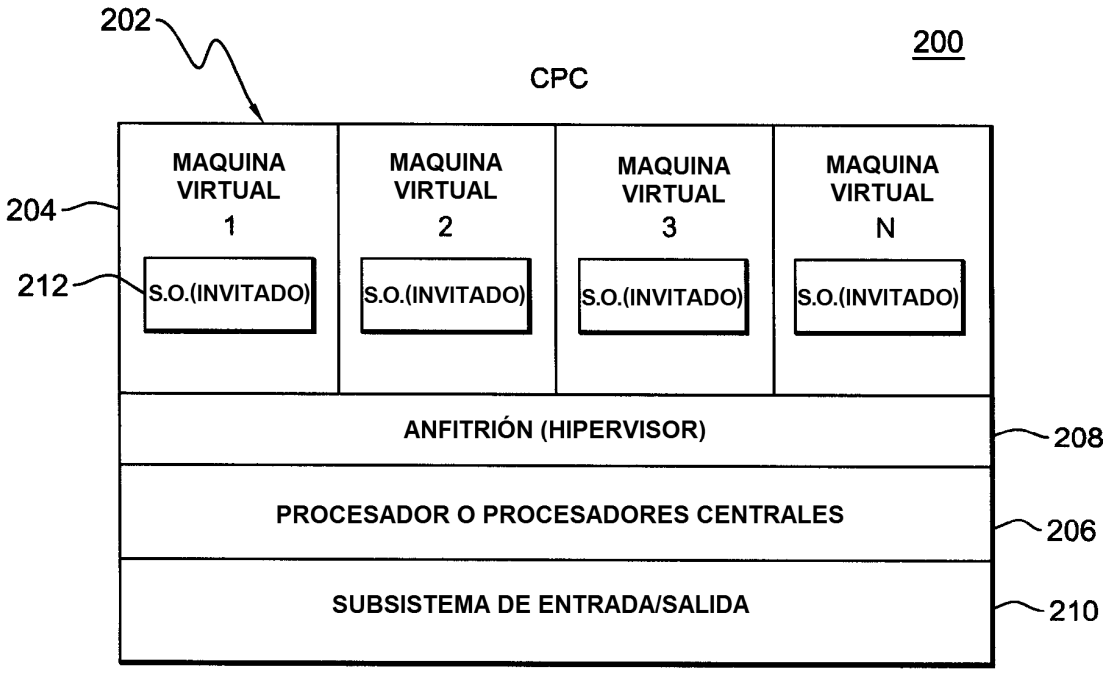
202
200
CPC
204
212
MAQUINA VIRTUAL 1
MAQUINA VIRTUAL 2
MAQUINA VIRTUAL 3
MAQUINA VIRTUAL N
S.O.(INVITADO)
S.O.(INVITADO)
S.O.(INVITADO)
S.O.(INVITADO)
ANFITRIÓN (HIPERVISOR)
208
PROCESADOR O PROCESADORES CENTRALES
206
SUBSISTEMA DE ENTRADA/SALIDA
210

FIG. 2

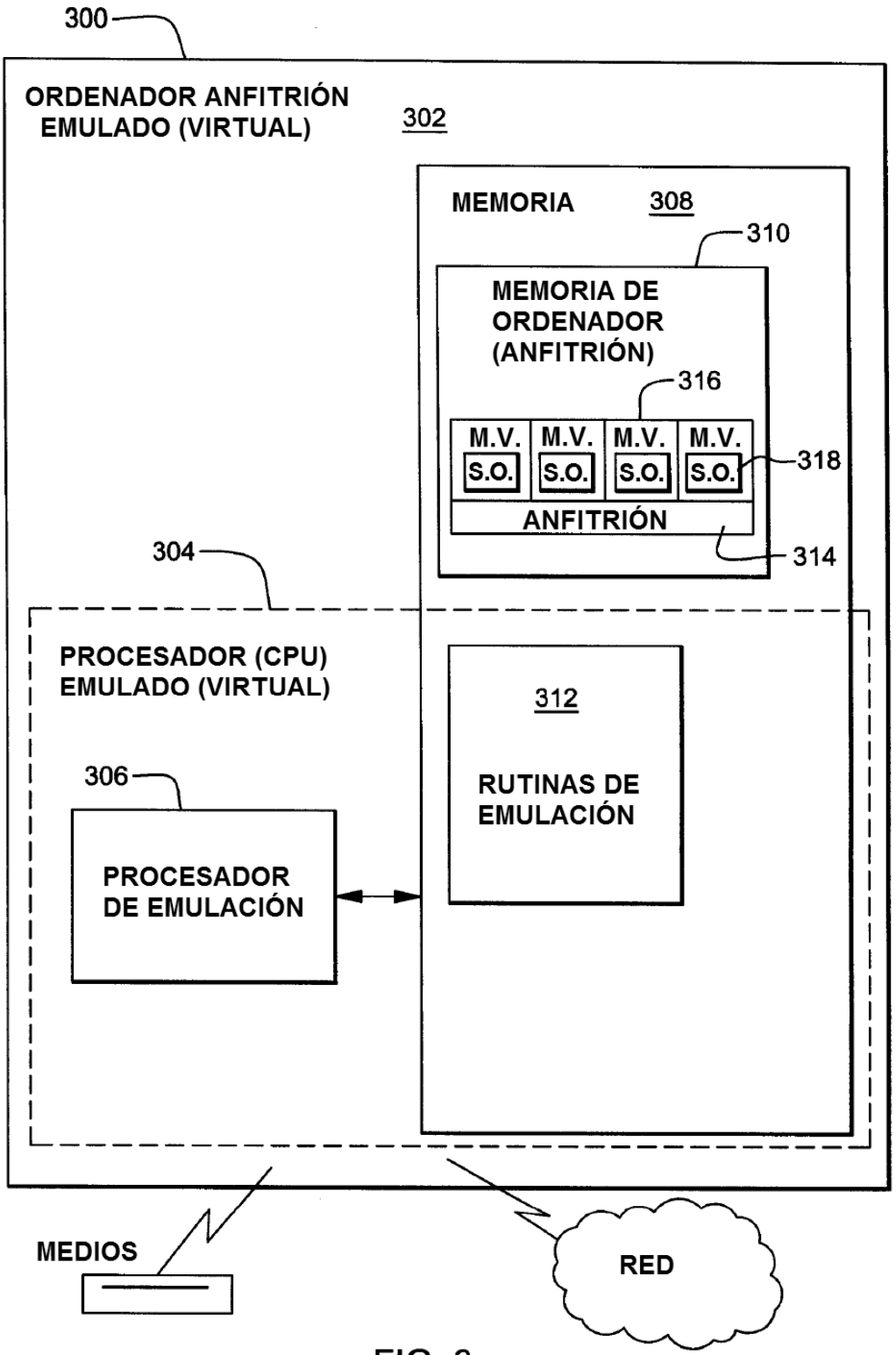
300
ORDENADOR ANFITRIÓN EMULADO (VIRTUAL)
302
MEMORIA
308
310
MEMORIA DE ORDENADOR (ANFITRIÓN)
316
M.V.
S.O.
M.V.
S.O.
M.V.
S.O.
M.V.
S.O.
318
ANFITRIÓN
314
304
PROCESADOR (CPU) EMULADO (VIRTUAL)
312
RUTINAS DE EMULACIÓN
306
PROCESADOR DE EMULACIÓN
MEDIOS
RED

FIG. 3

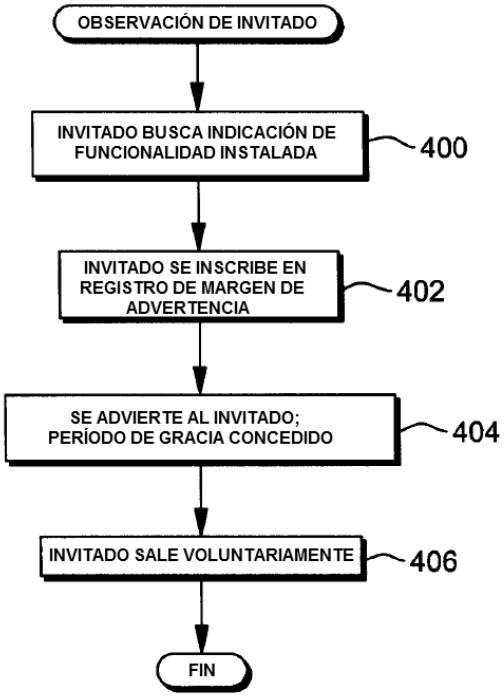
OBSERVACIÓN DE INVITADO
INVITADO BUSCA INDICACIÓN DE FUNCIONALIDAD INSTALADA
400
INVITADO SE INSCRIBE EN REGISTRO DE MARGEN DE ADVERTENCIA
402
SE ADVIERTE AL INVITADO; PERÍODO DE GRACIA CONCEDIDO
404
INVITADO SALE VOLUNTARIAMENTE
406
FIN

FIG. 4

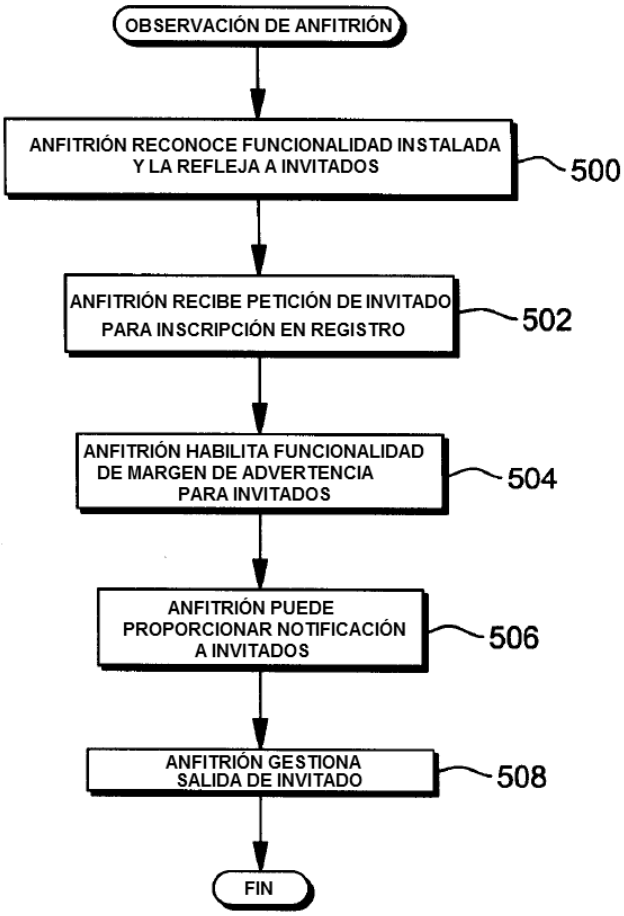
OBSERVACIÓN DE ANFITRIÓN
ANFITRIÓN RECONOCE FUNCIONALIDAD INSTALADA Y LA REFLEJA A INVITADOS
500
ANFITRIÓN RECIBE PETICIÓN DE INVITADO PARA INSCRIPCIÓN EN REGISTRO
502
ANFITRIÓN HABILITA FUNCIONALIDAD DE MARGEN DE ADVERTENCIA PARA INVITADOS
504
ANFITRIÓN PUEDE PROPORCIONAR NOTIFICACIÓN A INVITADOS
506
ANFITRIÓN GESTIONA SALIDA DE INVITADO
508
FIN

FIG. 5

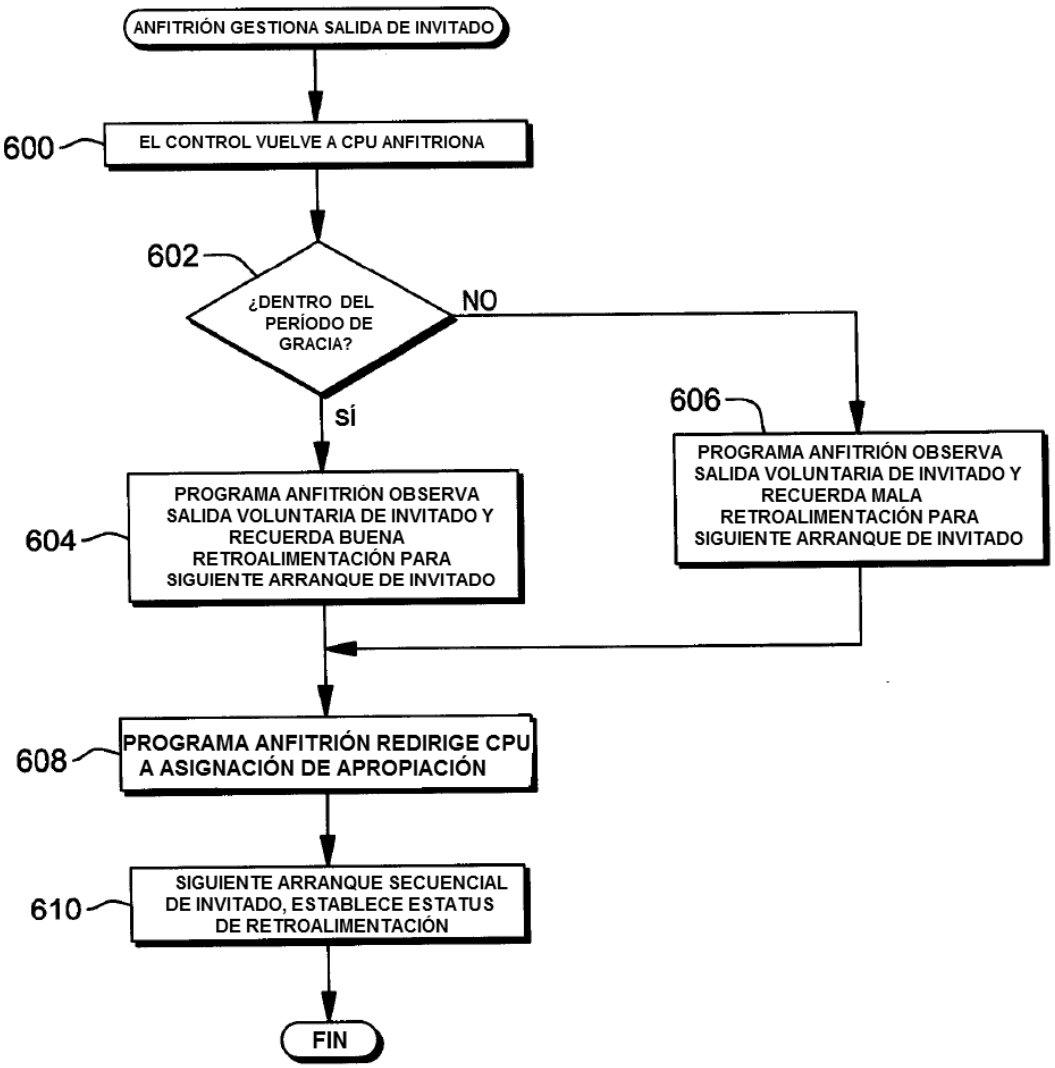
ANFITRIÓN GESTIONA SALIDA DE INVITADO
600
EL CONTROL VUELVE A CPU ANFITRIONA
602
¿DENTRO DEL PERÍODO DE GRACIA?
NO
SÍ
606
PROGRAMA ANFITRIÓN OBSERVA SALIDA VOLUNTARIA DE INVITADO Y RECUERDA MALA RETROALIMENTACIÓN PARA SIGUIENTE ARRANQUE DE INVITADO
604
PROGRAMA ANFITRIÓN OBSERVA SALIDA VOLUNTARIA DE INVITADO Y RECUERDA BUENA RETROALIMENTACIÓN PARA SIGUIENTE ARRANQUE DE INVITADO
608
PROGRAMA ANFITRIÓN REDIRIGE CPU A ASIGNACIÓN DE APROPIACIÓN
610
SIGUIENTE ARRANQUE SECUENCIAL DE INVITADO, ESTABLECE ESTATUS DE RETROALIMENTACIÓN
FIN

FIG. 6

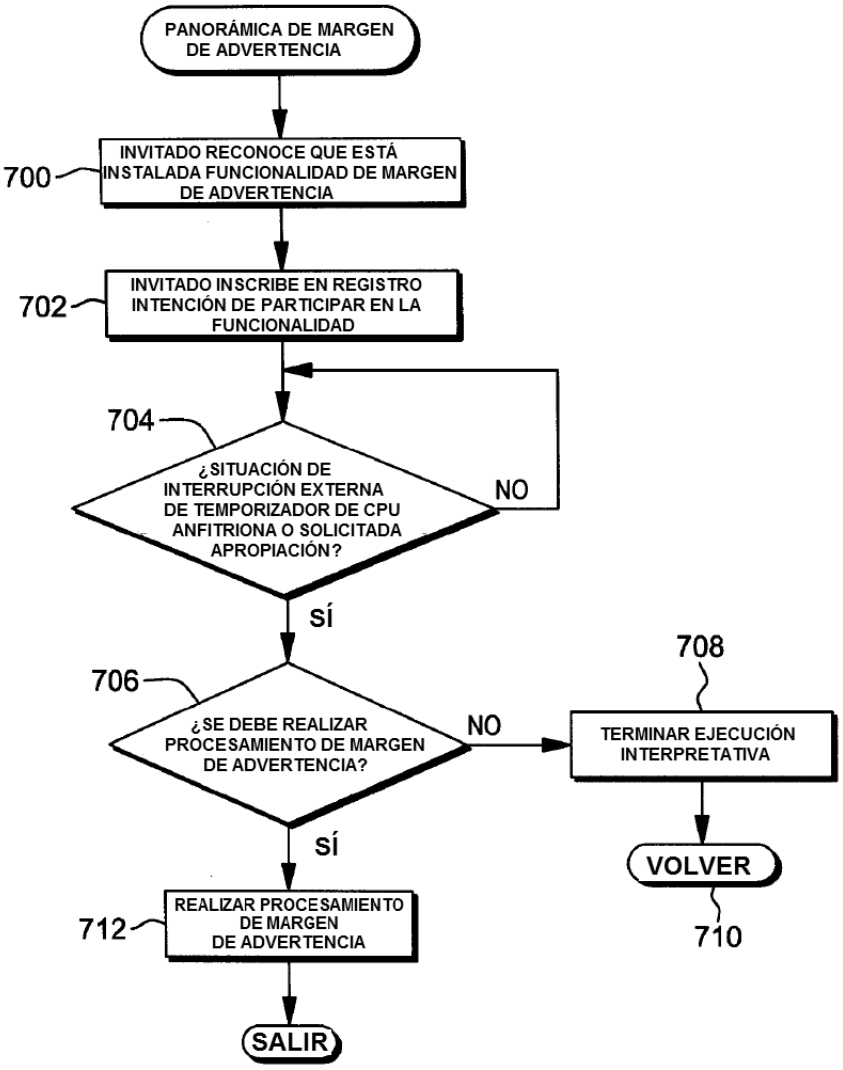
PANORÁMICA DE MARGEN DE ADVERTENCIA
700
INVITADO RECONOCE QUE ESTÁ INSTALADA FUNCIONALIDAD DE MARGEN DE ADVERTENCIA
702
INVITADO INSCRIBE EN REGISTRO INTENCIÓN DE PARTICIPAR EN LA FUNCIONALIDAD
704
¿SITUACIÓN DE INTERRUPCIÓN EXTERNA DE TEMPORIZADOR DE CPU ANFITRIONA O SOLICITADA APROPIACIÓN?
NO
SÍ
706
¿SE DEBE REALIZAR PROCESAMIENTO DE MARGEN DE ADVERTENCIA?
NO
708
TERMINAR EJECUCIÓN INTERPRETATIVA
VOLVER
710
SÍ
712
REALIZAR PROCESAMIENTO DE MARGEN DE ADVERTENCIA
SALIR

FIG. 7

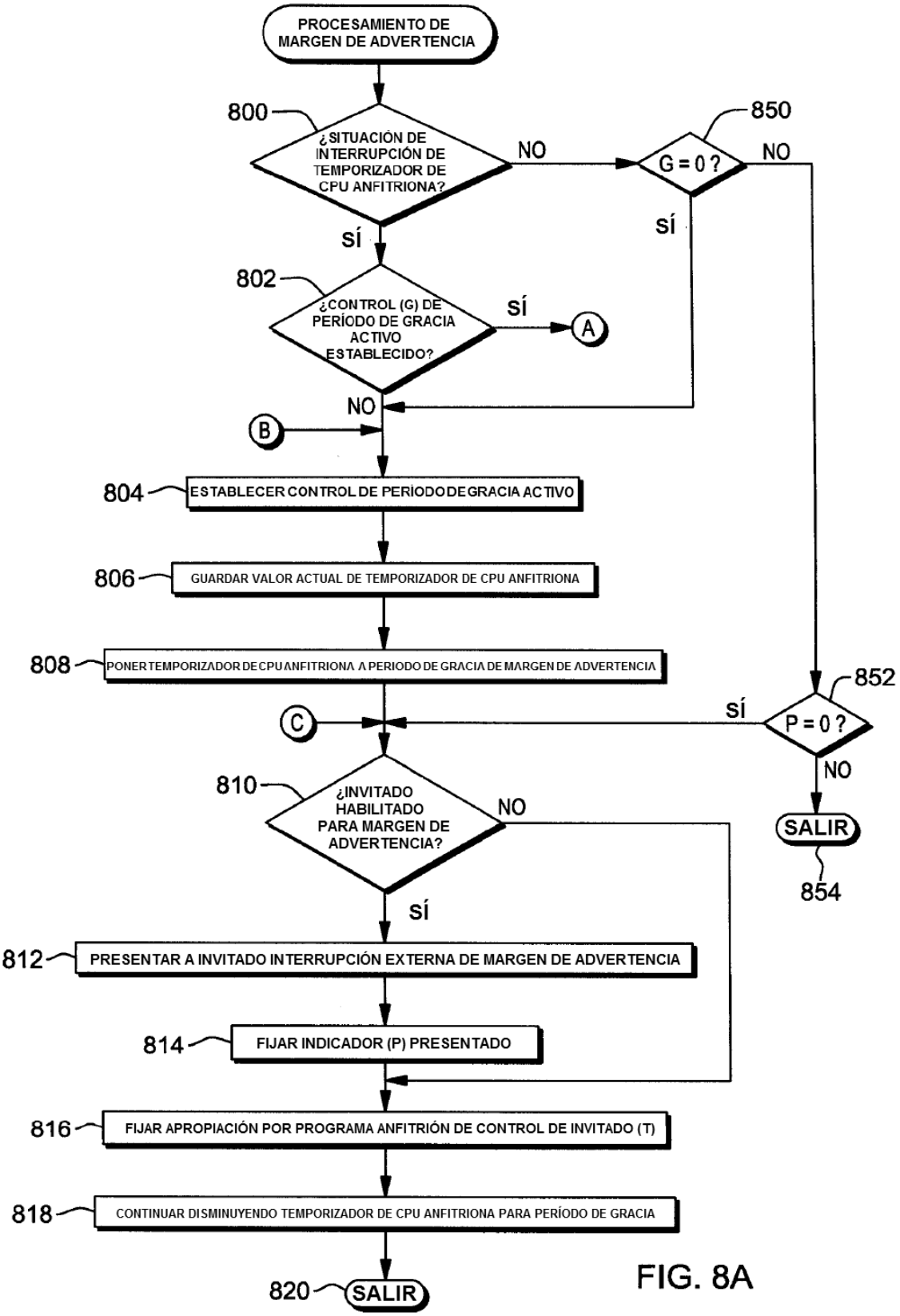
PROCESAMIENTO DE MARGEN DE ADVERTENCIA
800
¿SITUACIÓN DE INTERRUPCIÓN DE TEMPORIZADOR DE CPU ANFITRIONA?
NO
850
G = 0 ?
NO
SÍ
SÍ
802
¿CONTROL (G) DE PERÍODO DE GRACIA ACTIVO ESTABLECIDO?
SÍ
A
NO
B
804
ESTABLECER CONTROL DE PERÍODO DE GRACIA ACTIVO
806
GUARDAR VALOR ACTUAL DE TEMPORIZADOR DE CPU ANFITRIONA
808
PONER TEMPORIZADOR DE CPU ANFITRIONA A PERIODO DE GRACIA DE MARGEN DE ADVERTENCIA
852
SÍ
C
P = 0 ?
NO
SALIR
854
810
¿INVITADO HABILITADO PARA MARGEN DE ADVERTENCIA?
NO
SÍ
812
PRESENTAR A INVITADO INTERRUPCIÓN EXTERNA DE MARGEN DE ADVERTENCIA
814
FIJAR INDICADOR (P) PRESENTADO
816
FIJAR APROPIACIÓN POR PROGRAMA ANFITRIÓN DE CONTROL DE INVITADO (T)
818
CONTINUAR DISMINUYENDO TEMPORIZADOR DE CPU ANFITRIONA PARA PERÍODO DE GRACIA
820
SALIR

FIG. 8A

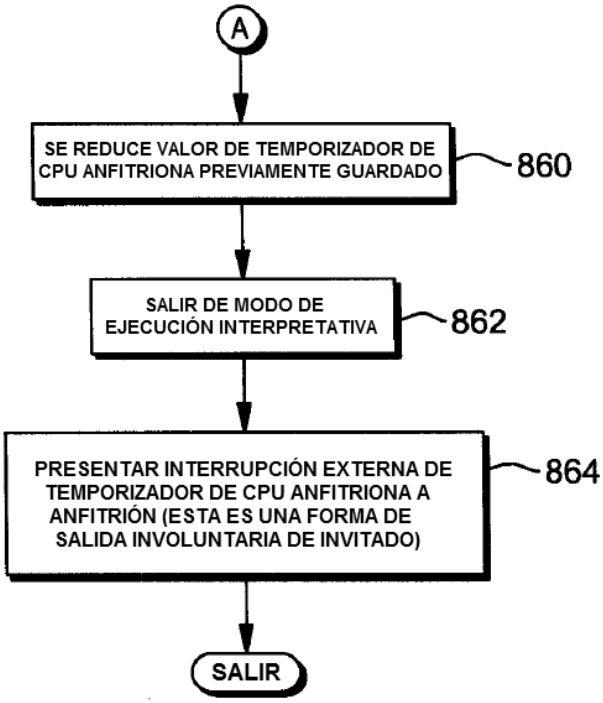
A
SE REDUCE VALOR DE TEMPORIZADOR DE CPU ANFITRIONA PREVIAMENTE GUARDADO
860
SALIR DE MODO DE EJECUCIÓN INTERPRETATIVA
862
PRESENTAR INTERRUPCIÓN EXTERNA DE TEMPORIZADOR DE CPU ANFITRIONA A ANFITRIÓN (ESTA ES UNA FORMA DE SALIDA INVOLUNTARIA DE INVITADO)
864
SALIR

FIG. 8B

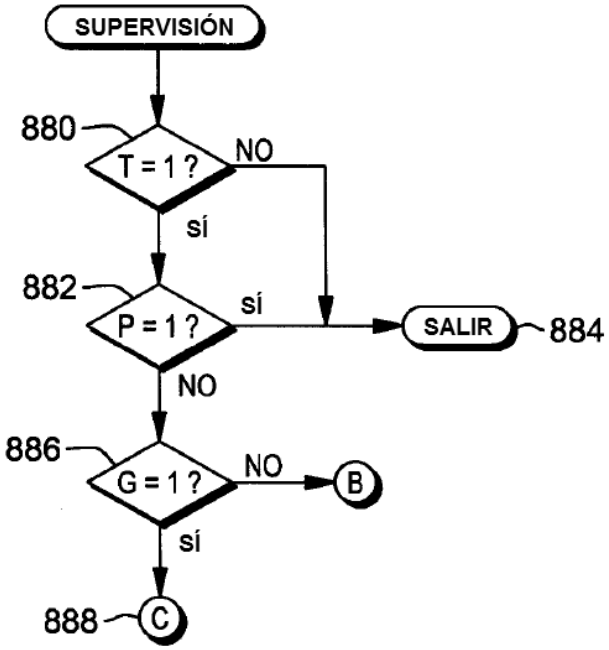
SUPERVISIÓN
880
T = 1 ?
NO
sí
882
P = 1 ?
sí
SALIR
884
NO
886
G = 1 ?
NO
B
sí
888
C

FIG. 8C

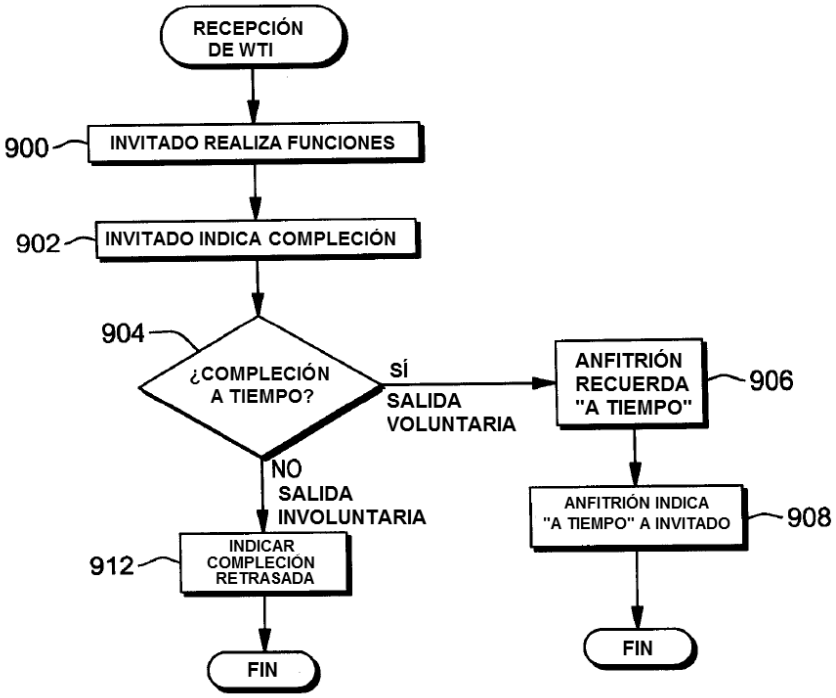
RECEPCIÓN DE WTI
900
INVITADO REALIZA FUNCIONES
902
INVITADO INDICA COMPLECIÓN
904
¿COMPLECIÓN A TIEMPO?
SÍ
SALIDA VOLUNTARIA
ANFITRIÓN RECUERDA "A TIEMPO"
906
NO
SALIDA INVOLUNTARIA
ANFITRIÓN INDICA "A TIEMPO" A INVITADO
908
912
INDICAR COMPLECIÓN RETRASADA
FIN
FIN

FIG. 9

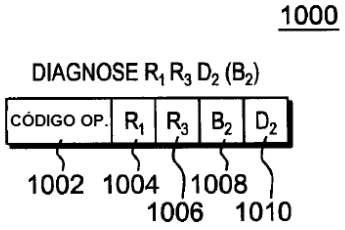
1000
DIAGNOSE R_1 R_3 D_2 (B_2)
CÓDIGO OP.
R_1
R_3
B_2
D_2
1002
1004
1006
1008
1010

FIG. 10


PRODUCTO
DE PROGRAMA
INFORMÁTICO
1100
1104
LÓGICA DE CÓDIGO
DE PROGRAMA
MEDIO DE
ALMACENAMIENTO
LEGIBLE POR
ORDENADOR
1102

FIG. 11

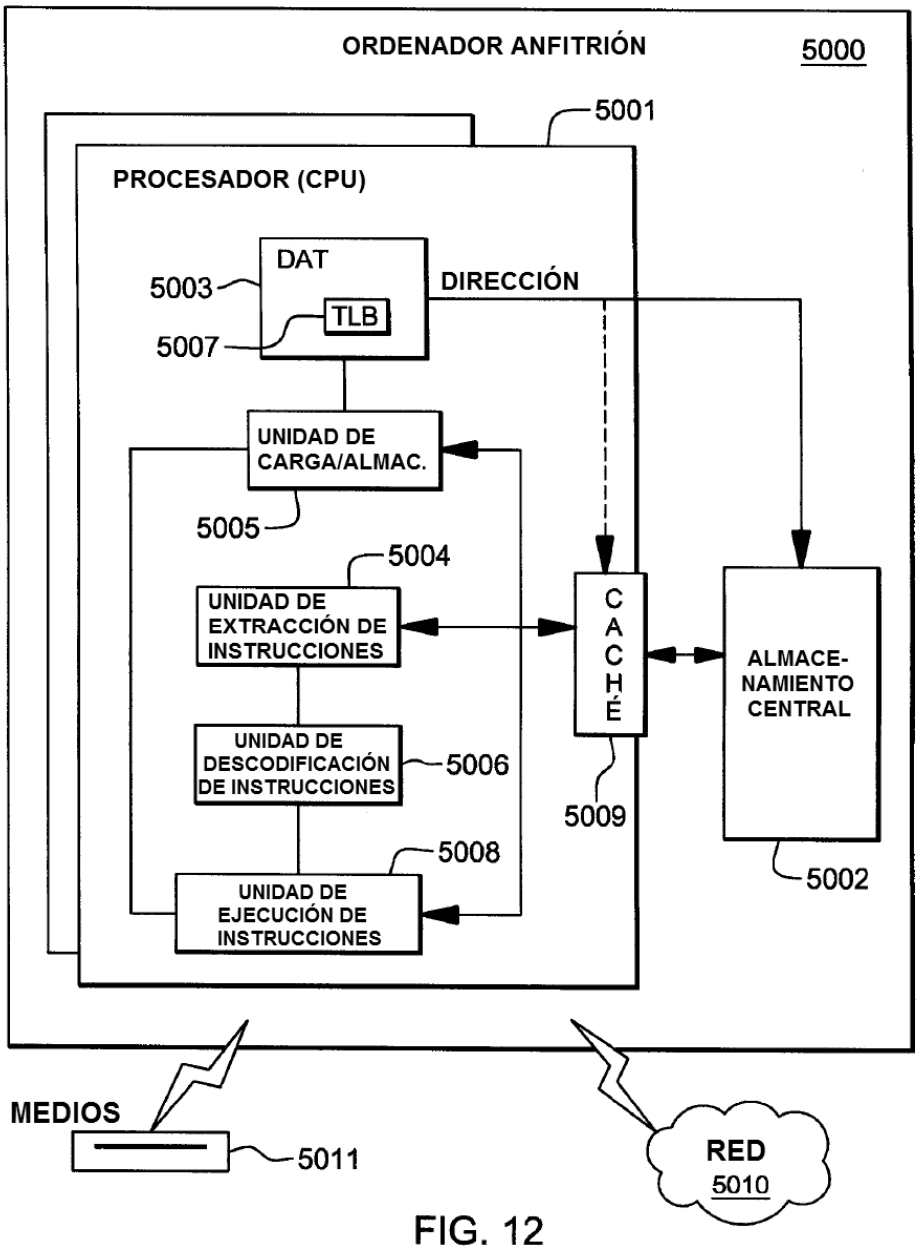
ORDENADOR ANFITRIÓN
5000
5001
PROCESADOR (CPU)
DAT
5003
TLB
5007
DIRECCIÓN
UNIDAD DE CARGA/ALMAC.
5005
5004
UNIDAD DE EXTRACCIÓN DE INSTRUCCIONES
UNIDAD DE DESCODIFICACIÓN DE INSTRUCCIONES
5006
UNIDAD DE EJECUCIÓN DE INSTRUCCIONES
5008
CACHÉ
5009
ALMACE-NAMIENTO CENTRAL
5002
MEDIOS
5011
RED
5010

FIG. 12

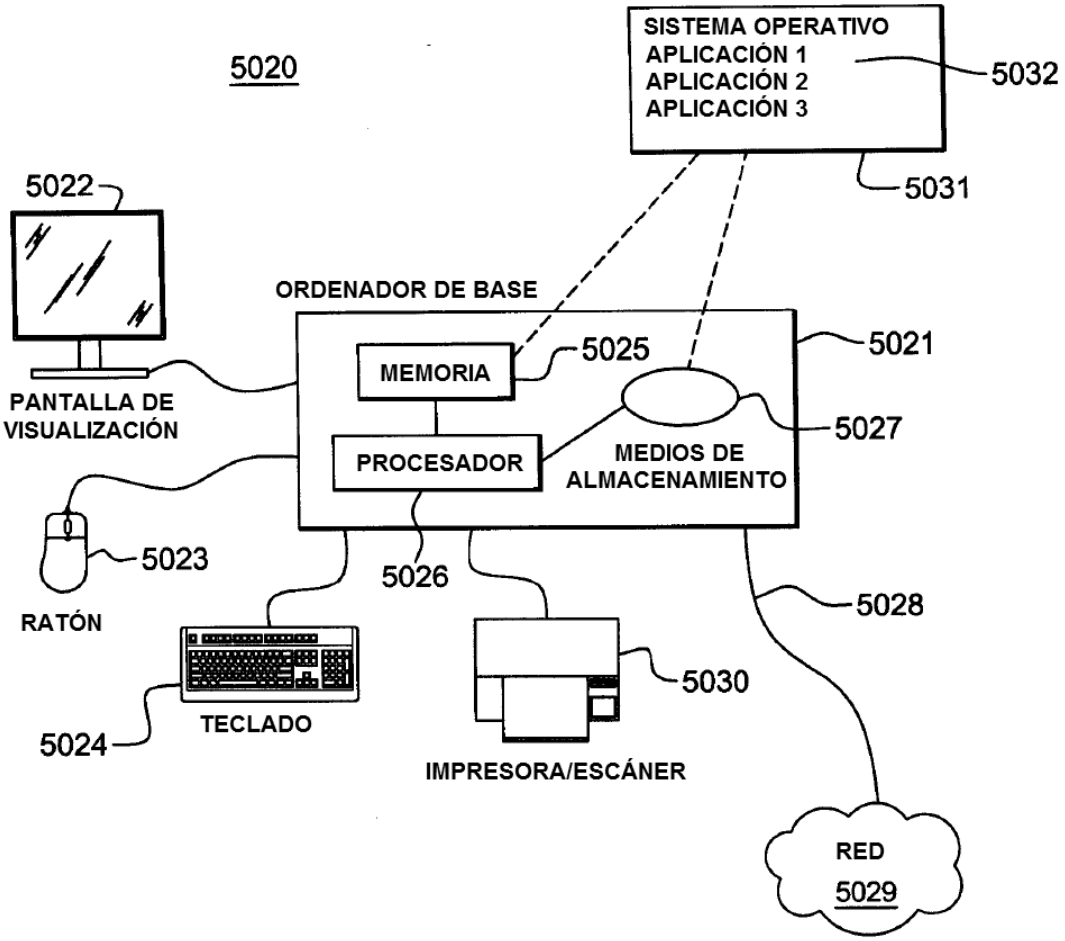
5020
SISTEMA OPERATIVO
APLICACIÓN 1
APLICACIÓN 2
APLICACIÓN 3
5032
5031
5022
ORDENADOR DE BASE
MEMORIA
5025
5021
PANTALLA DE VISUALIZACIÓN
PROCESADOR
MEDIOS DE ALMACENAMIENTO
5027
5023
RATÓN
5026
5028
TECLADO
5024
5030
IMPRESORA/ESCÁNER
RED
5029

FIG. 13

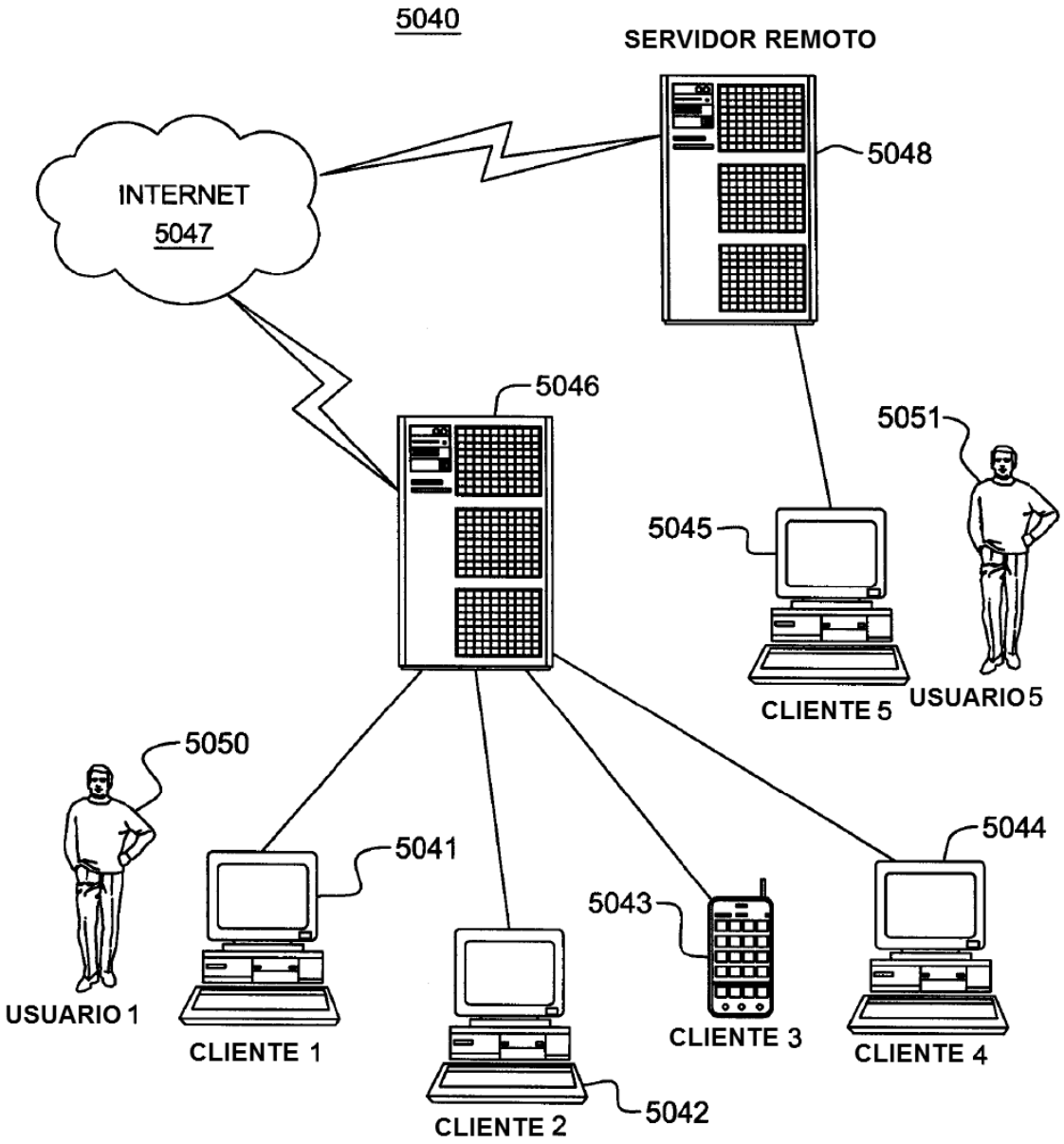
5040
SERVIDOR REMOTO
5048
INTERNET
5047
5046
5051
5045
CLIENTE 5
USUARIO 5
5050
5041
5044
5043
USUARIO 1
CLIENTE 1
CLIENTE 3
CLIENTE 4
CLIENTE 2
5042

FIG. 14

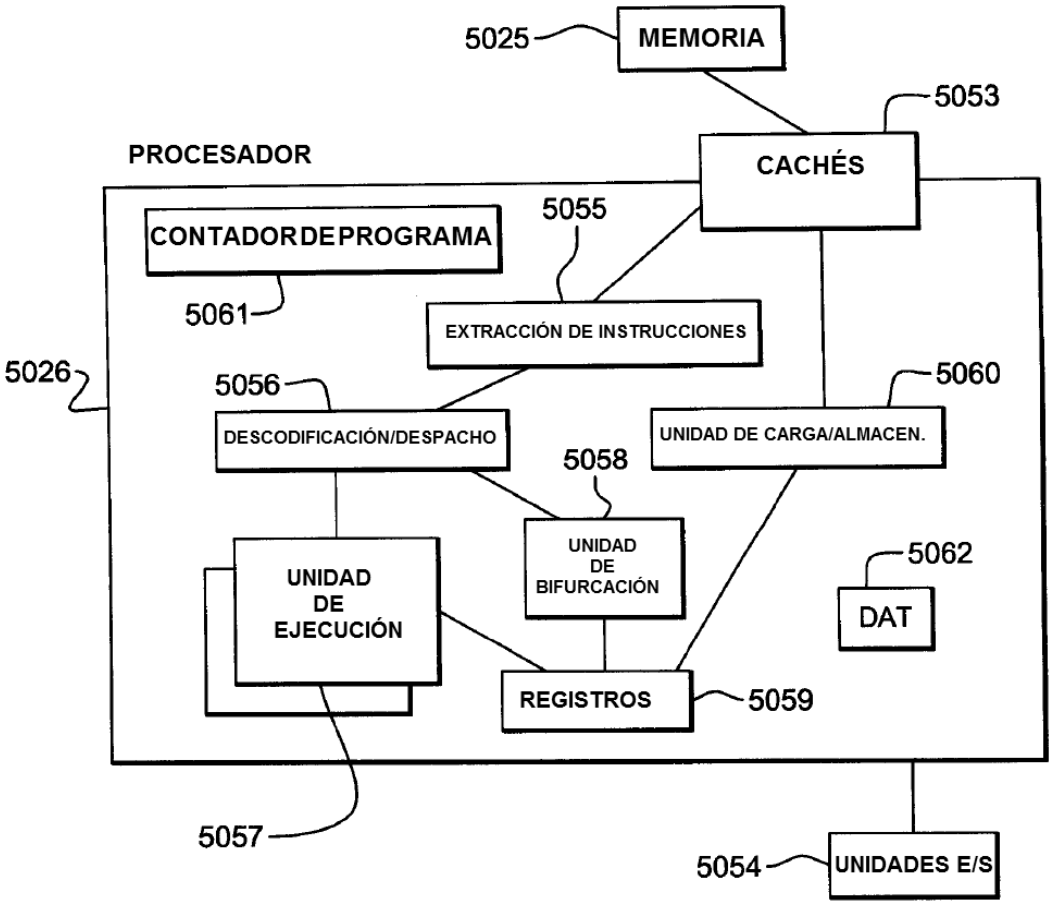
5025
MEMORIA
5053
CACHÉS
PROCESADOR
CONTADOR DE PROGRAMA
5055
5061
EXTRACCIÓN DE INSTRUCCIONES
5026
5056
5060
DESCODIFICACIÓN/DESPACHO
UNIDAD DE CARGA/ALMACEN.
5058
UNIDAD DE BIFURCACIÓN
5062
DAT
UNIDAD DE EJECUCIÓN
REGISTROS
5059
5057
5054
UNIDADES E/S

FIG. 15

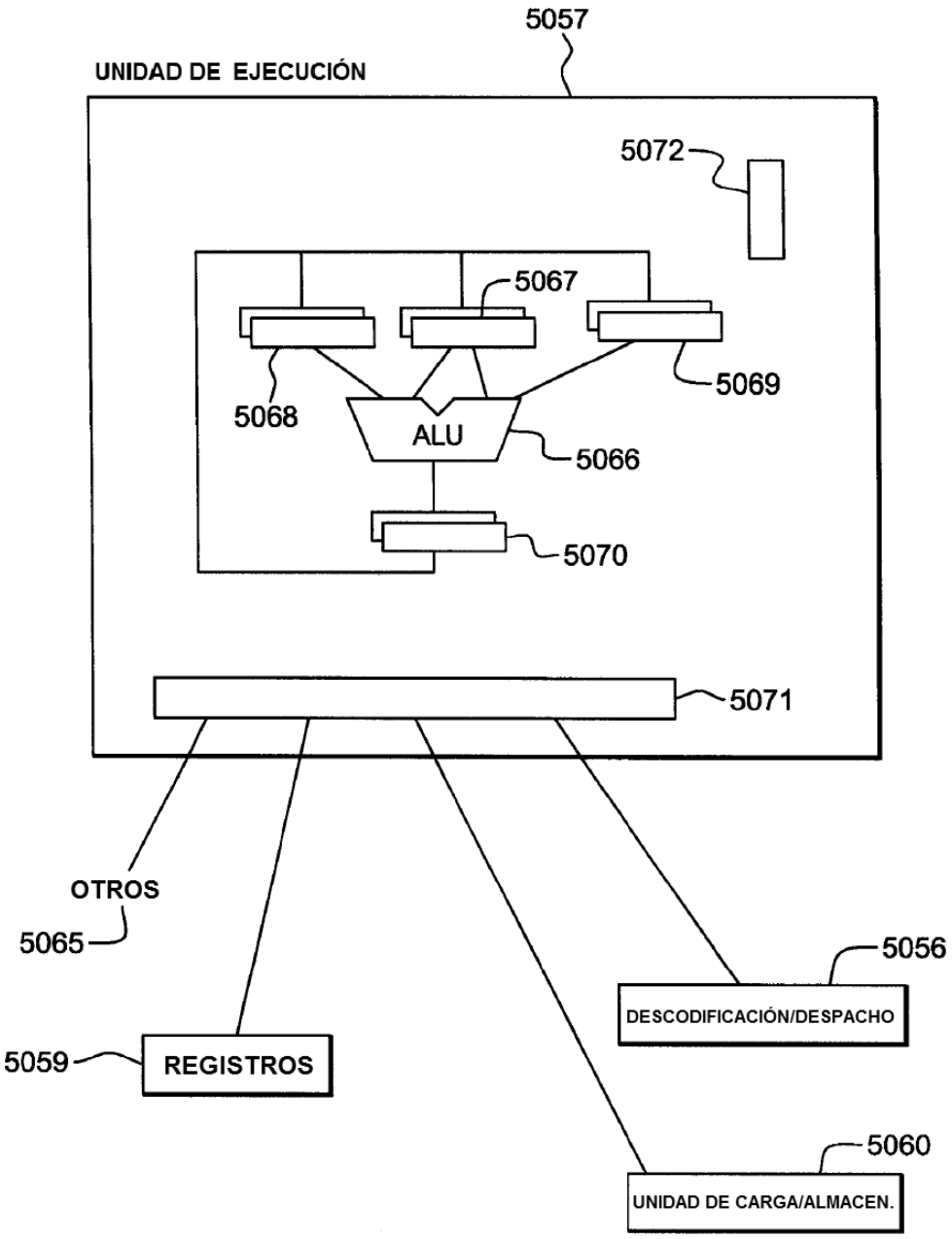
5057
UNIDAD DE EJECUCIÓN
5072
5067
5068
5069
ALU
5066
5070
5071
OTROS
5065
5056
DESCODIFICACIÓN/DESPACHO
5059
REGISTROS
5060
UNIDAD DE CARGA/ALMACEN.

FIG. 16A

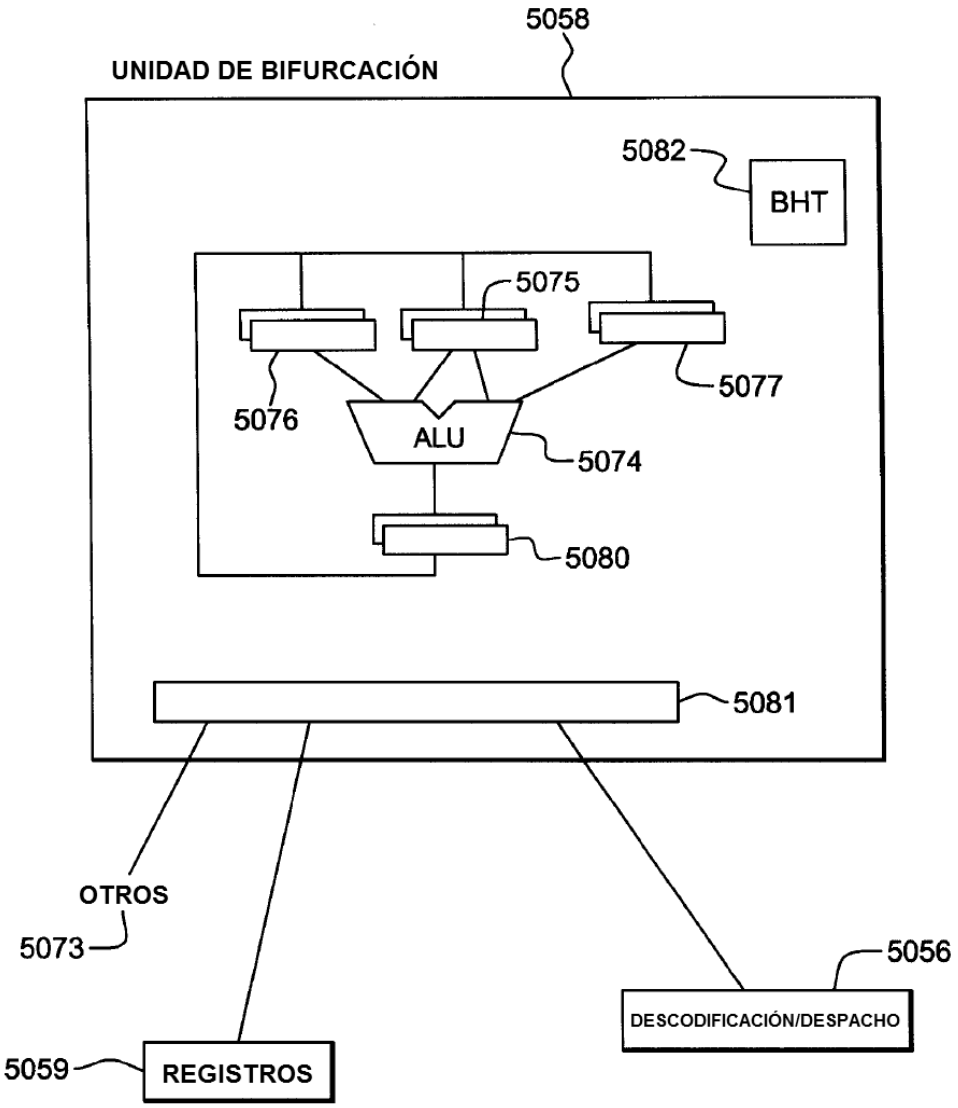
5058
UNIDAD DE BIFURCACIÓN
5082
BHT
5075
5076
5077
ALU
5074
5080
5081
OTROS
5073
5056
DESCODIFICACIÓN/DESPACHO
5059
REGISTROS

FIG. 16B

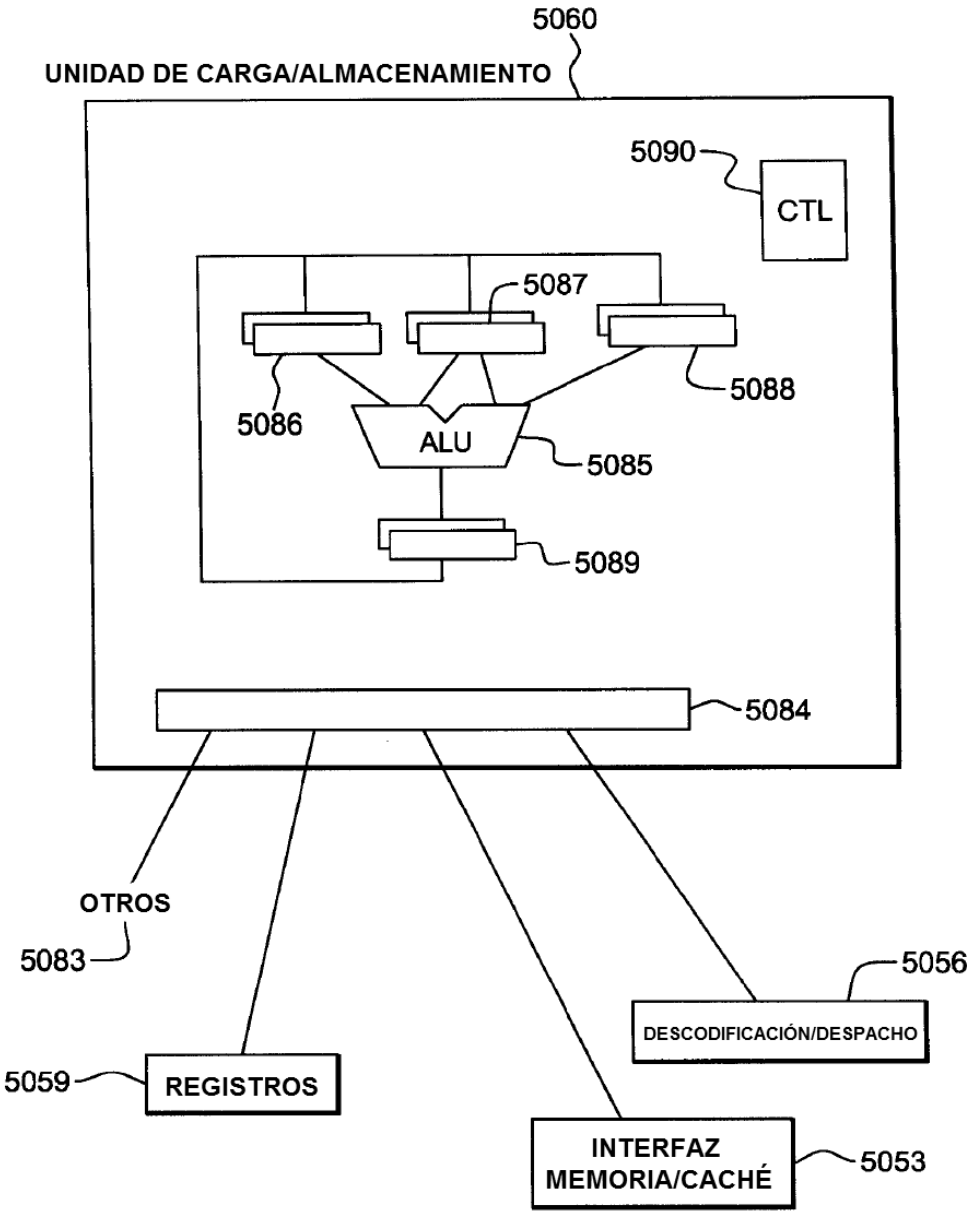
5060
UNIDAD DE CARGA/ALMACENAMIENTO
5090
CTL
5087
5086
5088
ALU
5085
5089
5084
OTROS
5083
5056
DESCODIFICACIÓN/DESPACHO
5059
REGISTROS
INTERFAZ
MEMORIA/CACHÉ
5053

FIG. 16C

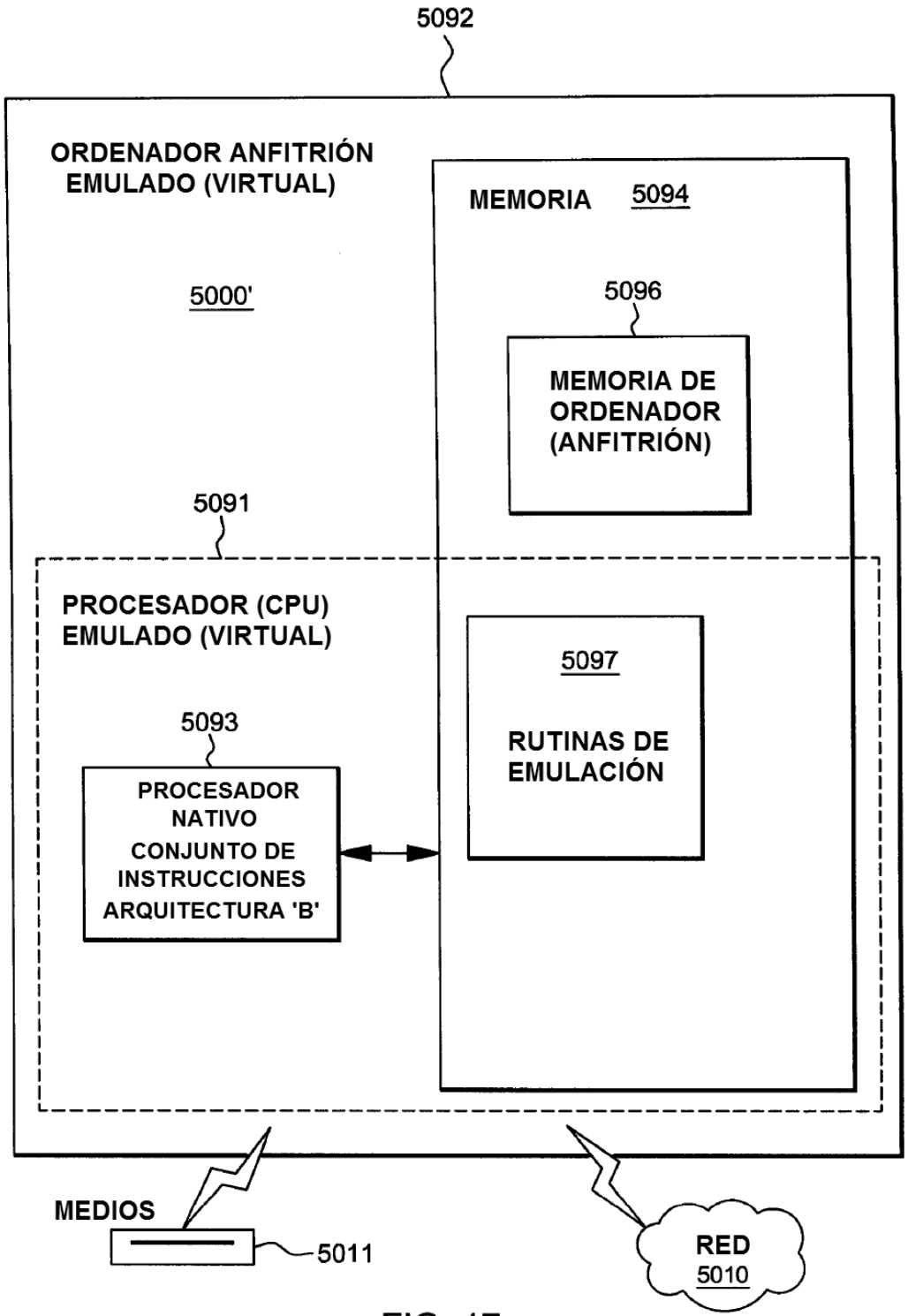
5092
ORDENADOR ANFITRIÓN EMULADO (VIRTUAL)
MEMORIA 5094
5000'
5096
MEMORIA DE ORDENADOR (ANFITRIÓN)
5091
PROCESADOR (CPU) EMULADO (VIRTUAL)
5097
RUTINAS DE EMULACIÓN
5093
PROCESADOR NATIVO CONJUNTO DE INSTRUCCIONES ARQUITECTURA 'B'
MEDIOS
5011
RED
5010

FIG. 17