(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 805 010**

(51) Int. CI.:

G06F 9/48 (2006.01)
G06F 9/38 (2008.01)
G06F 9/30 (2008.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **21.10.2015 PCT/EP2015/074332**

(87) Fecha y número de publicación internacional: **06.05.2016 WO16066486**

(96) Fecha de presentación y número de la solicitud europea: **21.10.2015 E 15787163 (3)**

(97) Fecha y número de publicación de la concesión europea: **24.06.2020 EP 3213187**

(54) Título: **Control de ejecución de hilos en un procesador multihilo**

(30) Prioridad:

28.10.2014 US 201414525800

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
10.02.2021

(73) Titular/es:

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

(72) Inventor/es:

SLEGEL, TIMOTHY;
ALEXANDER, KHARY JASON;
BUSABA, FADI YUSUF;
FARRELL, MARK y
RELL JR, JOHN GILBERT

(74) Agente/Representante:

ISERN JARA, Jorge

DESCRIPCIÓN

Control de ejecución de hilos en un procesador multihilo

Antecedentes

Uno o más aspectos se refieren, en general, a procesadores multihilo y, en particular, al control de la ejecución de hilos en tales procesadores.

Un procesador puede incluir múltiples hilos de hardware que tienen instrucciones ejecutándose simultáneamente. Se dice que un procesador de este tipo implementa multihilo simultáneo (SMT), que es una técnica usada para mejorar la eficiencia general de un procesador que permite que múltiples hilos independientes de ejecución utilicen mejor los recursos proporcionados por arquitecturas de procesadores modernas.

Controlando la ejecución de los hilos de un procesador multihilo, pueden obtenerse eficiencias adicionales.

El documento US 2007/124568 A1 divulga un sistema que da preferencia a un hilo sobre otro hilo en un entorno de procesamiento múltiple.

Sumario

Se superan deficiencias de la técnica anterior y se proporcionan ventajas a través de la provisión de un método de acuerdo con la reivindicación 1, y sistema y producto de programa informático correspondientes de acuerdo con las reivindicaciones 6 y 7, respectivamente.

Se realizan características y ventajas adicionales. Otras realizaciones y aspectos se describen en detalle en este documento y se consideran una parte de la invención reivindicada.

Breve descripción de las diversas vistas de los dibujos

Se señalan particularmente uno o más aspectos y reivindican de forma distintiva como ejemplos en las reivindicaciones al final de la memoria descriptiva. Lo anterior y otros objetos, características y ventajas son evidentes a partir de la siguiente descripción detallada tomada en conjunto con los dibujos adjuntos en los que:

La Figura 1 representa un ejemplo de un entorno informático para incorporar y usar uno o más aspectos de control de ejecución de hilos;
La Figura 2 representa otro ejemplo de un entorno informático para incorporar y usar uno o más aspectos de control de ejecución de hilos;
La Figura 3A representa otro ejemplo más de un entorno informático para incorporar y usar uno o más aspectos de control de ejecución de hilos;
La Figura 3B representa adicionalmente detalles de una memoria del entorno informático de la Figura 3A;
La Figura 3C representa un ejemplo adicional de un entorno informático para incorporar y usar uno o más aspectos de control de ejecución de hilos;
La Figura 4A representa un ejemplo de un registro de control usado de acuerdo con un aspecto de control de ejecución de hilos;
La Figura 4B representa un ejemplo de un registro de dirección de instrucción usado de acuerdo con un aspecto de control de ejecución de hilos;
La Figura 5 representa un ejemplo de etapas de una segmentación;
La Figura 6 representa un ejemplo de lógica para controlar la ejecución de hilos en un procesador multihilo;
La Figura 7A representa un ejemplo de un formato de una instrucción de drenaje;
La Figura 7B representa una realización de lógica asociada con la instrucción de drenaje de la Figura 7A;
La Figura 8A representa un ejemplo de un formato de una instrucción de Comparar y Conmutar Registro de Unidad R;
La Figura 8B representa una realización de lógica asociada con la instrucción de Comparar y Conmutar Registro de Unidad R de la Figura 8A;
La Figura 9A representa un ejemplo de un formato de una instrucción de Cargar y Registro O de Unidad R;
La Figura 9B representa una realización de lógica asociada con la instrucción de Cargar y Registro O de Unidad R de la Figura 9A;
La Figura 10A representa un ejemplo de un formato de una instrucción de Cargar y Registro Y de Unidad R;
La Figura 10B representa una realización de lógica asociada con la instrucción de Cargar y Registro Y de Unidad R de la Figura 10A;
Las Figuras 11A-11B representan un ejemplo de lógica asociada con el enclavamiento usado por una o más instrucciones; y
La Figura 12 representa una realización de un producto de programa informático.

Descripción detallada

De acuerdo con uno o más aspectos, se proporciona una capacidad para controlar la ejecución de hilos (por ejemplo, hilos de hardware) en un núcleo (por ejemplo, un procesador de hardware físico; también denominado en este documento como un procesador o núcleo de procesador) que opera dentro de un entorno informático. El núcleo soporta, por ejemplo, multihilo, tal como multihilo simultáneo (SMT), que significa que pueden haber de forma efectiva múltiples unidades de procesamiento central (CPU) lógicas operando simultáneamente en el mismo hardware de procesador físico. Cada una de estas CPU lógicas se considera un hilo.

En un entorno multihilo simultáneo de este tipo, puede desearse que un hilo detenga la ejecución de otros hilos en el núcleo de procesador. Esto puede ser en respuesta a que se lleve a cabo una secuencia crítica u otra secuencia que necesita los recursos de núcleo de procesador o está manipulando recursos de núcleo de procesador de una forma que otros hilos interferirían con su ejecución. En un ejemplo, como parte de la capacidad, puede ser deseable esperar hasta que alguna condición se haya satisfecho para todos los hilos en el núcleo de procesador. Por ejemplo, supóngase que el software o firmware que se está llevando a cabo en un hilo de hardware particular quiere realizar una acción de sistema que primero requiere que ningún almacenamiento esté en curso desde todo el núcleo de procesador, es decir, ningún almacenamiento está en curso en todos los hilos en el núcleo de procesador. Para determinar si los otros hilos se detienen, se proporciona una instrucción, denominada en este documento como una instrucción de drenaje, de acuerdo con un aspecto, que supervisa el estado de los hilos en el núcleo de procesador.

Además, de acuerdo con uno o más aspectos, en el control de ejecución de los hilos, pueden usarse diversas instrucciones atómicas. Estas instrucciones operan en registros accesibles a y compartidos por los hilos del procesador de SMT, en lugar de almacenamiento o memoria. (Memoria y almacenamiento se usan indistintamente en este documento, a menos que se indique de otra manera implícita o explícitamente). Esto permite que múltiples hilos se comuniquen y compartan información usando los registros compartidos, en lugar de almacenamiento. Estas instrucciones, denominadas en este documento como instrucción de Comparar y Conmutar Registro de Unidad R o Comparar y Conmutar Registro, instrucción de Cargar y Registro O de Unidad R o Cargar y Registro O, e instrucción de Cargar y Registro Y de Unidad R o Cargar y Registro Y, controlan el acceso a los registros compartidos usando enclavamiento, como se describe en este documento.

Un ejemplo de un entorno informático para incorporar y usar uno o más aspectos de control de ejecución de hilos se describe con referencia a la Figura 1. Haciendo referencia a la Figura 1, en un ejemplo, un entorno informático 100 se basa en la z/Architecture, ofrecida por International Business Machines (IBM®) Corporation, Armonk, Nueva York. La z/Architecture se describe en una publicación de IBM titulada "z/Architecture - Principles of Operation", Publicación N.º SA22-7832-09, 10ª edición, septiembre de 2012.

z/Architecture, IBM, y Z/VM, Z/OS, POWER y POWERPC (referenciados en este documento) son marcas registradas de International Business Machines Corporation, Armonk, Nueva York. Otros nombres usados en este documento pueden ser marcas comerciales registradas, marcas comerciales o nombres de producto de International Business Machines Corporation u otras compañías.

Como un ejemplo, entorno informático 100 incluye un complejo de procesador central (CPC) 102 acoplado a uno o más dispositivos de entrada/salida (E/S) 106 a través de una o más unidades de control 108. El complejo de procesador central 102 incluye, por ejemplo, una memoria 104 de procesador (también conocida como, memoria principal, almacenamiento principal, almacenamiento central) acoplada a uno o más núcleos de procesador 110, y un subsistema de entrada/salida 111, cada uno de los cuales se describe a continuación.

La memoria 104 de procesador incluye, por ejemplo, una o más particiones 112 (por ejemplo, particiones lógicas) y firmware 113 de procesador, que incluye, por ejemplo, un hipervisor de particiones lógicas 114 y otro firmware 115 de procesador. Un ejemplo de hipervisor de particiones lógicas 114 es el Recurso de Procesador/Gestor de Sistema (PR/SM), presentado por International Business Machines Corporation, Armonk, Nueva York.

Una partición lógica funciona como un sistema separado y tiene una o más aplicaciones 120, y opcionalmente, un sistema operativo residente 122 en el mismo, que puede diferir para cada partición lógica. En una realización, el sistema operativo es el sistema operativo z/OS, el sistema operativo z/VM, el sistema operativo z/Linux, o el sistema operativo TPF, presentado por International Business Machines Corporation, Armonk, Nueva York.

Las particiones lógicas 112 se gestionan mediante el hipervisor de particiones lógicas 114, que se implementa por firmware que se lleva a cabo en los núcleos 110. Como se usa en este documento, firmware incluye, por ejemplo, el microcódigo y/o milicódigo del núcleo de procesador. Incluye, por ejemplo, las instrucciones a nivel de hardware y/o estructuras de datos usadas en la implementación de código máquina de nivel superior. En una realización, incluye, por ejemplo, código propietario que se entrega típicamente como microcódigo que incluye software confiable o microcódigo específico para el hardware subyacente y controla el acceso del sistema operativo al sistema hardware.

Los núcleos 110 de procesador son recursos de procesador físicos asignados a las particiones lógicas. En particular, cada partición lógica 112 tiene uno o más procesadores lógicos, cada uno de los cuales representa todo o una parte de un núcleo 110 asignado a la división. Los procesadores lógicos de una partición 112 particular pueden o bien

dedicarse a la división, de modo que el recurso de núcleo subyacente 110 se reserva para esa partición; o bien compartirse con otra partición, de modo que el recurso de núcleo subyacente está potencialmente disponible para otra partición.

En un ejemplo, al menos uno de los núcleos es un procesador multihilo, tal como un procesador multihilo simultáneo, que incluye múltiples hilos (es decir, múltiples CPU lógicas que operan simultáneamente). En un ejemplo, el núcleo incluye dos hilos, pero en otras realizaciones, puede haber más de dos hilos. Dos hilos, denominados en este documento como T0 (126) y T1 (128), son únicamente un ejemplo.

En soporte de multihilo simultáneo, el hardware de núcleo de procesador contiene el estado de arquitectura completo (por ejemplo, estado de z/Architecture y microarquitectura) para cada hilo. Por lo tanto, se proporcionan registros para todo el procesador 130, que son comunes a todos los hilos (denominados en este documento como registros comunes), así como registros específicos 132 de hilo, que son únicos a un hilo (denominado en este documento como registros únicos). A continuación, se describe adicionalmente el uso de estos registros.

Para controlar la ejecución de los múltiples hilos, el núcleo 110 incluye hardware y/o lógica para proporcionar tal control como se describe en este documento. Este hardware y/o lógica se denomina en este documento como un servicio de control 134 de hilos por conveniencia.

El subsistema de entrada/salida 111 dirige el flujo de información entre dispositivos de entrada/salida 106 y almacenamiento principal 104. Se acopla al complejo de procesamiento central, en que puede ser una parte del complejo de procesamiento central o estar separado del mismo. El subsistema de E/S releva los núcleos de procesador de la tarea de comunicarse directamente con los dispositivos de entrada/salida y permite que el procesamiento de datos continúe simultáneamente con procesamiento de entrada/salida. Para proporcionar comunicaciones, el subsistema de E/S emplea adaptadores de comunicación de E/S. Estos son diversos tipos de adaptadores de comunicaciones que incluyen, por ejemplo, canales, adaptadores de E/S, tarjetas de PCI, tarjetas de Ethernet, tarjetas de Interfaz Pequeña de Almacenamiento Informático (SCSI), etc. En el ejemplo particular descrito en este documento, los adaptadores de comunicación de E/S son canales y, por lo tanto, el subsistema de E/S se denomina en este documento como un subsistema de canal. Sin embargo, esto es únicamente un ejemplo. Pueden usarse otros tipos de subsistemas de E/S.

El subsistema de E/S usa una o más trayectorias de entrada/salida como enlaces de comunicación en la gestión del flujo de información a o desde los dispositivos de entrada/salida 106. En este ejemplo particular, estas trayectorias se llaman trayectorias de canal, ya que los adaptadores de comunicación son canales.

Otro ejemplo de un entorno informático para incorporar y usar uno o más aspectos de control de ejecución de hilos se describe con referencia a la Figura 2. En este ejemplo, un entorno informático 200 incluye un entorno no particionado implementado basándose en la z/Architecture (u otra arquitectura en otra realización). Incluye un núcleo 202 que incluye, por ejemplo, una o más memorias caché 204; al menos dos hilos, T0 (206), T1 (208); un conjunto común de registros 210 para los hilos; y un conjunto único de registros 212 para cada hilo, así como un servicio de control 214 de hilos.

El núcleo 202 se acopla comunicativamente a una memoria 216 que tiene una o más memorias caché 218 y al menos un servicio de control 220, tal como un sistema operativo; y a un subsistema de entrada/salida (E/S) 222. El subsistema de E/S 222 se acopla comunicativamente a dispositivos de E/S 224 externos que pueden incluir, por ejemplo, dispositivos de entrada de datos, sensores y/o dispositivos de salida, tal como visualizadores.

Otra realización de un entorno informático para incorporar y usar uno o más aspectos de control de ejecución de hilos se describe con referencia a la Figura 3A. En este ejemplo, un entorno informático 300a incluye, por ejemplo, un núcleo nativo 302, una memoria 304 y uno o más dispositivos de entrada/salida y/o interfaces 306 acoplados entre sí a través de, por ejemplo, uno o más buses 308 y/u otras conexiones. Como ejemplos, el entorno informático 300a puede incluir un procesador PowerPC o un servidor Power Systems presentados por International Business Machines Corporation, Armonk, Nueva York; un HP Superdome con procesadores Intel Itanium II ofrecido por Hewlett Packard Co., Palo Alto, California; y/u otras máquinas basadas en arquitecturas ofrecidas por International Business Machines Corporation, Hewlett Packard, Intel, Oracle u otros.

El núcleo nativo 302 incluye uno o más registros nativos 310, tal como uno o más registros de fin general y/o uno o más registros de fin especial usados durante el procesamiento dentro del entorno que incluyen información que representa el estado del entorno en cualquier punto en tiempo particular. Además, el núcleo nativo puede incluir, por ejemplo, al menos dos hilos, T0 (311), T1 (313); un conjunto de registros comunes 315 para los hilos; un conjunto de registros específicos de hilo 317 para cada hilo; y un servicio de control 319 de hilos.

Además, el núcleo nativo 302 ejecuta instrucciones y código que se almacenan en la memoria 304. En un ejemplo particular, el núcleo de procesador ejecuta el código emulador 312 almacenado en la memoria 304. Este código habilita que el entorno informático configurado en una arquitectura emule una o más otras arquitecturas. Por ejemplo, el código emulador 312 permite que máquinas basadas en arquitecturas distintas de la z/Architecture, tal como procesadores

PowerPC, servidores Power Systems, servidores HP Superdome u otros, emulen la z/Architecture y ejecuten software e instrucciones desarrolladas basadas en la z/Architecture.

En una realización adicional, como se muestra en la Figura 3C, el núcleo 302 es un núcleo de un solo hilo, pero se está emulando un núcleo multihilo e incluyéndose dentro del código emulador 312. Por ejemplo, el código emulador 312 incluye un servicio de control 320 de hilos emulados; hilos emulados 322, 324; registros comunes emulados 326 y registros únicos emulados 328, cada uno de los cuales se basa en una arquitectura diferente de la arquitectura del núcleo nativo 302, tal como la z/Architecture.

Detalles adicionales relacionados con el código emulador 312 se describen con referencia a la Figura 3B. Las instrucciones de invitado 350 almacenadas en la memoria 304 comprenden instrucciones de software (por ejemplo, correlacionadas con instrucciones de máquina) que se desarrollaron para ejecutarse en una arquitectura distinta de la del núcleo nativo 302. Por ejemplo, las instrucciones de invitado 350 pueden haberse diseñado para ejecutarse en un núcleo de z/Architecture 202, pero en su lugar, se están emulando en el núcleo nativo 302, que puede ser, por ejemplo, un procesador Intel Itanium II. En un ejemplo, el código emulador 312 incluye una rutina de búsqueda 352 de instrucciones para obtener una o más instrucciones de invitado 350 desde la memoria 304, y para proporcionar opcionalmente almacenamiento en memoria intermedia local para las instrucciones obtenidas. También incluye una rutina 354 de traducción de instrucción para determinar el tipo de instrucción de invitado que se ha obtenido y para traducir la instrucción de invitado en una o más instrucciones nativas 356 correspondientes. Esta traducción incluye, por ejemplo, identificar la función a realizarse por la instrucción de invitado y elegir la instrucción o instrucciones nativas para realizar esa función.

Además, el código emulador 312 incluye una rutina 360 de control de emulación para provocar que las instrucciones nativas se ejecuten. La rutina 360 de control de emulación puede provocar que el núcleo nativo 302 ejecute una rutina de instrucciones nativas que emulan una o más instrucciones de invitado anteriormente obtenidas y, en la finalización de tal ejecución, devuelva el control a la rutina de búsqueda de instrucción para emular la obtención de la siguiente instrucción de invitado o un grupo de instrucciones de invitado. Las instrucciones de invitado pueden ser instrucciones del servicio de control de hilos descrito en este documento. La ejecución de las instrucciones nativas 356 puede incluir cargar datos en un registro desde la memoria 304; almacenar datos de vuelta a la memoria desde un registro; o realizar algún tipo de operación aritmética o lógica, como se determina por la rutina de traducción.

Cada rutina se implementa, por ejemplo, en software, que se almacena en memoria y ejecuta por el núcleo nativo 302. En otros ejemplos, una o más de las rutinas u operaciones se implementan en firmware, hardware, software o alguna combinación de los mismos. Los registros del procesador emulado pueden emularse usando los registros 310 del núcleo nativo o usando ubicaciones en la memoria 304. En las realizaciones, las instrucciones de invitado 350, instrucciones nativas 356 y el código emulador 312 pueden residir en la misma memoria o pueden distribuirse entre diferentes dispositivos de memoria.

Los entornos informáticos descritos anteriormente son únicamente ejemplos de entornos informáticos que pueden usarse. Pueden usarse otros entornos, incluyendo, pero sin limitación a, otros entornos sin particiones, otros entornos con particiones y/u otros entornos emulados; realizaciones no se limitan a un entorno cualquiera.

Como se ha indicado anteriormente, hay una pluralidad de registros asociados con cada hilo. Un registro compartido común a los hilos es un registro de control, tal como un registro de control de milicódigo (MCR), MCR002, un ejemplo del cual se representa en la Figura 4A. MCR002 (400) incluye diversos controles para SMT que determina cómo se comportan los hilos. En una realización, MCR002 (400) incluye una pluralidad de campos 402, y esos campos usados de acuerdo con uno o más aspectos incluyen, por ejemplo:

(a) Un campo 404 de parada transitoria de búsqueda de instrucción (I-fetch): los dos bits de este campo se corresponden uno a uno con hilos 0 y 1 (si hubiera más de dos hilos, entonces puede haber más de dos bits). Cuando un bit es '1'b, este se vuelve una anulación maestra transitoria efectiva para bloquear búsqueda de instrucción independientemente del estado de otros bits de control; y

(b) Un campo 406 de no se permite parada de búsqueda de instrucción: los dos bits de este campo se corresponden uno a uno con hilos 0 y 1 (si hubiera más de dos hilos, entonces puede haber más de dos bits). Cuando un bit es '1'b, indica que este hilo está entrando en una sección de código (por ejemplo, sección crítica) en la que no se permite que el otro hilo active el bit de detener búsqueda de instrucción para este hilo.

Otro registro usado es un registro de dirección de instrucción, que es único para cada hilo. Este registro, denominado como IAREGFA, incluye información acerca de una interrupción de programa detectada por hardware. Un ejemplo de IAREGFA se representa en la Figura 4B. Como se muestra, IAREGFA 450 incluye una pluralidad de campos 452. Un campo usado de acuerdo con uno o más aspectos es el campo 454 que indica que el hilo está en proceso de tomar una excepción.

Cada uno de los registros anteriores puede incluir campos adicionales, menos y/o diferentes. Además, puede haber otros registros que se usan. Los registros y campos descritos en este documento son ejemplos de registros y/o campos

que pueden usarse. Además, MCR y IAREGFA son solo ejemplos de nombres de los registros. Son posibles muchas variaciones.

Para aumentar el caudal de instrucciones, cada hilo usa una segmentación de instrucción para procesar que se permita que se realicen múltiples operaciones al mismo tiempo. Una segmentación de instrucción incluye una pluralidad de etapas, y un ejemplo de una segmentación de este tipo se describe con referencia a la Figura 5. Haciendo referencia a la Figura 5, una segmentación 500 que soporta procesamiento fuera de orden, incluye, por ejemplo, una etapa de búsqueda 502 de instrucción en la que se buscan instrucciones desde la memoria; una etapa de decodificación/expedición 504 de instrucción que forma grupos de expedición/finalización y pone instrucciones en la cola de emisión; una etapa de emisión 506 en la que se emiten las instrucciones (fuera de orden); una etapa de ejecución 508 en la que se ejecutan las instrucciones (fuera de orden); una etapa final 510 en la que se finalizan instrucciones (fuera de orden); una etapa de finalización 512 que se refiere a un punto de comprobación de arquitectura; y una etapa de punto de comprobación de recuperación 514. Otras segmentaciones pueden incluir etapas adicionales, menos y/o diferentes. Las etapas descritas en este documento son únicamente ejemplos.

En un ejemplo, pueden situarse hasta tres instrucciones (en particular, micro operaciones) en un grupo. Sin embargo, ciertas instrucciones, tal como instrucciones de bifurcación, finalizan un grupo incluso si no está lleno. Un grupo completo de instrucciones se conduce a la misma cola de emisión y, a continuación, el siguiente grupo va a otra cola de emisión.

De acuerdo con un aspecto de la presente invención, se proporciona una capacidad para que un hilo que se está llevando a cabo en un núcleo detenga uno o más hilos distintos que se están ejecutando dentro del núcleo para realizar una o más operaciones. En los ejemplos descritos en este documento, el núcleo es un diseño de SMT-2 que indica que hay dos hilos. Sin embargo, en otras realizaciones, puede haber más de dos hilos.

Una realización de la lógica usada para controlar la ejecución de uno o más hilos se describe con referencia a la Figura 6. En este ejemplo, el hilo 0 (T0) que se ejecuta en un núcleo está intentado detener el hilo 1 (T1) que se ejecuta en el núcleo y, por lo tanto, la descripción se refiere a T0 y T1; sin embargo, en otras realizaciones, T1 puede estar intentando detener T0; y/o puede haber más de un hilo ejecutándose en el núcleo que se detiene. Por ejemplo, T0 puede estar deteniendo T1, T2, T3, etc. Además, en los ejemplos descritos en este documento, la lógica se realiza mediante firmware del núcleo; sin embargo, en una o más otras realizaciones, puede realizarse mediante software de fin general. Son posibles muchas otras variaciones.

Como se describe con referencia a la Figura 6, en una realización, un hilo detiene la ejecución de otro hilo, y la parada usa uno o más controles (por ejemplo, indicadores, bits, etc.) en uno o más registros (por ejemplo, registros de hardware) compartidos por los hilos.

Haciendo referencia a la Figura 6, en una realización, el hilo 0 comprueba si se prohíbe que se detenga T1 (o en otras realizaciones, uno o más hilos del núcleo), ETAPA 600. En un ejemplo, esto se determina comprobando un bit seleccionado (por ejemplo, el bit 25) de MCR002, así como un bit seleccionado (por ejemplo, el bit 4) de IAREGFA. Esto se logra, en un ejemplo, mediante T0 probando un punto de bifurcación, denominado como un STPIFALW. STPIFALW prueba los bits seleccionados de MCR002 y IAREGFA. Por ejemplo, si MCR0002.25 (es decir, el bit 25 de MCR002) se establece a cero y IAREGIFA.4 (es decir, bit 4 de IAREGIFA) se establece a cero, entonces se permite la parada de búsqueda de instrucción de T1.

Si STPIFALW indica que se prohíbe que se detenga T1, CONSULTA 602, entonces el procesamiento continúa a la ETAPA 600. Sin embargo, si no se está prohibiendo que se detenga T1, como se indica por STPIFALW, y en particular MCR002.25=0 y IAREGFA.4=0, entonces el procesamiento continúa deteniendo T0 la búsqueda y ejecución de instrucción en T1, ETAPA 604. En un ejemplo, esto incluye T0 estableciendo el bit de parada transitoria de búsqueda de instrucción para T1 (por ejemplo, MCR002.9), que detiene la búsqueda y ejecución de instrucción en T1. Este bit se establece usando, por ejemplo, una instrucción de Comparar y Conmutar Registro de Unidad R (CSGRU) o una instrucción de Cargar y Registro O de Unidad R, cada de una de las cuales se describe a continuación.

Posteriormente, T0 realiza una operación de drenaje para todos los hilos (DRAIN ALLTIDS), ETAPA 606, que retiene la expedición de instrucción para T0 hasta que se drenan o evacuan todas las instrucciones en T1 de la segmentación, y consulta el estado al T1. En un ejemplo, se usa una instrucción de drenaje para realizar la operación de drenaje, un ejemplo de la cual se describe a continuación.

Mientras se está drenando la segmentación de las instrucciones en los otros hilos, CONSULTA 608, el procesamiento continúa con la ETAPA 606. Sin embargo, en respuesta al drenaje de las instrucciones en T1, T0 continúa la expedición y ejecución de instrucción para T0, ETAPA 610.

Posteriormente, T0 comprueba de nuevo si se prohíbe que se detenga T1 (y otros hilos, si hubiera), para garantizar que T1 no cambió su estado después de probarse, pero antes de ser detenido, ETAPA 612. Esta comprobación se realiza, como se ha descrito anteriormente, usando STPIFALW. Si ahora se prohíbe que se detenga T1, CONSULTA 614, entonces T0 permite que T1 continúe ejecutándose, apagando el bit 9 de MCR002 (es decir, estableciéndolo en

cero), ETAPA 616. El procesamiento continúa a la ETAPA 600.

De otra manera, si no se está prohibiendo que se detenga T1, CONSULTA 614, entonces T0 realiza la secuencia de instrucciones (por ejemplo, una o más operaciones) que provocó la parada de T1, ETAPA 618. Después de que se finaliza esa secuencia de instrucciones, se permite que T1 continúe, ETAPA 620. Por lo tanto, T0 restablece el bit 9 en MCR002 usando, por ejemplo, una instrucción de Cargar y Y de Unidad R (LNRU) o CSGRU, como se describe a continuación. Posteriormente, ambos hilos se ejecutan normalmente, ETAPA 622.

Como se ha descrito anteriormente, se usan un número de instrucciones para controlar la ejecución de uno o más hilos de un procesador de múltiples hilos. Cada una de estas instrucciones se describe a continuación.

Haciendo referencia a las Figuras 7A-7B, se describe una realización de una instrucción de drenaje. En particular, la Figura 7A representa una realización de un formato de la instrucción de drenaje, y la Figura 7B representa una realización de la lógica asociada con la instrucción de drenaje.

Con referencia a la Figura 7A, una instrucción de drenaje 700 incluye un campo de opcode 702 que incluye un código de operación que identifica una operación de drenaje; un campo de máscara (M3) 704 que incluye un valor que indica un recuento de paralización, que especifica cuántos ciclos está paralizado el procesamiento; y un campo de instrucción 706 (12) que indica el tipo de drenaje, que, en este ejemplo, es un drenaje de todos los TIDS (ID de hilo) que especifica que todos los hilos tienen que drenarse.

En la operación y con referencia a la Figura 7B, el hilo T0 para el procesamiento de instrucción para T0, en la etapa de decodificación o expedición de instrucción de la segmentación hasta que se cumplen las condiciones especificadas, ETAPA 750. Bits especificados del campo 12 de la instrucción (por ejemplo, bits 0:31 de 12, que son, por ejemplo, bits 16:47 del campo de texto de instrucción (I-text), que incluye todos los campos de la instrucción) especifican qué una o más condiciones de hardware tienen que cumplirse antes de continuar el procesamiento de instrucción. En una realización, las condiciones especificadas incluyen un control de hilos cruzados (por ejemplo, bit 0 del campo 12; bit 16 del campo de texto de instrucción), que comprueba el estado de T1 (u otros hilos) para determinar si se ha parado el procesamiento en T1. Cuando bit 0 del campo 12 es '1'b, especifica que todas las otras condiciones de drenaje tienen que cumplirse en ambos hilos para continuar el procesamiento en este hilo (el otro hilo o hilos no están bloqueados por un DRENAJE en este hilo). Cuando se usa esta función, debe tenerse cuidado para evitar congelamientos.

En una o más realizaciones, pueden especificarse otras condiciones en el campo 12. Un uno en una posición de bit indicada indica que esa condición debe cumplirse antes de reanudar el procesamiento de instrucción; si hay más de un bit activado, tienen que cumplirse todas las condiciones seleccionadas. En implementación, en una realización, cuando bit 16 de texto de instrucción (es decir, bit 0 del campo 12) es 1, se realizan la lógica O de ambas (o todas) las funciones de estado de los hilos de hardware, en una base de bit a bit, antes de combinar por lógica O juntas todas las funciones que se seleccionan para determinar el valor final de si se satisfacen las condiciones de DRENAJE.

Se realiza una determinación en cuanto a si se han cumplido las condiciones especificadas, CONSULTA 752. Si no, entonces continúa la parada, ETAPA 750. De otra manera, si se han cumplido las condiciones, el procesamiento se paraliza un número adicional de ciclos, ETAPA 754. Este número adicional puede ser cero o más, y se especifica en el campo M3 de la instrucción de drenaje. Por ejemplo, el campo M3 especifica un número adicional de ciclos entre 0 y 15, como ejemplos, a paralizar después de que se satisfacen las condiciones especificadas en el campo 12. Posterior a la paralización del número adicional de ciclos, se reanuda procesamiento de ciclos, ETAPA 756.

En una realización, si una instrucción anterior y el drenaje se están expidiendo simultáneamente, se permite que la instrucción anterior finalice la expedición y continúe a través de la segmentación normalmente, pero la instrucción de drenaje y todas las instrucciones posteriores se bloquearán en la expedición hasta que se satisfacen las condiciones. Obsérvese que la instrucción de drenaje únicamente opera en este hilo en procesamiento de retardo. Para detener otro hilo, se usa la técnica descrita en este documento. Sin embargo, un bit especificado (por ejemplo, bit 0 de 12), cuando es 1, indica que tienen que cumplirse todas las condiciones especificadas en todos los hilos para continuar el procesamiento después de la instrucción de drenaje en este hilo.

Como se indica, el campo M3 de la instrucción especifica el número de ciclos adicionales a paralizar en la segmentación. Esto puede usarse en conjunto con cualquiera de las condiciones permitidas en el campo 12. También puede especificarse con el campo 12 todo ceros que proporciona un retado de recuento de ciclos inmediato en la expedición. Existe una paralización de un ciclo en la expedición de la instrucción de drenaje incluso cuando el campo M3 es cero. Por lo tanto, este recuento especifica el número de ciclos a retardar más un ciclo. El hardware puede emitir el drenaje junto con otras instrucciones y puede emitirse fuera de orden ya que afecta únicamente a las etapas de extremo frontal de la segmentación.

Esta instrucción se concibe para su uso cuando los enclavamientos necesarios para garantizar una operación correcta no están incorporados en el hardware. En la mayoría de los casos, el hardware cubre automáticamente ventanas a partir de instrucciones anteriores en la segmentación.

El código de condición se cambia por esta instrucción.

Otra instrucción usada es la instrucción de Comparar y Conmutar Registro de Unidad R, que se describe con referencia a las Figuras 8A-8B. En particular, la Figura 8A representa una realización de un formato de la instrucción de Comparar y Conmutar Registro de Unidad R, y la Figura 8B representa una realización de la lógica asociada con la instrucción de Comparar y Conmutar Registro de Unidad R. Se ha de observar que la unidad R en las instrucciones analizadas en este documento se refiere a una unidad particular dentro del núcleo que realiza la instrucción. Sin embargo, no es necesario el uso de una unidad particular. Puede realizarse por otras unidades o simplemente por el núcleo.

Con referencia a la Figura 8A, una instrucción de CSGRU 800 incluye al menos un campo de opcode 802a, 802b que incluye un código de operación que especifica una operación de comparar y conmutar registro; un primer campo 804 de registro (R1); un segundo campo 806 de registro (R3); y un campo 808 de instrucción (12), cada uno de los cuales se describe a continuación.

En la operación y con referencia a la Figura 8B, los contenidos del registro de unidad R (denominado en este documento como MCR) especificados por el número de registro absoluto de 10 bits indicado en bits seleccionados (por ejemplo, bits 22:31 del texto de instrucción (por ejemplo, bits 6:15 del campo 12 (808)) se comparan con los contenidos de un registro general (GR) especificado en R1, ETAPA 850. Si son iguales, CONSULTA 852, entonces los contenidos de MCR se escriben en el registro general especificado en R1, ETAPA 854, y los contenidos del registro general especificado en R3 se escriben en MCR, ETAPA 856. Adicionalmente, el código de condición se establece a cero, ETAPA 858, y se finaliza el procesamiento de CSGRU.

Volviendo a la CONSULTA 852, si los contenidos de MCR y el registro especificado en R1 no son iguales, entonces los contenidos de MCR se escriben en el registro especificado en R1, ETAPA 860, y el código de condición se establece a uno, ETAPA 858. Esto finaliza el procesamiento de CSGRU.

La función de leer-comparar-sustituir de CSGRU es una operación atómica según se observa por este hilo, T0, y los otros hilos de este procesador (por ejemplo, T1). En una realización, CSGRU se ejecuta con la opción LENTO activada para evitar congelamientos de hilos cruzados. La opción LENTO se indica estableciendo un bit seleccionado (por ejemplo, bit 17) de 12 (808) a uno, y se usa para solicitar modo lento, que significa que hay únicamente una instrucción en toda la segmentación en cada vez. Además, se realiza enclavamiento con esta instrucción, como se describe a continuación y, por lo tanto, un bit seleccionado (por ejemplo, bit 16) de 12 (808), denominado en este documento como ILOCK, se establece a uno.

En una realización, esta instrucción se rechaza y emite de nuevo, si otra instrucción seleccionada, tal como RSR (Leer Registro Especial), WSR (Escribir Registro Especial), NSR (Registro Especial Y), OSR (Registro Especial O), XSR (Registro Exclusivo o Especial), TRBIT (Probar Bit de Registro), RASR (Leer Registro Especial Absoluto), WASR (Escribir Registro Especial Absoluto), TARBIT (Probar Bit de Registro Absoluto), NASR (Registro Especial Absoluto Y), OASR (Registro Especial Absoluto O), XASR (Registro Especial Absoluto O Exclusivo), LORU (Cargar y Registro O de Unidad R), LNRU (Cargar y Registro Y de Unidad R) o CSGRU (Comparar y Conmutar Registro de Unidad R), está en la segmentación para este hilo (T0) o cualquier otro hilo y el bit ILOCK (por ejemplo, 16 de 12) está en la otra instrucción. Esta instrucción se emite, por ejemplo, únicamente después de que se han emitido todas las instrucciones anteriores desde este hilo y también fuerza que todas las instrucciones futuras de este hilo dependan de esta.

Los ajustes de código de condición incluyen, por ejemplo: comparación de CC0 igual, registro de unidad R sustituido por GR R1; comparación de CC1 distinta, registro de unidad R no se cambia.

Otra instrucción usada es la instrucción de Comparar y Conmutar Registro de Unidad R, que se describe con referencia a las Figuras 9A-9B. En particular, la Figura 9A representa una realización de un formato de la instrucción de Cargar y Registro O de Unidad R, y la Figura 9B representa una realización de la lógica asociada con la instrucción de Cargar y Registro O de Unidad R.

Con referencia a la Figura 9A, una instrucción de LORU 900 incluye al menos un campo de opcode 902a, 902b que incluye un código de operación que especifica una operación de carga y registro O; un primer campo 904 de registro (R1); un segundo campo 906 de registro (R3); y un campo 908 de instrucción (12), cada uno de los cuales se describe a continuación.

En la operación y con referencia a la Figura 9B, los contenidos del registro de unidad R (denominado en este documento como MCR) especificados por el número de registro absoluto de 10 bits indicado en bits seleccionados (por ejemplo, bits 22:31 del texto de instrucción (por ejemplo, bits 6:15) del campo 12 (908)) se cargan en el registro general especificado en R1, ETAPA 950. Además, los contenidos del registro general especificado en R3 se combinan por lógica O con los contenidos de MCR, ETAPA 952, y el resultado se escribe en MCR, ETAPA 954.

La función de leer-O-sustituir de LORU es una operación atómica según se observa por este hilo, T0, y los otros hilos de este procesador (por ejemplo, T1). En una realización, LORU se ejecuta con la opción LENTO activada para evitar

congelamientos de hilos cruzados. La opción LENTO se indica estableciendo un bit seleccionado (por ejemplo, bit 17) de 12 (908) a uno. Además, se realiza enclavamiento con esta instrucción, como se describe a continuación y, por lo tanto, un bit seleccionado (por ejemplo, bit 16) de 12 (908), denominado en este documento como ILOCK, se establece a uno.

En una realización, esta instrucción se rechaza y emite de nuevo, si otra instrucción seleccionada, tal como RSR (Leer Registro Especial), WSR (Escribir Registro Especial), NSR (Registro Especial Y), OSR (Registro Especial O), XSR (Registro Exclusivo o Especial), TRBIT (Probar Bit de Registro), RASR (Leer Registro Especial Absoluto), WASR (Escribir Registro Especial Absoluto), TARBIT (Probar Bit de Registro Absoluto), NASR (Registro Especial Absoluto Y), OASR (Registro Especial Absoluto O), XASR (Registro Especial Absoluto O Exclusivo), LORU (Cargar y Registro O de Unidad R), LNRU (Cargar y Registro Y de Unidad R) o CSGRU (Comparar y Conmutar Registro de Unidad R), está en la segmentación para este hilo (T0) o cualquier otro hilo y el bit ILOCK (bit 16 de 12) está en la otra instrucción. Esta instrucción se emite, por ejemplo, únicamente después de que se han emitido todas las instrucciones anteriores desde este hilo y también fuerza que todas las instrucciones futuras de este hilo dependan de esta.

El código de condición no se cambia.

Otra instrucción usada es la instrucción de Cargar y Registro Y de Unidad R (LNRU), que se describe con referencia a las Figuras 10A-10B. En particular, la Figura 10A representa una realización de un formato de la instrucción de Cargar y Registro Y de Unidad R, y la Figura 10B representa una realización de la lógica asociada con la instrucción de Cargar y Registro Y de Unidad R.

Con referencia a la Figura 10A, una instrucción de LNRU 1000 incluye al menos un campo de opcode 1002a, 1002b que incluye un código de operación que especifica una operación de cagar y registro Y; un primer campo 1004 de registro (R1); un segundo campo 1006 de registro (R3); y un campo 1008 de instrucción (12), cada uno de los cuales se describe a continuación.

En la operación y con referencia a la Figura 10B, los contenidos del registro de unidad R (denominado en este documento como MCR) especificados por su número de registro absoluto de 10 bits indicado en bits seleccionados (por ejemplo, bits 22:31 del texto de instrucción (por ejemplo, bits 6:15 del campo 12 (1008)) se cargan en el registro general especificado en R1, ETAPA 1050. Además, los contenidos del registro general especificado en R3 se combinan por lógica Y con los contenidos de MCR, ETAPA 1052, y el resultado se escribe en MCR, ETAPA 1054.

La función de leer-Y-sustituir de LNRU es una operación atómica según se observa por este hilo, T0, y los otros hilos de este procesador (por ejemplo, T1). En una realización, LNRU se ejecuta con la opción LENTO activada para evitar congelamientos de hilos cruzados. La opción LENTO se indica estableciendo un bit seleccionado (por ejemplo, bit 17) de 12 (1008) a uno. Además, se realiza enclavamiento con esta instrucción, como se describe a continuación y, por lo tanto, un bit seleccionado (por ejemplo, bit 16) de 12 1008, denominado en este documento como ILOCK, se establece a uno.

En una realización, esta instrucción se rechaza y emite de nuevo, si otra instrucción seleccionada, tal como RSR (Leer Registro Especial), WSR (Escribir Registro Especial), NSR (Registro Especial Y), OSR (Registro Especial O), XSR (Registro Exclusivo o Especial), TRBIT (Probar Bit de Registro), RASR (Leer Registro Especial Absoluto), WASR (Escribir Registro Especial Absoluto), TARBIT (Probar Bit de Registro Absoluto), NASR (Registro Especial Absoluto Y), OASR (Registro Especial Absoluto O), XASR (Registro Especial Absoluto O Exclusivo), LORU (Cargar y Registro O de Unidad R), LNRU (Cargar y Registro Y de Unidad R) o CSGRU (Comparar y Conmutar Registro de Unidad R), está en la segmentación para este hilo (T0) o cualquier otro hilo y el bit ILOCK (por ejemplo, 16 de 12) está en la otra instrucción. Esta instrucción se emite, por ejemplo, únicamente después de que se han emitido todas las instrucciones anteriores desde este hilo y también fuerza que todas las instrucciones futuras de este hilo dependan de esta.

El código de condición no se cambia.

LNRU, así como LORU y CSGRU, usa registros que son accesibles para todos los hilos en el núcleo de SMT, en lugar de almacenamiento como un medio de comunicación compartida. Estos registros son, por ejemplo, registros de hardware separados de la memoria o almacenamiento del procesador. Por ejemplo, en un diseño de un núcleo, hay aproximadamente 64 registros que se comparten (son comunes) con todos los hilos en el núcleo; los hilos pueden leer y escribir libremente estos registros compartidos. En algunos casos de registros de control, si ambos hilos intentaran escribir los mismos sin enclavamientos especiales, podría perderse una actualización por uno de los hilos. En otros casos, únicamente se permite que uno de los hilos "posea" un recurso controlado por bits en el registro. Por lo tanto, estas instrucciones atómicas que operan en registros compartidos se usan para controlar y ordenar acceso a estos registros compartidos.

LNRU, LORU y CSGRU permiten cada uno una operación atómica entre registros generales y MCR a través de hilos usando enclavamiento para controlar operaciones y ejecución entre hilos. Como se indica, cada una de las instrucciones tiene un bit ILOCK, y cuando ese bit está activado por una instrucción ejecutándose en la segmentación, si una segunda instrucción entra en la segmentación con su bit ILOCK también establecido, la segunda instrucción se

rechaza (y ejecuta de nuevo posteriormente cuando la primera instrucción finaliza). Esto garantiza la atomicidad con acceso a estos registros entre hilos.

Existen, por ejemplo, dos tipos de instrucciones de enclavamiento: una instrucción de una sola micro operación *m*op, tal como LNRU y LORU; y una instrucción de dos *m*op, tal como CSGRU. Con la instrucción de una sola *m*op, el enclavamiento se establece en la emisión de *m*op (instrucción de tipo RSR y WSR) y elimina en la finalización de *m*op para un tipo RSR y en punto de comprobación para un tipo WSR. En una instrucción de dos *m*op, el enclavamiento se establece en la emisión de primera *m*op (tipo RSR) y se elimina en punto de comprobación de la segunda *m*op (tipo WSR).

Detalles adicionales con respecto al uso de enclavamiento y enclavamiento se describen con referencia a las Figuras 11A-11B. Esta lógica se realiza por el núcleo y, en particular, por un segmento en el que se emite la instrucción.

Haciendo referencia de manera inicial a la Figura 11A, se obtiene una instrucción a ejecutarse (por ejemplo, LNRU, LORU, CSGRU) por un procesador multihilo, ETAPA 1100. La ejecución de la instrucción se inicia por el procesador multihilo para realizar una operación, ETAPA 1102. La operación incluye múltiples suboperaciones a realizarse atómicamente. Se realiza una determinación en cuanto a si la instrucción tiene que continuar ejecutándose, CONSULTA 1104. La determinación usa, por ejemplo, enclavamiento para determinar si la instrucción tiene acceso atómico a uno o más registros compartidos por el hilo y uno o más hilos distintos.

Si la instrucción tiene que continuar ejecutándose, la ejecución continúa, que incluye realizar la operación usando al menos un registro compartido, ETAPA 1106. De otra manera, si la instrucción no tiene que continuar, se rechaza, ETAPA 1108.

Detalles adicionales relacionados con el enclavamiento se describen con referencia a la Figura 11B. Inicialmente, cuando una instrucción entra en la unidad R, en un ejemplo, se hace una comprobación en cuanto a si se establece un indicador de bloqueo, tal como el bit ILOCK (por ejemplo, bit 32 de texto de instrucción - también conocido como, bit 16 de 12) de la instrucción entrante (por ejemplo, establece a 1), CONSULTA 1150. Si el bit ILOCK no se establece, entonces procesamiento de enclavamiento se finaliza; sin embargo, si se establece el bit ILOCK en la instrucción entrante, entonces se hace una determinación adicional en cuanto a si se establece un bloqueo, referido como un enclavamiento, CONSULTA 1152. El enclavamiento se sitúa en un registro de hardware accesible a múltiples hilos.

Si se establece el enclavamiento (por ejemplo, un bit se establece a uno) indicando que otra instrucción está procesando que tiene su bit ILOCK establecido, entonces se rechaza la instrucción entrante, ETAPA 1153.

Sin embargo, si el enclavamiento no se establece, entonces se establece, ETAPA 1154, y procesamiento de la instrucción continúa (por ejemplo, en el segmento), ETAPA 1156. Cuando la instrucción finaliza (o tiene punto de comprobación), el enclavamiento se reestablece (por ejemplo, establece a cero), ETAPA 1158.

Detalles adicionales con respecto a enclavamiento incluyen:

(A) Enclavamiento puede establecerse por el segmento 0 cuando, por ejemplo:

- existe una instrucción en el segmento 0 que necesita establecer el enclavamiento y se emite solo
- existe una instrucción en el segmento 0 que necesita establecer el enclavamiento y existe otra instrucción en el segmento 1 que no quiere establecer el bloqueo - ambas instrucciones desde el mismo hilo.
- existe una instrucción en el segmento 0 que necesita establecer el enclavamiento y existe otra instrucción en el segmento 1 que necesita establecer el bloqueo, pero la instrucción en el segmento 0 es más antigua - ambas instrucciones desde el mismo hilo.
- existe una instrucción en el segmento 0 que necesita establecer el enclavamiento y existe otra instrucción en el segmento 1 que no quiere establecer el bloqueo - ambas instrucciones desde diferentes hilos.
- existe una instrucción en el segmento 0 que necesita establecer el enclavamiento y existe otra instrucción en el segmento 1 que necesita establecer el bloqueo - ambas instrucciones desde diferentes hilos - y el LFSR (Registro de Desplazamiento de Realimentación Lineal) apunta a el segmento 0. El LFSR se usa para producir un número pseudoaleatorio y tomando el bit más significativo del número, se proporciona una selección pseudoaleatoria entre los dos segmentos (es decir, elige aleatoriamente que segmento establecería el enclavamiento).

En un ejemplo, el enclavamiento es un vector que tiene un bit para cada posible instrucción en un grupo de expediciones. Por ejemplo, en un ejemplo, puede haber hasta tres instrucciones en un grupo de expediciones y, por lo tanto, el enclavamiento incluye tres bits, uno para cada instrucción. Cuando se establece un bit, por ejemplo, a 1, esto indica que la instrucción asociada con ese bit tiene el enclavamiento. El enclavamiento también puede establecerse por el segmento 1, como se ha descrito anteriormente, sin embargo, el segmento 0 se sustituye con el segmento 1, y el segmento 1 con el segmento 0.

(B) Establecimiento del enclavamiento se realiza cuando, por ejemplo:

- hay una instrucción válida en el segmento Y
- el ILOCK se establece Y
- la predec_rd (es decir, una indicación temprana de una instrucción de tipo lectura (RSR)) o predec_wr (es decir, indicación temprana de una instrucción de tipo escritura (WSR)) se establece Y
- la instrucción en el segmento no se evacua/condiciona por x Y
- el enclavamiento puede establecerse por ese segmento (de acuerdo con (A)) Y
- el enclavamiento aún no se ha establecido

(C) Se actualiza enclavamiento cuando, por ejemplo:

- hay una instrucción válida en el segmento Y
- el ILOCK se establece Y
- la predec_rd o predec_wr se establece Y
- la instrucción en el segmento o se evacua/condiciona por x Y
- el enclavamiento ya se ha establecido Y
- la instrucción. GTAG (el identificador de un grupo de expediciones que incluye la instrucción) = enclavamiento.GTAG (es decir, es el identificador asociado con la instrucción = al identificador que establece el enclavamiento) Y
- la instrucción.th _id (id de hilo) = enclavamiento.th_id

En una realización, se realiza un restablecimiento de un enclavamiento en finalización de grupo si no hay micro operación (*m*op) de tipo escritura en el grupo que tomó el bloqueo. Si hay una *m*op de tipo escritura en el grupo pero no tomó el bloqueo, entonces el bloqueo se libera también en esa finalización (no tomó el bloqueo = bit ILOCK es 0 - esto es porqué el bit ILOCK para CSGRU también se establece en la parte WSR de forma que no se libera en la finalización de la RSR). Si la instrucción de tipo lectura también tomó el bloqueo, entonces el bloqueo se liberará únicamente en punto de comprobación. De esta manera se verá la atomicidad. Una excepción es para CSGRU en la que WSR está en el segundo grupo - por lo tanto la RSR del primer grupo establece el bloqueo y la WSR en el segundo grupo libera el bloqueo. En ese caso, el primer grupo vendrá antes que el segundo grupo (que tiene una GTAG que es mayor por 1 de la GTAG del primer grupo).
Un rechazo de una *m*op en el grupo podría no restablecer el bloqueo si esa *m*op no lo sostiene. El rechazo liberará el bloqueo únicamente, en un ejemplo, si no hay otras *m*op en el grupo que también sostienen el bloqueo.
Una revocación de una *m*op en el grupo podría no restablecer el bloqueo si esa *m*op no lo sostiene. La revocación liberará el bloqueo únicamente, en un ejemplo, si no hay otras *m*op en el grupo que también sostienen el bloqueo.
Cuando viene la condición x, se hace una comprobación en cuanto a si puede liberarse el enclavamiento. El problema es que la condición x debería liberar el bloqueo únicamente si la instrucción que tomó no se finalizó aún. Si la instrucción que tomó el bloqueo ya se completó, entonces la condición x no debería tener ningún efecto en el bloqueo (esto es cierto para la instrucción de tipo lectura que tomó el bloqueo ya que esa instrucción lo liberará en punto de comprobación. Para una instrucción de tipo lectura, la liberación ya se hizo en la finalización). Una excepción es CSGRU cuya parte de tipo lectura puede finalizarse ya, pero si hubiera una condición x antes de que el tipo escritura finalice, el bloque tiene que liberarse (si el tipo escritura finaliza entonces la condición x que vendrá posteriormente no debería tener ningún efecto en el enclavamiento).
Un restablecimiento en caso de la instrucción que estableció el bloqueo necesita evacuarse: el restablecimiento real se hará únicamente, por ejemplo, si el bloqueo no se sostiene más por ninguna instrucción de ese grupo. Por ejemplo, si la evacuación alcanza la primera *m*op en el grupo y esta *m*op sostiene el bloqueo, entonces el bloqueo se libera (por supuesto, las dos otras *m*op también pueden sostenerlo pero se evacuan). Si la evacuación viene de la segunda *m*op en el grupo y esta *m*op sostiene el bloqueo, entonces el bloqueo se libera únicamente, por ejemplo, si la primera *m*op tampoco lo mantiene (la tercera se evacuará de cualquier forma, por tanto, no ha necesidad de comprobarlo).

(D) Enclavamiento se restablece cuando, por ejemplo:

- Enclavamiento ya se ha establecido Y
- Sin actualizaciones desde (C) Y

(la instrucción que tomó el bloqueo está finalizando:

- la instrucción de lectura que lo bloqueo está finalizando Y

- este no es la finalización de primer grupo de CSGRU

O
(

La instrucción que tomó el bloqueo tiene punto de comprobación:

- la instrucción de escritura que lo bloqueó tiene punto de comprobación Y

- Enclavamiento.GTAG = Instrucción.GTAG

O

- si esta es la finalización de segundo grupo de la CSGRU, entonces espera a que tenga punto de comprobación Y

- Enclavamiento.GTAG+1 = Instrucción.GTAG

)
O
La *m*op que tomó el bloqueo se rechaza y no hay otros soportes en ese grupo
O
La *m*op que tomó el bloqueo se rescinde y no hay otros soportes en ese grupo
O
La *m*op que tomó el bloqueo se evacua/condiciona por x y no hay otros soportes en ese grupo
O
recuperación en marcha
)
(E) Rechazar cuando, por ejemplo:

1) el enclavamiento se bloquea Y
la instrucción.th_id != (no igual) enclavamiento.th_id Y
la instrucción.GTAG != enclavamiento.GTAG
Para un opcode de CSGRU, esto elimina el rechazo de la *m*op de WSR cuando el enclavamiento se bloqueó por la *m*op de RSR (tienen la misma GTAG y mismo ID de hilo).
También es cierto para grupos como (RSR, x, WSR) en los que el problema es en orden pero se rechaza la RSR por alguna razón y, por lo tanto, la WSR bloquea el bloqueo. En tal caso si el rechazo será sobre una base de id de instrucción individual, la RSE no sería capaz de entrar ya que el bloque se bloquea y todo el grupo no será capaz de finalizar ==> un punto muerto ya que la WRS no puede liberar el bloqueo. La solución es usar la GTAG de forma que la RSR será capaz de entrar y cuando finalice, la WSR también sería capaz de completar y liberaría el bloqueo.
2) mismo hilo en ambos segmentos Y
el ILOCK está activado en ambos segmentos Y
el segmento actual sostiene la instrucción más joven ==> debería rechazarse la instrucción más joven actual (también si el bit de enclavamiento no se activa aún por la instrucción más antigua).
En caso de que el enclavamiento está activado, debería rechazarse también la más antigua por la (1) condición (a no ser que esta es la WRS de la instrucción de CSGRU).
3) diferentes hilos en ambos segmentos y el ILOCK está activado en ambos segmentos y número de segmentos actual no es igual al valor de LFSR (que es 0 para el segmento 0 y 1 para el segmento 1) ==> la instrucción x de segmento actual debería rechazarse (también si el bit de enclavamiento aún no está activado por la instrucción más antigua).

En el caso de que el enclavamiento está activado, ambos deberían rechazarse por la (1) condición (a no ser que uno de ellos es la WSR de la instrucción de CSGRU).

En este documento se describe una realización de una técnica para que un hilo detenga la ejecución de uno o más hilos distintos de un procesador multihilo. La técnica se implementa para evitar congelamientos y para garantizar que todas las instrucciones asociadas con los otros hilos se finalizan antes de que se detengan. Esta técnica incluye, en un aspecto, una segmentación instrucción de drenaje que mira la información de estado de todos los hilos de hardware del procesador (o hilos seleccionados en otra realización) para ver si se satisfacen las condiciones antes de continuar la operación en este hilo.

Además, una realización de esta técnica usa instrucciones atómicas, tales como CSGRU, LORU y LNRU, para operar en registros compartidos. Por ejemplo, cuando dos o más hilos comparten un núcleo común, en un diseño de núcleo multihilo (por ejemplo, SMT), a menudo necesitan comunicar y compartir información; esto podría incluir semáforos, cerraduras, etc. Esto podría implicar firmware, milicódigo, o podría implicar software. Los hilos podrían usar instrucciones ISA existentes que se comunican a través de almacenamiento. Sin embargo, estas pueden ser lentas e implicar conflictos de almacenamiento-alcanzar-carga (store-hit-load) o carga-alcanzar-almacenamiento (load-hit-store) (comúnmente conocidos como Comparar Almacenamiento de Comandos (OSC)). Además, si la comunicación se hace por firmware, puede no ser deseable o ser imposible comunicarse a través de almacenamiento; una rutina de firmware podría estar en la mitad de una secuencia crítica en la que cargas de operandos y almacenamientos están prohibidas. Por lo tanto, estas instrucciones operan en registros, en lugar de almacenamiento.

Aunque las instrucciones atómicas se describen en relación con el control de ejecución de hilos, pueden usarse para otros fines. Cada instrucción se distingue del uso descrito en este documento y puede utilizarse en otras situaciones.

Haciendo referencia a la Figura 12, en un ejemplo, un producto de programa informático 1200 incluye, por ejemplo, uno o más medios de almacenamiento legible por ordenador no transitorios 1202 para almacenar medios de código de programa legible por ordenador, lógica y/o instrucciones 1204 en los mismos para proporcionar y facilitar una o más realizaciones.

La presente invención puede ser un sistema, un método y/o un producto de programa informático. El producto de programa informático puede incluir un medio de almacenamiento legible por ordenador (o medios) que tiene instrucciones de programa legibles por ordenador en el mismo para provocar que un procesador lleve a cabo aspectos de la presente invención.

El medio de almacenamiento legible por ordenador puede ser un dispositivo tangible que puede mantener y almacenar instrucciones para su uso por un dispositivo de ejecución de instrucciones. El medio de almacenamiento legible por ordenador puede ser, por ejemplo, pero sin limitación, un dispositivo de almacenamiento electrónico, un dispositivo de almacenamiento magnético, un dispositivo de almacenamiento óptico, un dispositivo de almacenamiento electromagnético, un dispositivo de almacenamiento de semiconductores, o cualquier combinación adecuada de lo anterior. Una lista no exhaustiva de ejemplos más específicos del medio de almacenamiento legible por ordenador incluye lo siguiente: un disquete de ordenador portátil, un disco duro, una memoria de acceso aleatorio (RAM), una memoria de solo lectura (ROM), una memoria de solo lectura programable borrable (EPROM o memoria Flash), una memoria de acceso aleatorio estática (SRAM), una memoria de solo lectura de disco compacto portátil (CD-ROM), un disco versátil digital (DVD), un lápiz de memoria, un disco flexible, un dispositivo codificado mecánicamente tal como tarjetas de perforación o estructuras elevadas en un surco que tienen instrucciones grabadas en las mismas, y cualquier combinación adecuada de lo anterior. Un medio de almacenamiento legible por ordenador, como se usa en el presente documento, no ha de interpretarse como que son señales transitorias *per se,* tales como ondas de radio u otras ondas electromagnéticas que se propagan libremente, ondas electromagnéticas que se propagan a través de una guía de onda u otro medio de transmisión (por ejemplo, pulsos de luz que pasan a través de un cable de fibra óptica), o señales eléctricas transmitidas a través de un alambre.

Las instrucciones de programa legibles por ordenador descritas en el presente documento pueden descargarse en respectivos dispositivos informáticos/de procesamiento desde un medio de almacenamiento legible por ordenador o a un ordenador externo o dispositivo de almacenamiento externo mediante una red, por ejemplo, Internet, una red de área local, una red de área extensa y/o una red inalámbrica. La red puede comprender cables de transmisión de cobre, fibras de transmisión óptica, transmisión inalámbrica, routers, cortafuegos, conmutadores, ordenadores de pasarela y/o servidores de borde. Una tarjeta adaptadora de red o interfaz de red en cada dispositivo informático/de procesamiento recibe instrucciones de programa legibles por ordenador de la red y reenvía las instrucciones de programa legibles por ordenador para su almacenamiento en un medio de almacenamiento legible por ordenador dentro del respectivo dispositivo informático/de procesamiento.

Las instrucciones de programa legibles por ordenador para llevar a cabo las operaciones de la presente invención pueden ser instrucciones de ensamblador, instrucciones de la arquitectura de conjunto de instrucciones (ISA), instrucciones de máquina, instrucciones dependientes de máquina, microcódigo, instrucciones de firmware, datos de ajuste de estado, o cualquier código fuente o código objeto escrito en cualquier combinación de uno o más lenguajes de programación, incluyendo un lenguaje de programación orientado a objetos tal como Smalltalk, C++ o similares, y lenguajes de programación procedurales convencionales, tales como el lenguaje de programación "C" o lenguajes de programación similares. Las instrucciones de programa legibles por ordenador pueden ejecutarse en su totalidad en el ordenador del usuario, parcialmente en el ordenador del usuario, como un paquete de software autónomo, parcialmente en el ordenador del usuario y parcialmente en un ordenador remoto o en su totalidad en el ordenador remoto o servidor. En el último escenario, el ordenador remoto puede conectarse al ordenador del usuario a través de cualquier tipo de red, incluyendo una red de área local (LAN) o una red de área extensa (WAN), o la conexión puede hacerse a un ordenador externo (por ejemplo, a través de la Internet usando un Proveedor de Servicio de Internet). En algunas realizaciones, la circuitería electrónica que incluye, por ejemplo, circuitería de lógica programable, campos de matrices de puertas programables (FPGA) o matrices de lógica programable (PLA), puede ejecutar las instrucciones de programa legibles por ordenador utilizando información de estado de las instrucciones de programa legibles por ordenador para personalizar la circuitería electrónica, para realizar aspectos de la presente invención.

Los aspectos de la presente invención se describen en el presente documento con referencia a ilustraciones de diagrama de flujo y/o diagramas de bloques de los métodos, aparatos (sistemas) y productos de programa informático de acuerdo con las realizaciones de la invención. Se entenderá que cada bloque de las ilustraciones de diagrama de flujo y/o diagramas de bloques, y combinaciones de bloques en las ilustraciones de diagrama de flujo y/o diagramas de bloques, pueden implementarse por instrucciones de programa legibles por ordenador.

Estas instrucciones de programa legibles por ordenador pueden proporcionarse a un procesador de un ordenador de fin general, ordenador de fin especial u otro aparato de procesamiento de datos programable para producir una máquina, de manera que las instrucciones, que se ejecutan mediante el procesador del ordenador u otro aparato de procesamiento de datos programable, crean medios para implementar las funciones/actos especificados en el diagrama de flujo y/o bloque o bloques del diagrama de bloques. Estas instrucciones de programa legibles por

ordenador pueden almacenarse también en un medio de almacenamiento legible por ordenador que puede dirigir un ordenador, un aparato de procesamiento de datos programable, y/u otros dispositivos para funcionar de una manera particular, de manera que el medio de almacenamiento legible por ordenador que tiene instrucciones almacenadas en el mismo comprende un artículo de fabricación que incluye instrucciones que implementan aspectos de la función/acto especificado en el diagrama de flujo y/o bloque o bloques del diagrama de bloques.

Las instrucciones de programa legibles por ordenador pueden cargarse también en un ordenador, otro aparato de procesamiento de datos programable u otro dispositivo para provocar que se realice una serie de etapas operacionales en el ordenador, otro aparato programable u otro dispositivo para producir un proceso implementado por ordenador, de manera que las instrucciones que se ejecutan en el ordenador, otro aparato programable u otro dispositivo implementan las funciones/actos especificados en el diagrama de flujo y/o bloque o bloques de diagrama de bloques.

El diagrama de flujo y diagramas de bloques en las figuras ilustran la arquitectura, funcionalidad y operación de posibles implementaciones de sistemas, métodos y productos de programa informático de acuerdo con diversas realizaciones de la presente invención. En este sentido, cada bloque en el diagrama de flujo o diagramas de bloques puede representar un módulo, segmento, o porción de instrucciones, que comprende una o más instrucciones ejecutables para implementar la función o funciones lógicas especificadas. En algunas implementaciones alternativas, las funciones indicadas en el bloque pueden tener lugar fuera del orden indicado en las figuras. Por ejemplo, dos bloques mostrados en serie, de hecho, pueden ejecutarse sustancialmente de manera concurrente, o los bloques pueden ejecutarse en ocasiones en el orden inverso, dependiendo de la funcionalidad implicada. Se observará también que cada bloque de los diagramas de bloques y/o ilustración de diagrama de flujo y combinaciones de bloques en los diagramas de bloques y/o ilustración de diagrama de flujo, puede implementarse por sistemas basados en hardware de fin especial que realizan las funciones especificadas o actos o llevan a cabo combinaciones de hardware de fin especial e instrucciones informáticas.

Aunque se han descrito anteriormente diversas realizaciones, estas son únicamente ejemplos. Por ejemplo, pueden usarse entornos informáticos de otras arquitecturas para incorporar y usar una o más realizaciones. Además, uno o más aspectos de la invención son aplicables a formas de multihilo, distintas de SMT. Aún además, pueden usarse diferentes instrucciones, formatos de instrucciones, campos de instrucciones y/o valores de instrucciones. Son posibles muchas variaciones.

Además, pueden aprovecharse y usarse otros tipos de entornos informáticos. Como un ejemplo, puede usarse un sistema de procesamiento de datos adecuado para almacenar y/o ejecutar código de programa que incluye 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 ejecución real del código de programa, almacenamiento en bruto y memoria caché que proporciona almacenamiento temporal de al menos algún código de programa para reducir el número de veces que debe recuperarse el código del almacenamiento en bruto durante la ejecución.

Los dispositivos de entrada/salida o E/S (que incluyen, pero sin limitación, teclados, pantallas, dispositivos apuntadores, DASD, cinta, CD, DVD, unidades de memoria y otros medios de memoria, etc.) pueden acoplarse al sistema ya sea directamente o a través de controladores de E/S intermedios. Los adaptadores de red pueden acoplarse también al sistema para posibilitar 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 por cable y tarjetas de Ethernet son solamente unos pocos de los tipos disponibles de adaptadores de red.

La terminología usada en este documento es para el propósito de describir realizaciones particulares únicamente y no pretende ser limitante. Como se usa en este documento, las formas singulares "un", "una", "el" y "la" se conciben para incluir también las formas plurales, a no ser que el contexto indique claramente lo contrario. Se entenderá adicionalmente que los términos "comprende" y/o "comprendiendo/que comprende", cuando se usan en esta memoria descriptiva, especifican la presencia de características indicadas, elementos integrantes, etapas, operaciones, elementos y/o componentes, pero no excluyen la presencia o adición de una o más otras características, elementos integrantes, etapas, operaciones, elementos, componentes y/o grupos de los mismos.

Las correspondientes estructuras, materiales, actos y equivalentes de todos los significados o elementos de etapa más función en las reivindicaciones a continuación, si las hubiera, se pretende que incluyan cualquier estructura, material o acto para realizar la función en combinación con otros elementos reivindicados como se revindica específicamente. La descripción de una o más realizaciones se ha presentado para propósitos de ilustración y descripción, pero no se concibe para ser exhaustiva o limitarse en la forma divulgada. La realización se eligió y describió para explicar mejor diversos aspectos y la aplicación práctica, y para habilitar que otros expertos en la materia entiendan diversas realizaciones con diversas modificaciones ya que se adecuan al uso particular contemplado.

REIVINDICACIONES

1. Un método para controlar la ejecución de hilos en un entorno informático, comprendiendo dicho método: detener (604), mediante un hilo que se llevan a cabo en un procesador del entorno informático, la ejecución de otro hilo que se ejecuta dentro del procesador, usando la parada uno o más bits en uno o más registros compartidos del procesador, estando el uno o más registros compartidos por el hilo y el otro hilo, comprendiendo la parada:

determinar si el otro hilo está prohibiendo su detención comprobando (600) un bit seleccionado en un registro de control, en donde el registro de control comprende un registro de dicho uno o más registros compartidos del procesador y un bit seleccionado en un registro de dirección de instrucción, en donde el registro de dirección de instrucción comprende un registro de uno o más registros del procesador únicos del otro hilo;
detener, mediante el hilo, la búsqueda y ejecución de instrucción en el otro hilo, basándose en la etapa de determinación inmediatamente anterior que determina que el otro hilo no está prohibiendo su detención;
determinar que el otro hilo ha dejado de ejecutarse dentro del procesador; y
caracterizado por realizar (606), mediante el hilo, una operación de drenaje para todos los hilos en el entorno informático, en donde la operación de drenaje retiene la expedición de instrucción para el hilo hasta que todas las instrucciones del otro hilo se hayan drenado o evacuado, cuando la etapa de determinación inmediatamente anterior determina que el otro hilo ha dejado de ejecutarse dentro del procesador; realizar (618) mediante el hilo una o más operaciones dentro del procesador después de que la ejecución del otro hilo fue detenida dentro del procesador por el hilo; y
basándose en la finalización de la una o más operaciones, permitir que (620) el otro hilo continúe ejecutándose dentro del procesador.

2. El método de la reivindicación 1, en donde detener la ejecución del otro hilo además comprende:
determinar, basándose en la información de estado, si la ejecución del otro hilo fue detenida por el hilo, en donde la una o más operaciones se realizan basándose en la determinación que indica que el otro hilo ha dejado de ejecutarse dentro del procesador.

3. El método de la reivindicación 1, en donde la determinación de si el otro hilo está prohibiendo su detención comprende:

comprobar un primer indicador seleccionado en el registro de control y comprobar un segundo indicador seleccionado en el registro de dirección de instrucción, y
detener la búsqueda y ejecución de instrucción en el otro hilo, basándose en el primer indicador seleccionado y el segundo indicador seleccionado que indican que el otro hilo no está prohibiendo su detención.

4. El método de la reivindicación 1, en donde la instrucción de drenaje especifica una o más condiciones que hay que a satisfacer antes de la realización de la una o más operaciones, en donde las condiciones se relacionan con el estado del otro hilo, y en donde la realización de la una o más operaciones se ejecuta basándose en un resultado de la instrucción de drenaje que indica la satisfacción de la una o más condiciones.

5. El método de la reivindicación 3, que además comprende:

redeterminar, basándose en la comprobación que indica que la ejecución del otro hilo se ha detenido, si el otro hilo está prohibiendo su detención;
permitir la ejecución en el otro hilo, basándose en la redeterminación que indica que el otro hilo está prohibiendo su detención; y
realizar la una o más operaciones, basándose en la redeterminación que indica que el otro hilo no está prohibiendo su detención.

6. Un sistema que comprende medios adaptados para llevar a cabo todas las etapas del método de acuerdo con cualquier reivindicación de método anterior.

7. Un programa informático que comprende instrucciones para llevar a cabo todas las etapas del método de acuerdo con cualquier reivindicación de método anterior, cuando dicho programa informático se ejecuta en un sistema informático.

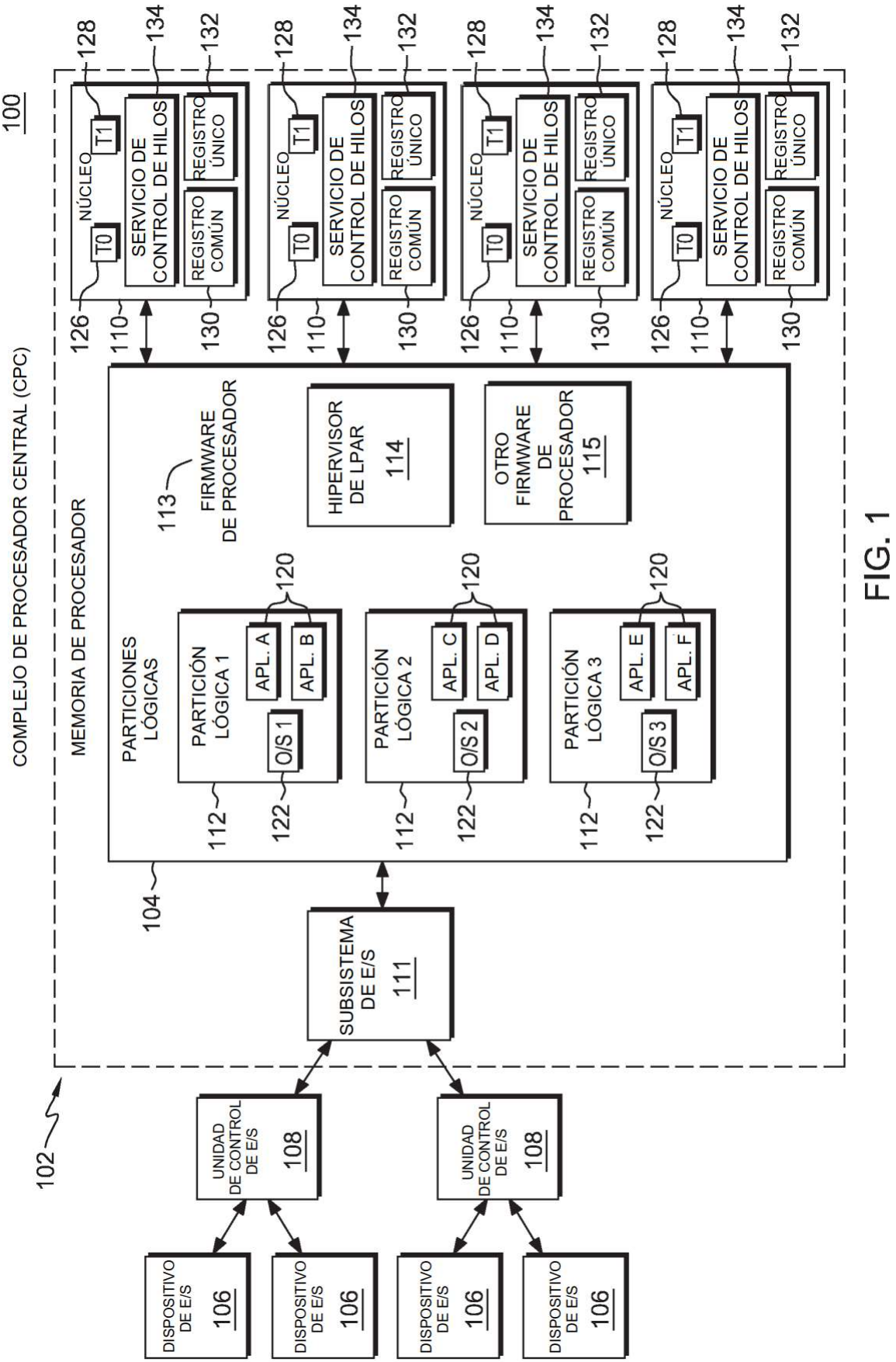
100
COMPLEJO DE PROCESADOR CENTRAL (CPC)
102
MEMORIA DE PROCESADOR
104
PARTICIONES LÓGICAS
PARTICIÓN LÓGICA 1
PARTICIÓN LÓGICA 2
PARTICIÓN LÓGICA 3
APL. A
APL. B
APL. C
APL. D
APL. E
APL. F
O/S 1
O/S 2
O/S 3
112
120
122
113
FIRMWARE DE PROCESADOR
HIPERVISOR DE LPAR 114
OTRO FIRMWARE DE PROCESADOR 115
SUBSISTEMA DE E/S 111
UNIDAD DE CONTROL DE E/S 108
DISPOSITIVO DE E/S 106
NÚCLEO
T0
T1
SERVICIO DE CONTROL DE HILOS
REGISTRO ÚNICO
REGISTRO COMÚN
126
128
130
132
134
110

FIG. 1

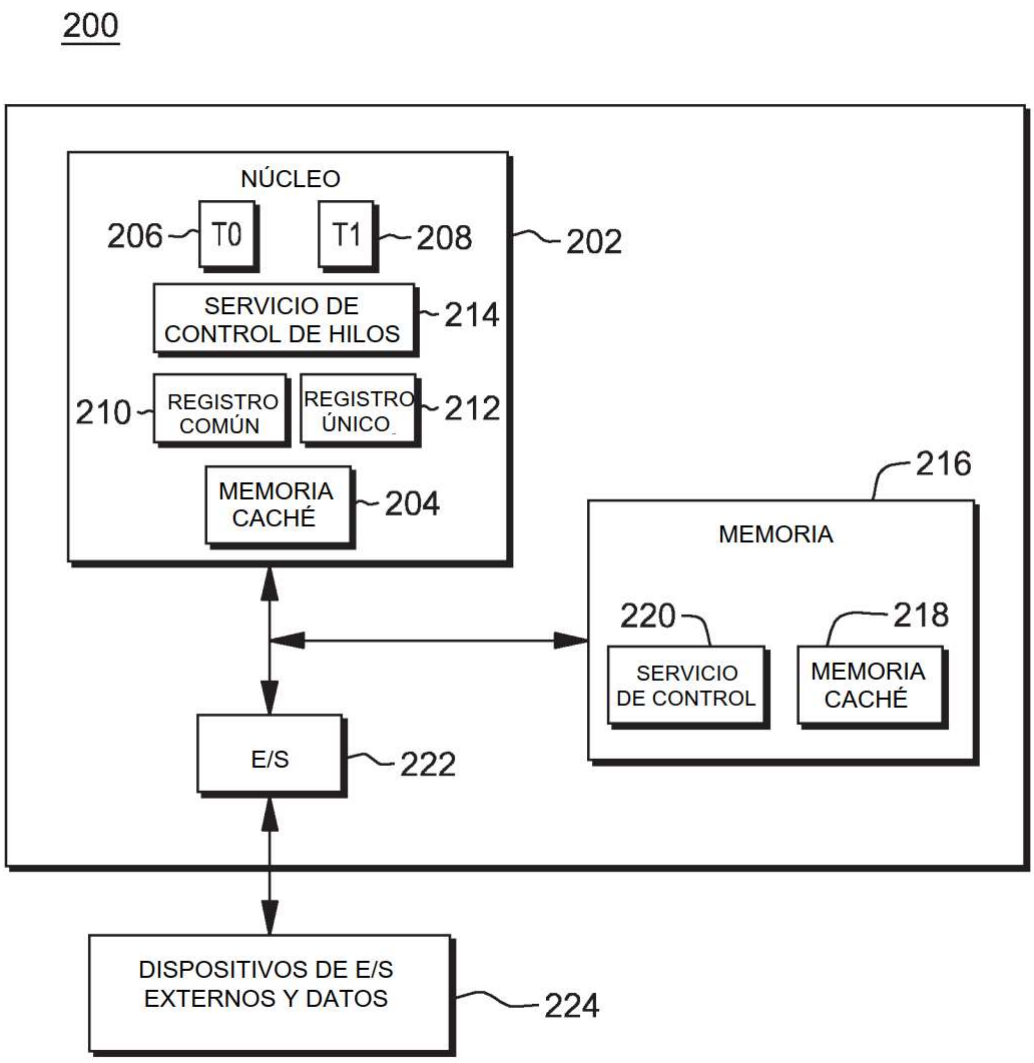
200
NÚCLEO
206
T0
T1
208
202
SERVICIO DE CONTROL DE HILOS
214
210
REGISTRO COMÚN
REGISTRO ÚNICO
212
MEMORIA CACHÉ
204
216
MEMORIA
220
218
SERVICIO DE CONTROL
MEMORIA CACHÉ
E/S
222
DISPOSITIVOS DE E/S EXTERNOS Y DATOS
224

FIG. 2

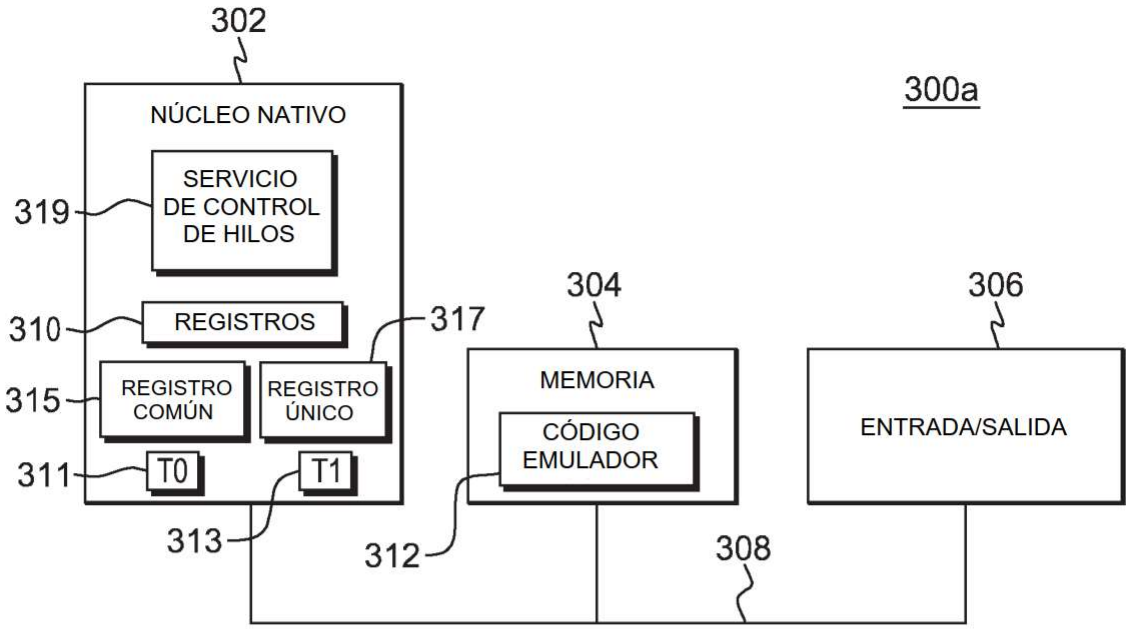
302
NÚCLEO NATIVO
300a
SERVICIO DE CONTROL DE HILOS
319
REGISTROS
310
317
REGISTRO COMÚN
315
REGISTRO ÚNICO
T0
311
T1
313
304
MEMORIA
CÓDIGO EMULADOR
312
306
ENTRADA/SALIDA
308

FIG. 3A

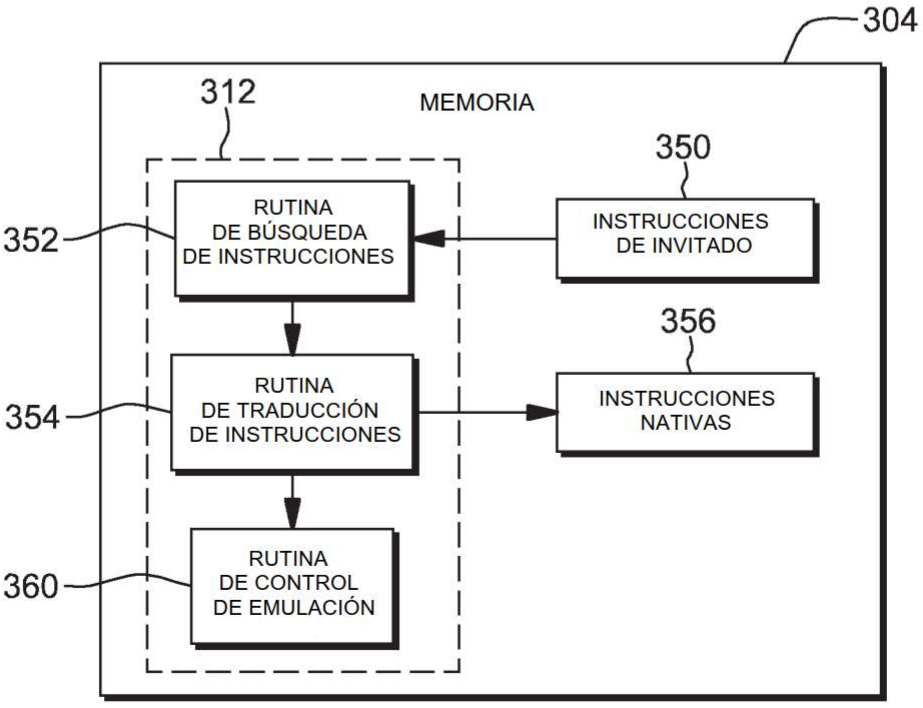
304
MEMORIA
312
350
RUTINA DE BÚSQUEDA DE INSTRUCCIONES
352
INSTRUCCIONES DE INVITADO
356
RUTINA DE TRADUCCIÓN DE INSTRUCCIONES
354
INSTRUCCIONES NATIVAS
RUTINA DE CONTROL DE EMULACIÓN
360

FIG. 3B

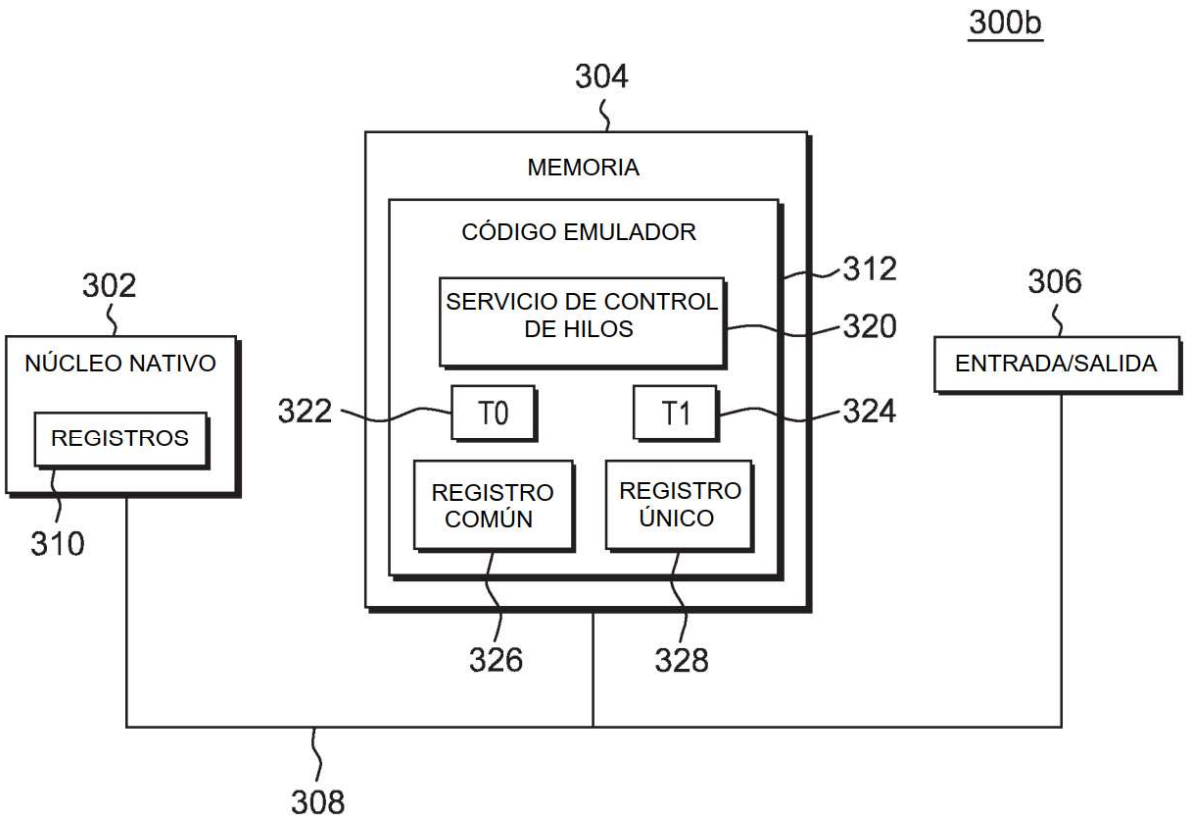
300b
304
MEMORIA
CÓDIGO EMULADOR
312
SERVICIO DE CONTROL DE HILOS
320
302
NÚCLEO NATIVO
REGISTROS
310
322
T0
T1
324
306
ENTRADA/SALIDA
REGISTRO COMÚN
REGISTRO ÚNICO
326
328
308

FIG. 3C

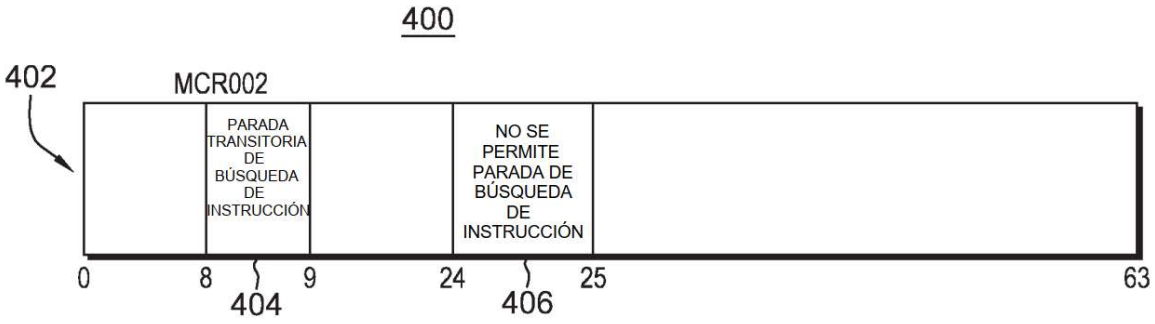

FIG. 4A

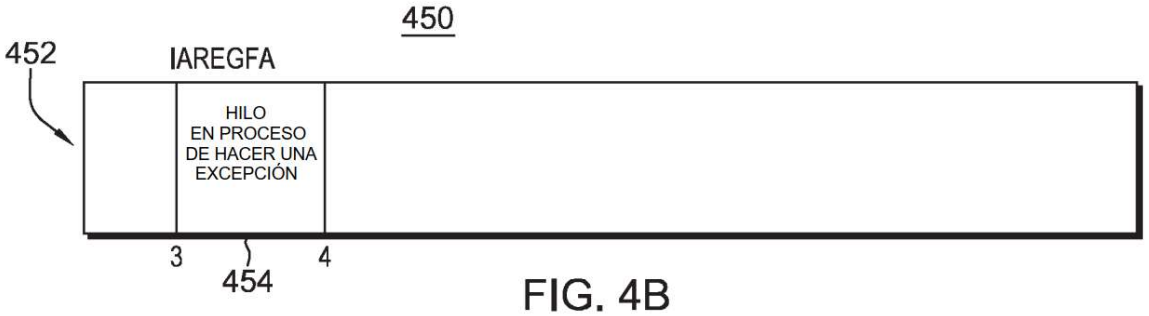

FIG. 4B

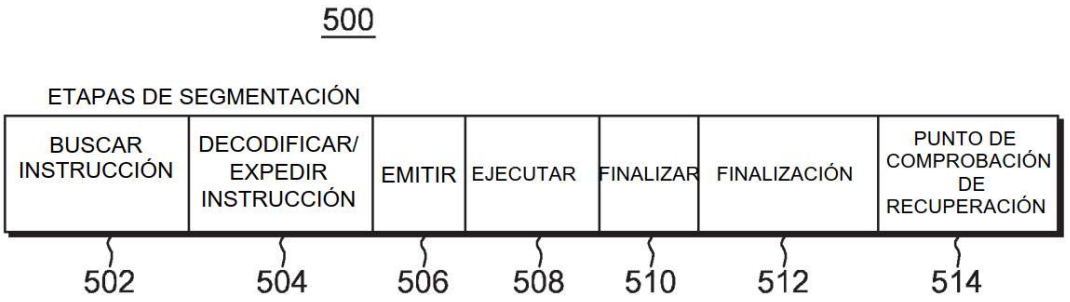

FIG. 5

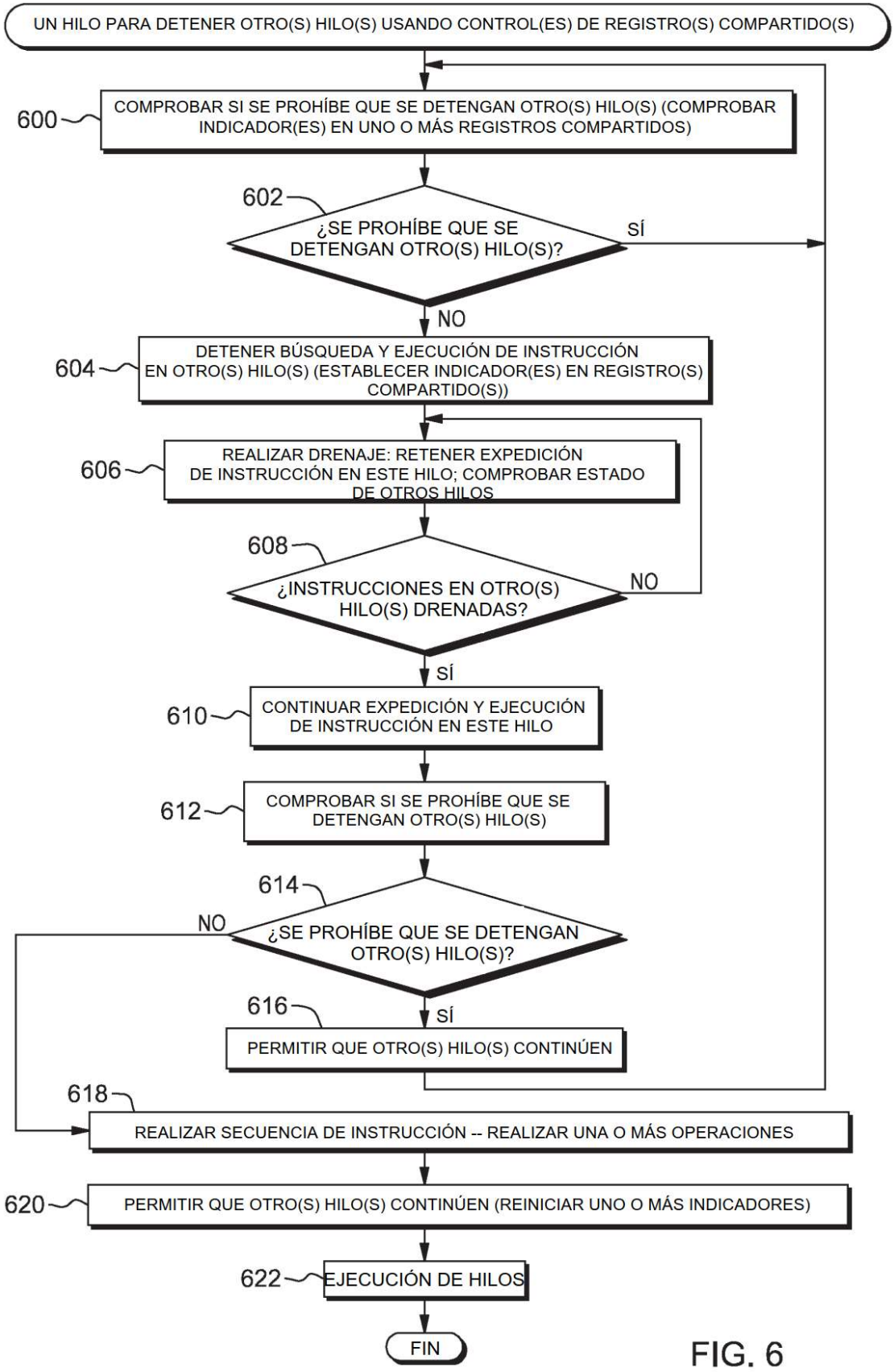
UN HILO PARA DETENER OTRO(S) HILO(S) USANDO CONTROL(ES) DE REGISTRO(S) COMPARTIDO(S)
600
COMPROBAR SI SE PROHÍBE QUE SE DETENGAN OTRO(S) HILO(S) (COMPROBAR INDICADOR(ES) EN UNO O MÁS REGISTROS COMPARTIDOS)
602
¿SE PROHÍBE QUE SE DETENGAN OTRO(S) HILO(S)?
SÍ
NO
604
DETENER BÚSQUEDA Y EJECUCIÓN DE INSTRUCCIÓN EN OTRO(S) HILO(S) (ESTABLECER INDICADOR(ES) EN REGISTRO(S) COMPARTIDO(S))
606
REALIZAR DRENAJE: RETENER EXPEDICIÓN DE INSTRUCCIÓN EN ESTE HILO; COMPROBAR ESTADO DE OTROS HILOS
608
¿INSTRUCCIONES EN OTRO(S) HILO(S) DRENADAS?
NO
SÍ
610
CONTINUAR EXPEDICIÓN Y EJECUCIÓN DE INSTRUCCIÓN EN ESTE HILO
612
COMPROBAR SI SE PROHÍBE QUE SE DETENGAN OTRO(S) HILO(S)
614
¿SE PROHÍBE QUE SE DETENGAN OTRO(S) HILO(S)?
NO
SÍ
616
PERMITIR QUE OTRO(S) HILO(S) CONTINÚEN
618
REALIZAR SECUENCIA DE INSTRUCCIÓN -- REALIZAR UNA O MÁS OPERACIONES
620
PERMITIR QUE OTRO(S) HILO(S) CONTINÚEN (REINICIAR UNO O MÁS INDICADORES)
622
EJECUCIÓN DE HILOS
FIN

FIG. 6


700
DRENAJE
OPCODE
M3
I2
702
704
706

FIG. 7A

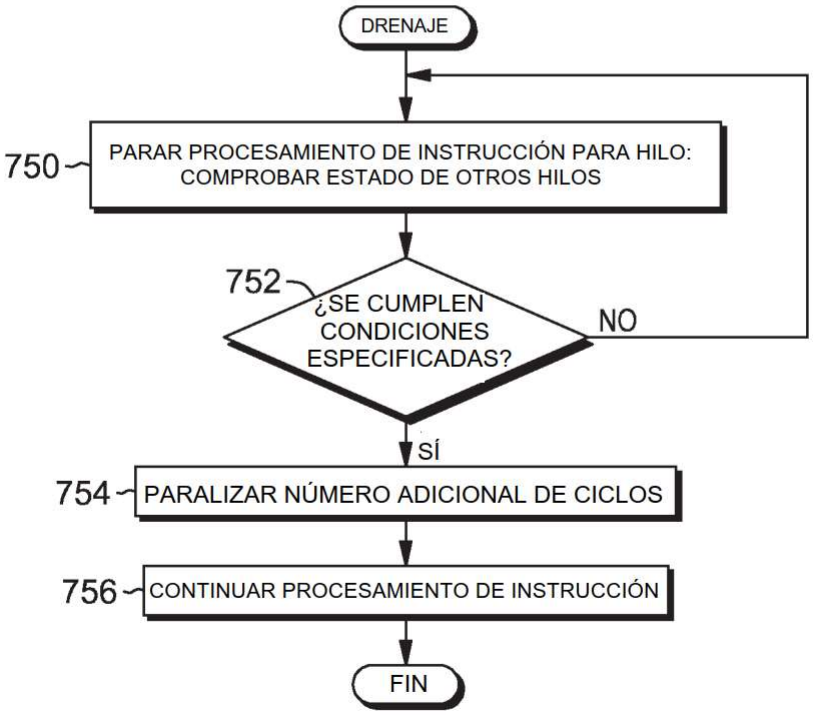
DRENAJE
750
PARAR PROCESAMIENTO DE INSTRUCCIÓN PARA HILO: COMPROBAR ESTADO DE OTROS HILOS
752
¿SE CUMPLEN CONDICIONES ESPECIFICADAS?
NO
SÍ
754
PARALIZAR NÚMERO ADICIONAL DE CICLOS
756
CONTINUAR PROCESAMIENTO DE INSTRUCCIÓN
FIN

FIG. 7B

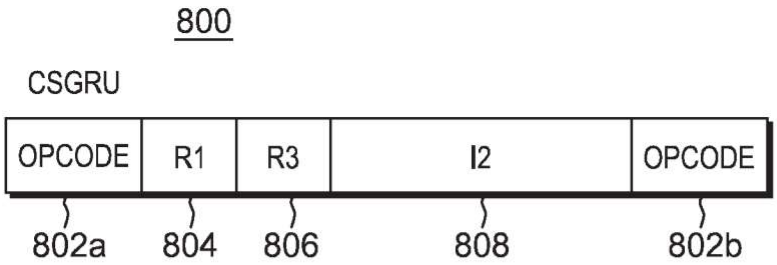
800
CSGRU
OPCODE
R1
R3
I2
OPCODE
802a
804
806
808
802b

FIG. 8A

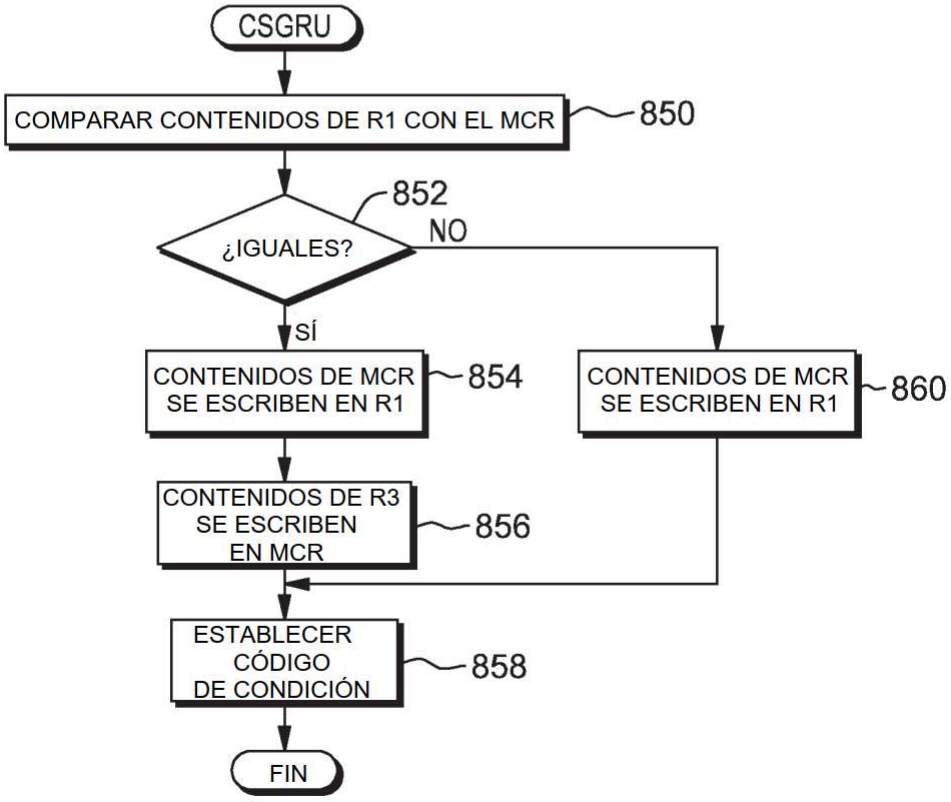
CSGRU
COMPARAR CONTENIDOS DE R1 CON EL MCR
850
852
¿IGUALES?
NO
SÍ
CONTENIDOS DE MCR SE ESCRIBEN EN R1
854
CONTENIDOS DE MCR SE ESCRIBEN EN R1
860
CONTENIDOS DE R3 SE ESCRIBEN EN MCR
856
ESTABLECER CÓDIGO DE CONDICIÓN
858
FIN

FIG. 8B

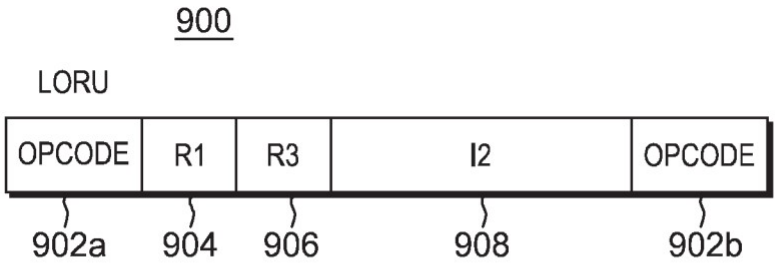
900
LORU
OPCODE
R1
R3
I2
OPCODE
902a
904
906
908
902b

FIG. 9A

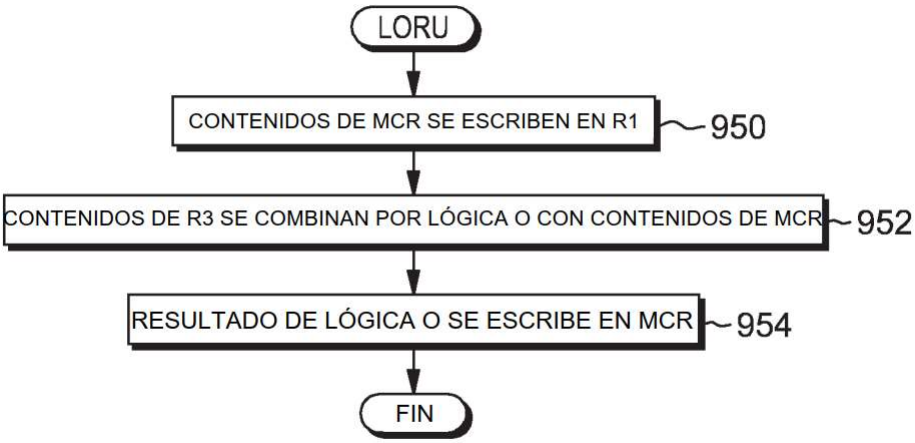
LORU
CONTENIDOS DE MCR SE ESCRIBEN EN R1
950
CONTENIDOS DE R3 SE COMBINAN POR LÓGICA O CON CONTENIDOS DE MCR
952
RESULTADO DE LÓGICA O SE ESCRIBE EN MCR
954
FIN

FIG. 9B

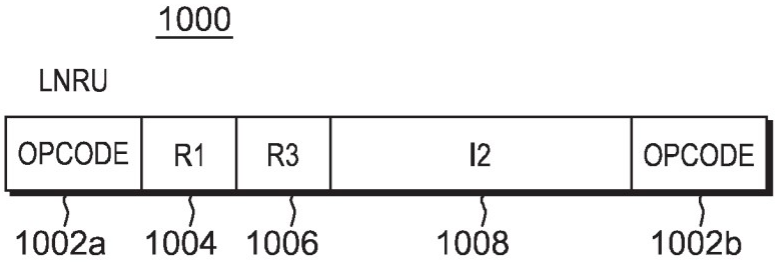

FIG. 10A

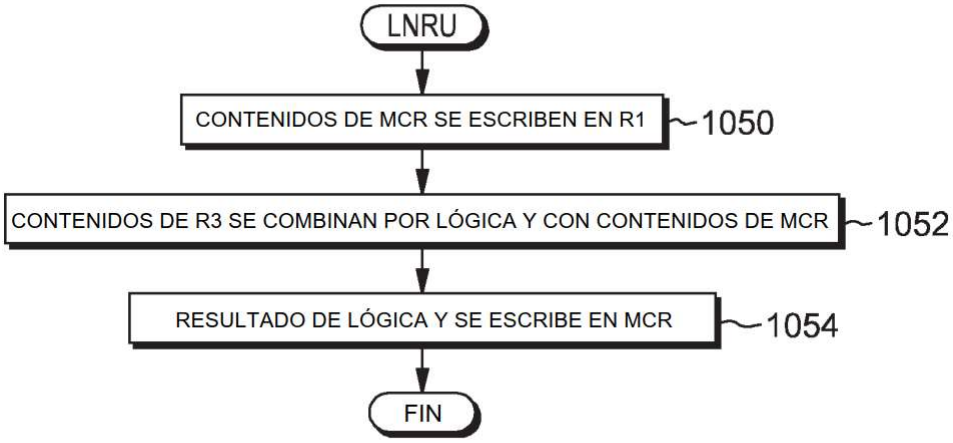

FIG. 10B

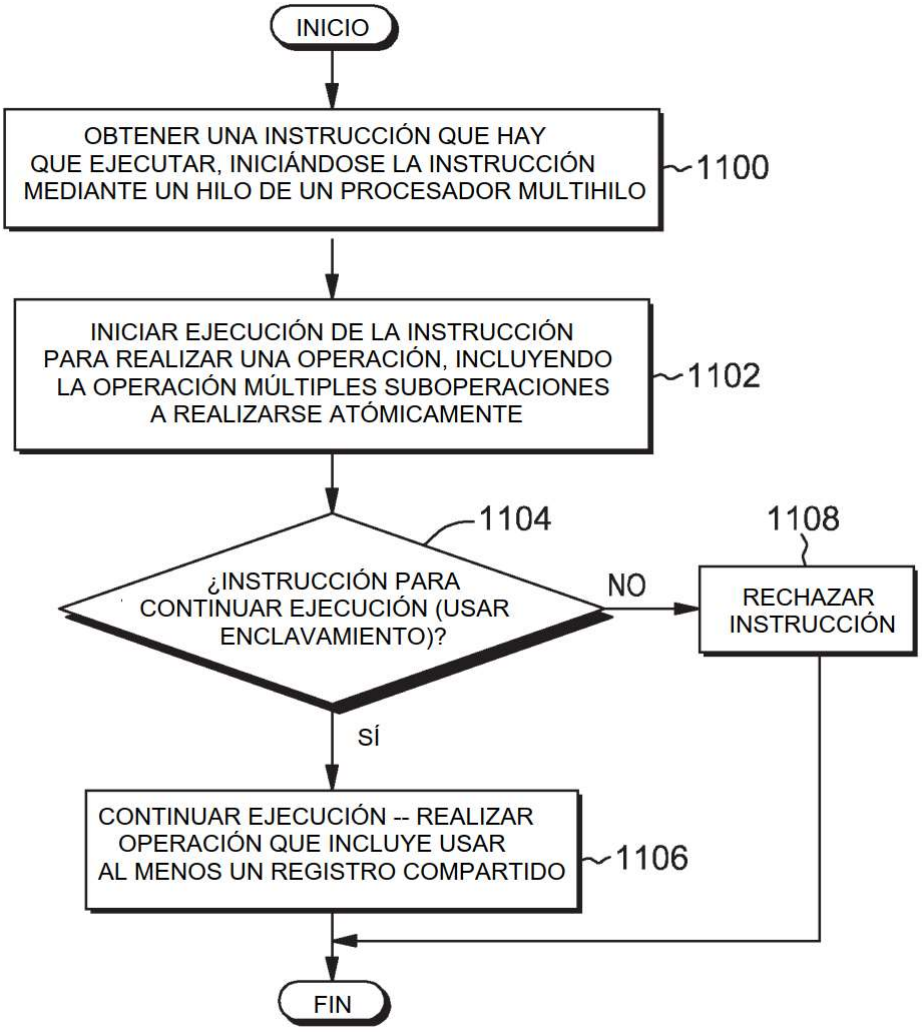
INICIO
OBTENER UNA INSTRUCCIÓN QUE HAY QUE EJECUTAR, INICIÁNDOSE LA INSTRUCCIÓN MEDIANTE UN HILO DE UN PROCESADOR MULTIHILO
1100
INICIAR EJECUCIÓN DE LA INSTRUCCIÓN PARA REALIZAR UNA OPERACIÓN, INCLUYENDO LA OPERACIÓN MÚLTIPLES SUBOPERACIONES A REALIZARSE ATÓMICAMENTE
1102
1104
¿INSTRUCCIÓN PARA CONTINUAR EJECUCIÓN (USAR ENCLAVAMIENTO)?
NO
1108
RECHAZAR INSTRUCCIÓN
SÍ
CONTINUAR EJECUCIÓN -- REALIZAR OPERACIÓN QUE INCLUYE USAR AL MENOS UN REGISTRO COMPARTIDO
1106
FIN

FIG. 11A

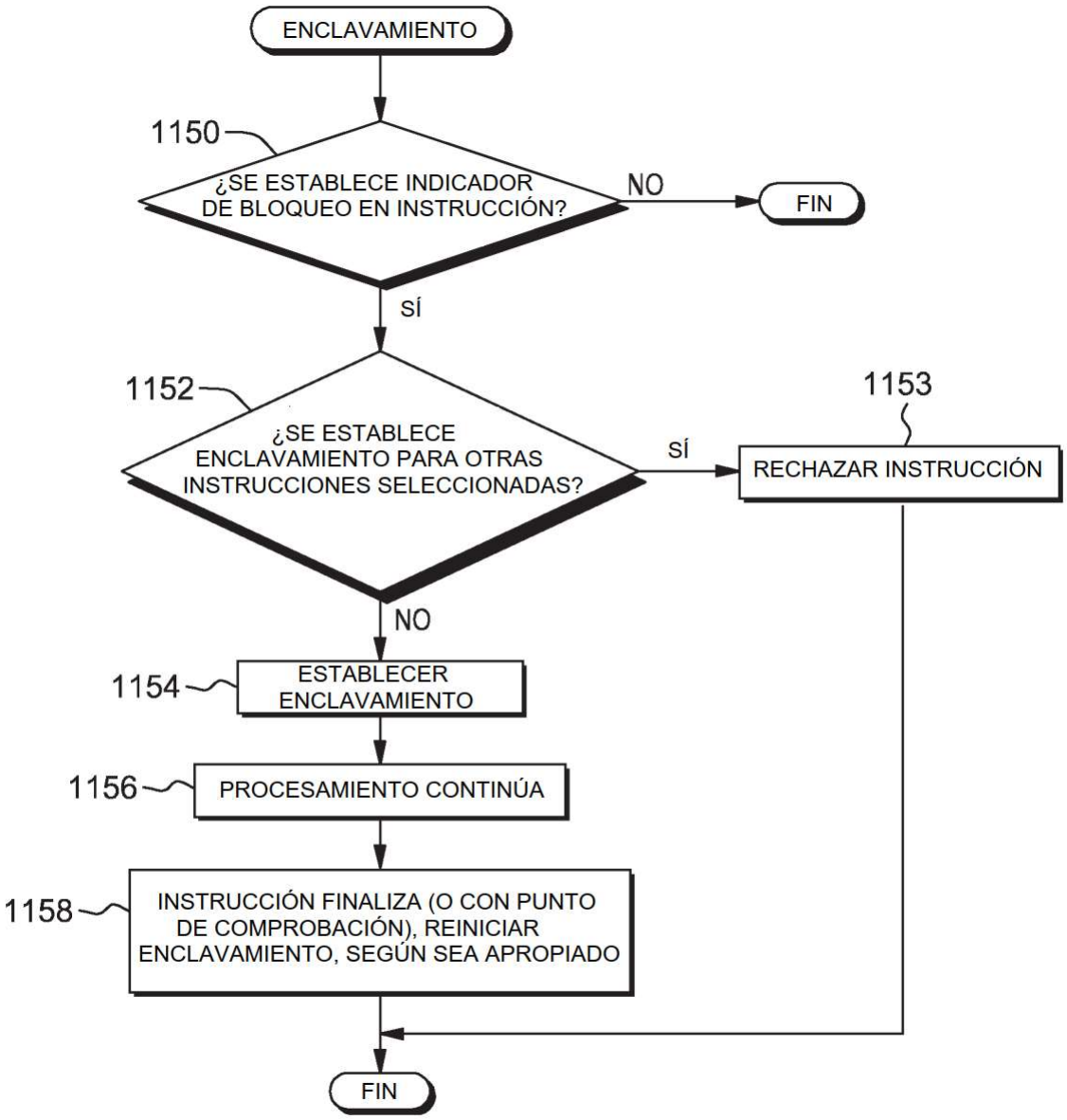
ENCLAVAMIENTO
1150
¿SE ESTABLECE INDICADOR DE BLOQUEO EN INSTRUCCIÓN?
NO
FIN
SÍ
1152
¿SE ESTABLECE ENCLAVAMIENTO PARA OTRAS INSTRUCCIONES SELECCIONADAS?
SÍ
1153
RECHAZAR INSTRUCCIÓN
NO
1154
ESTABLECER ENCLAVAMIENTO
1156
PROCESAMIENTO CONTINÚA
1158
INSTRUCCIÓN FINALIZA (O CON PUNTO DE COMPROBACIÓN), REINICIAR ENCLAVAMIENTO, SEGÚN SEA APROPIADO
FIN

FIG. 11B

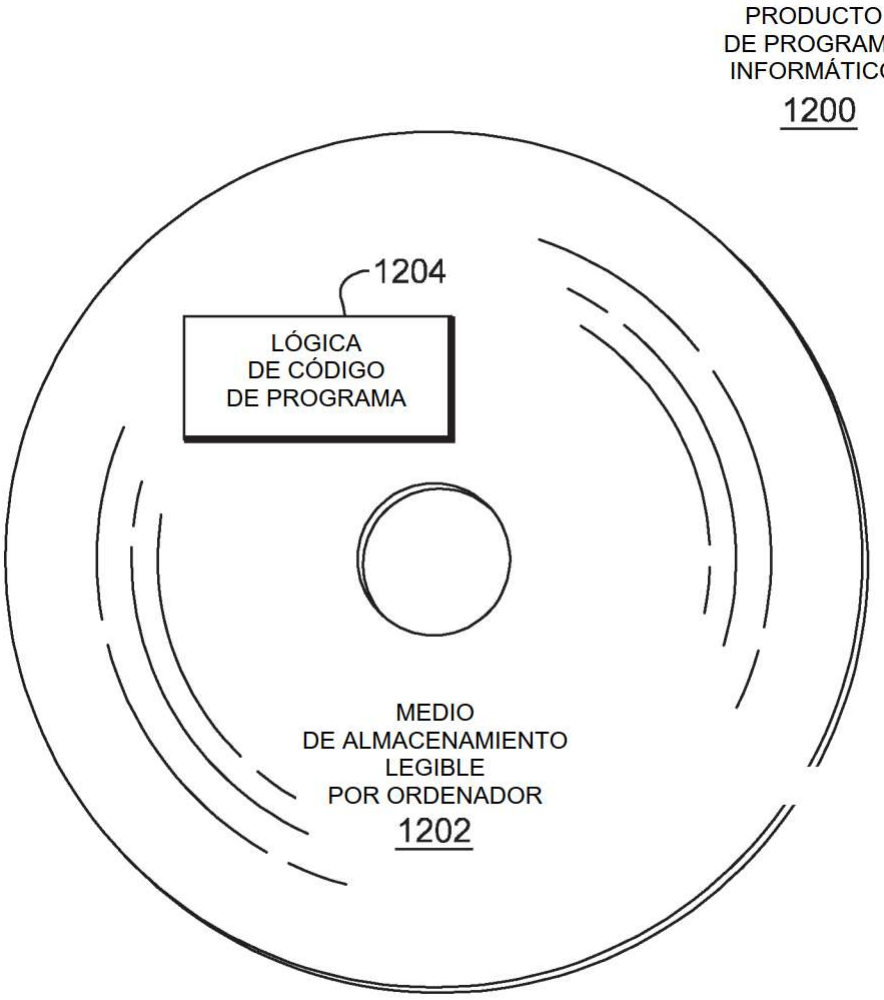
PRODUCTO
DE PROGRAMA
INFORMÁTICO
1200
1204
LÓGICA
DE CÓDIGO
DE PROGRAMA
MEDIO
DE ALMACENAMIENTO
LEGIBLE
POR ORDENADOR
1202

FIG. 12