(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 689 560**

(51) Int. CI.:

G06F 11/07 (2006.01)
G06F 11/36 (2006.01)
G06F 9/38 (2008.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **22.11.2012 PCT/IB2012/056622**

(87) Fecha y número de publicación internacional: **19.12.2013 WO13186600**

(96) Fecha de presentación y número de la solicitud europea: **22.11.2012 E 12878813 (0)**

(97) Fecha y número de publicación de la concesión europea: **22.08.2018 EP 2834739**

(54) Título: **Bloque de diagnóstico de transacción**

(30) Prioridad:

15.06.2012 US 201213524916

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
14.11.2018

(73) Titular/es:

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

(72) Inventor/es:

GREINER, DAN;
JACOBI, CHRISTIAN;
SLEGEL, TIMOTHY y
MITRAN, MARCEL

(74) Agente/Representante:

ELZABURU, S.L.P

DESCRIPCIÓN

Bloque de diagnóstico de transacción

Campo técnico

La presente invención se refiere, en general, a entornos informáticos de multiprocesamiento y, en concreto, al procesamiento de transacciones dentro de tales entornos informáticos.

Antecedentes

Un desafío permanente en la programación de los multiprocesadores es el de las actualizaciones a la misma ubicación de almacenamiento por múltiples unidades de procesamiento central (CPU – Central Processing Units, en inglés). Muchas instrucciones que actualizan ubicaciones de almacenamiento, incluidas operaciones lógicas simples, tales como AND, lo hacen con múltiples accesos a la ubicación. Por ejemplo, primero, se busca la ubicación de almacenamiento y, a continuación, se almacena el resultado actualizado.

Para que múltiples CPU puedan actualizar de manera segura la misma ubicación de almacenamiento, el acceso a la ubicación está serializado. Una instrucción, la instrucción TEST AND SET, presentada con la arquitectura S/360, anteriormente comercializada por International Business Machines Corporation, proporcionó una actualización enclavada de una ubicación de almacenamiento. Actualización enclavada significa que, tal como lo observan otras CPU y el subsistema entrada / salida (I/O – Input / Output, en inglés) (por ejemplo, el subsistema canales), todo el acceso de almacenamiento de la instrucción parece ocurrir atómicamente. Más tarde, la arquitectura S/370, comercializada por International Business Machines Corporation, introdujo las instrucciones COMPARAR E INTERCAMBIAR (COMPARE AND SWAP, en inglés) y COMPARAR DUPLICADO E INTERCAMBIAR (COMPARE DOUBLE AND SWAP, en inglés), que proporcionan un medio más sofisticado de llevar a cabo la actualización enclavada, y permiten la implementación de lo que comúnmente se conoce como palabra de bloqueo (o semáforo). Las instrucciones recientemente agregadas han proporcionado capacidades adicionales de actualización enclavada, que incluyen COMPARAR Y MODIFICAR Y PURGAR (COMPARE AND MODIFY AND PURGE, en inglés), y COMPARAR E INTERCAMBIAR Y ALMACENAR (COMPARE AND SWAP AND STORE, en inglés). No obstante, todas estas instrucciones proporcionan enclavamiento solo para una única ubicación de almacenamiento.

Las técnicas de programación más complejas pueden requerir la actualización enclavada de múltiples ubicaciones de almacenamiento, tal como cuando se agrega un elemento a una lista doblemente vinculada. En dicha operación, tanto un puntero hacia adelante como hacia atrás deben aparecer actualizados simultáneamente, tal como lo observan otras CPU y el subsistema I/O. Con el fin de efectuar dicha actualización de ubicación múltiple, el programa es obligado a utilizar un único punto de serialización separado, tal como una palabra de bloqueo. No obstante, las palabras de bloqueo pueden proporcionar un nivel de serialización mucho mayor de lo que está garantizado; por ejemplo, las palabras de bloqueo pueden serializar una cola completa de millones de elementos, aunque solo se estén actualizando dos elementos. El programa puede estructurar los datos para utilizar serialización de grano más fino (por ejemplo, una jerarquía de puntos de bloqueo), pero eso introduce problemas adicionales, tales como potenciales condiciones de interbloqueo si se viola la jerarquía, y problemas de búsqueda, si el programa encuentra un error mientras mantiene uno o más bloqueos, o si no se puede obtener el bloqueo.

Además de lo anterior, existen numerosos escenarios en los que un programa puede ejecutar una secuencia de instrucciones que pueden o no resultar en una condición de excepción. Si no ocurre ninguna condición de excepción, entonces el programa continúa; sin embargo, si se reconoce una excepción, entonces el programa puede tomar medidas correctoras para eliminar la condición de excepción. Java, como ejemplo, puede aprovechar dicha ejecución, por ejemplo, en la ejecución especulativa, en la colocación parcial en línea de una función y/o en la re-secuenciación de la verificación del puntero nulo (null, en inglés).

En entornos clásicos del sistema operativo, tales como z/OS y sus predecesores comercializados por International Business Machines Corporation, el programa configura un entorno de búsqueda para interceptar cualquier condición de excepción del programa que este pueda encontrar. Si el programa no intercepta la excepción, el sistema operativo típicamente termina anormalmente el programa para excepciones que el sistema operativo no está preparado para manejar. Configurar y aprovechar dicho entorno es costoso y complicado.

En el documento US2012/0030518, un procesador incluye una unidad de ejecución y, por lo menos, un registro del tipo de registro de última bifurcación (LBR – Last Branch Record, en inglés) para almacenar información de la dirección de una bifurcación tomada durante la ejecución del programa. Este registro puede almacenar además un indicador de transacción para indicar si la bifurcación fue tomada durante una trans acción de memoria de transacción (TM – Transaction Memory, en inglés). Este registro puede almacenar además un indicador de cancelación, para indicar si la bifurcación fue causada por una cancelación de transacción.

La Publicación de la Solicitud de Patente de Estados Unidos Nº US2008307267 presenta técnicas para diagnosticar de manera automática errores en un sistema de software.

La Publicación de la Solicitud de Patente de Estados Unidos Nº US2010332901 propone un sistema que facilita la ejecución de una transacción para un programa en un sistema de memoria de transacción soportado por hardware.

La Publicación de la Solicitud de Patente de Estados Unidos Nº US2011191545 se refiere a un sistema y a un método para llevar a cabo operaciones de memoria en un sistema informático.

Compendio

La presente invención se define en las reivindicaciones adjuntas. Las realizaciones o ejemplos de la siguiente descripción que no están cubiertos por las reivindicaciones adjuntas se considera que no forman parte de la invención de acuerdo con esta descripción. Se abordan los inconvenientes de la técnica anterior y se proporcionan ventajas por medio de la provisión de un producto de programa informático tal como se define en la reivindicación 1 adjunta, para proporcionar información de diagnóstico sobre cancelaciones de transacciones.

Asimismo, en el presente documento se describen y reivindican métodos y sistemas relacionados con una o más realizaciones.

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

Breve descripción de los dibujos

A continuación, se describirán realizaciones de la invención, solamente a modo de ejemplo, haciendo referencia a los dibujos adjuntos, en los que:

la figura 1 representa una realización de un entorno informático;

la figura 2A representa un ejemplo de una instrucción de Inicio de transacción (TBEGIN – Transaction BEGIN, en inglés);

la figura 2B representa una realización de más detalles de un campo de la instrucción TBEGIN de la figura 2A;

la figura 3A representa un ejemplo de una instrucción de Restricción de inicio de transacción (TBEGINC – Transaction BEGIN Constraint, en inglés);

la figura 3B representa una realización de más detalles de un campo de la instrucción TBEGINC de la figura 3A;

la figura 4 representa un ejemplo de una instrucción de Fin de transacción (TEND – Transaction END, en inglés);

la figura 5 representa un ejemplo de una instrucción de Cancelación de Transacción (TABORT – Transaction ABORT, en inglés);

la figura 6 representa un ejemplo de transacciones anidadas;

la figura 7 representa un ejemplo de una instrucción de Almacenamiento no de transacción (NTSTG – Non Transactional Store, en inglés);

la figura 8 representa un ejemplo de una instrucción de Extraer profundidad de anidamiento de transacciones (ETND - Extract Transaction Nesting Depth, en inglés);

la figura 9 representa un ejemplo de un bloque de diagnóstico de transacción;

la figura 10 representa ejemplos de motivos de cancelación, junto con códigos de cancelación y códigos de condición asociados;

la figura 11 representa una realización de la lógica asociada con la ejecución de una instrucción TBEGINC;

la figura 12 representa una realización de la lógica asociada con la ejecución de una instrucción TBEGIN;

la figura 13 representa una realización de la lógica asociada con la ejecución de una instrucción TEND;

la figura 14 representa una realización de la lógica asociada con el procesamiento de cancelación de transacción;

la figura 15 representa una realización de la lógica asociada con el almacenamiento selectivo de información en uno o más bloques de diagnóstico de transacción;

las figuras 16A - 16B representan un ejemplo de inserción de un elemento de cola en una lista doblemente vinculada de elementos de cola;

la figura 17 representa una realización de un producto de programa informático;

la figura 18 representa una realización de un sistema informático anfitrión (host, en inglés);

la figura 19 representa otro ejemplo de un sistema informático;

la figura 20 representa otro ejemplo de un sistema informático que comprende una red informática;

la figura 21 representa una realización de diversos elementos de un sistema informático;

la figura 22A representa una realización de la unidad de ejecución del sistema informático de la figura 21;

la figura 22B representa una realización de la unidad de conexión del sistema informático de la figura 21;

la figura 22C representa una realización de la unidad de carga / almacenamiento del sistema informático de la figura 21; y

la figura 23 representa una realización de un sistema informático anfitrión emulado.

Descripción detallada

De acuerdo con una realización, se proporciona una función de ejecución de transacción (TX – Transaction Execution, en inglés). Esta función proporciona procesamiento de transacción para instrucciones, y, en una o más realizaciones, ofrece modos de ejecución diferentes, tal como se describe a continuación, así como niveles anidados de procesamiento de transacción.

La función de ejecución de transacción introduce un estado de CPU denominado modo de ejecución de transacción (TX). Después de un reinicio de la CPU, la CPU no está en el modo TX. La CPU entra en el modo TX mediante una instrucción TRANSACTION BEGIN. La CPU sale del modo TX mediante (a) una instrucción TRANSACTION END más exterior (más detalles sobre interior y exterior se muestran a continuación) o (b) la transacción se cancela. Mientras está en el modo TX, los accesos al almacenamiento por parte de la CPU parecen ser simultáneos al bloque, tal como lo observan otras CPU y el subsistema I/O. Los accesos al almacenamiento (a) están comprometidos para el almacenamiento cuando la transacción más exterior finaliza sin ser cancelada (es decir, las actualizaciones realizadas en una memoria caché o en una memoria de almacenamiento temporal local a la CPU se propagan y almacenan en la memoria real y son visibles para otras CPU), o (b) son descartados si se cancela la transacción.

Las transacciones pueden estar anidadas. Es decir, mientras la CPU está en el modo TX, puede ejecutar otra instrucción TRANSACTION BEGIN. La instrucción que hace que la CPU entre en el modo TX se denomina TRANSACTION BEGIN más exterior; de manera similar, se dice que el programa está en la transacción más exterior. Las ejecuciones posteriores de TRANSACTION BEGIN se denominan instrucciones interiores; y el programa está ejecutando una transacción interior. El modelo proporciona una profundidad mínima de anidamiento y una profundidad máxima de anidamiento que depende del modelo. Una instrucción EXTRACT TRANSACTION NESTING DEPTH (Extraer la profundidad de anidamiento de la transacción) devuelve el valor de la profundidad de anidamiento actual y, en una realización adicional, puede devolver un valor máximo de la profundidad de anidamiento. Esta técnica utiliza un modelo denominado "anidamiento aplanado" en el que una condición de cancelación a cualquier profundidad de anidamiento provoca la cancelación de todos los niveles de la transacción, y el control es devuelto a la instrucción que sigue a la TRANSACTION BEGIN más exterior.

Durante el procesamiento de una transacción, se dice que un acceso de transacción realizado por una CPU entra en conflicto con (a) un acceso de transacción o un acceso no de transacción realizado por otra CPU, o (b) un acceso no de transacción realizado por el subsistema I/O, si ambos accesos están en cualquier ubicación dentro de la misma línea de memoria caché, y uno o los dos de los accesos son un almacén. En otras palabras, para que la ejecución de transacción sea productiva, no se debe observar que la CPU lleve a cabo accesos de transacción hasta que esté comprometida. Este modelo de programación puede ser altamente efectivo en ciertos entornos; por ejemplo, la actualización de dos puntos en una lista doblemente vinculada de un millón de elementos. No obstante, puede ser menos eficaz, si existe mucho conflicto para las ubicaciones de almacenamiento a las que se accede en modo de transacción.

En un modelo de ejecución de transacción (denominado en el presente documento transacción no restringida), cuando se cancela una transacción, el programa puede intentar redirigir la transacción con la esperanza de que la condición de cancelación ya no esté presente, o el programa puede "retroceder" a una ruta no de transacción equivalente. En otro modelo de ejecución de transacción (denominado en el presente documento transacción restringida), la CPU redirige de manera automática una transacción cancelada; en ausencia de violaciones de restricciones, se garantiza la eventual finalización de la transacción restringida.

Cuando se inicia una transacción, el programa puede especificar varios controles, tales como (a) qué registros generales son restaurados a sus contenidos originales si se cancela la transacción, (b) si la transacción puede modificar el contexto de registro de coma flotante, incluyendo, por ejemplo, registros de coma flotante y el registro de control de coma flotante, (c) si la transacción puede modificar los registros de acceso (AR – Access Registers, en

inglés), y (d) si ciertas condiciones de excepción de programa deben estar bloqueadas para causar una interrupción. Si se cancela una transacción no restringida, se puede proporcionar diversa información de diagnóstico. Por ejemplo, la instrucción TBEGIN más exterior que inicia una transacción no restringida puede designar un bloque de diagnóstico de transacción (TDB – Transaction Diagnostic Block, en inglés) especificado por el programa. Además, el TDB en el área del prefijo de la CPU o designado por la descripción del estado del servidor puede ser utilizado asimismo si la transacción se interrumpe debido a una interrupción del programa o a una condición que hace que finalice la ejecución interpretativa, respectivamente.

Anteriormente se han indicado diversos tipos de registros. Estos se explican más detalladamente en el presente documento. Los registros generales pueden ser utilizados como acumuladores en operaciones aritméticas y lógicas generales. En una realización, cada registro contiene posiciones de 64 bits, y hay 16 registros generales. Los registros generales están identificados por los números 0 a 15, y están designados por un campo R de cuatro bits en una instrucción. Algunas instrucciones permiten direccionar múltiples registros generales al tener varios campos R. Para algunas instrucciones, la utilización de un registro general específico está implícita en lugar de designada explícitamente por un campo R de la instrucción.

Además de su utilización como acumuladores en operaciones aritméticas generales y lógicas, 15 de los 16 registros generales se utilizan asimismo como registros de base de direcciones e índices en la generación de direcciones. En estos casos, los registros están designados mediante un campo B o un campo X de cuatro bits en una instrucción. Un valor de cero en el campo B o X especifica que no se aplica ninguna base o índice, y, por lo tanto, el registro general 0 no se debe designar como que contiene una dirección o índice de base.

Las instrucciones de coma flotante utilizan un conjunto de registros de coma flotante. La CPU tiene 16 registros de coma flotante, en una realización. Los registros de coma flotante se identifican con los números 0 a 15 y se designan mediante un campo R de cuatro bits en instrucciones de coma flotante. Cada registro de coma flotante tiene una longitud de 64 bits y puede contener un operando de coma flotante corto (32 bits) o uno largo (64 bits).

Un registro de control de coma flotante (FPC) es un registro de 32 bits que contiene bits de máscara, indicadores, un código de excepción de datos y bits de modo de redondeo, y se utiliza durante el procesamiento de operaciones de coma flotante.

Además, en una realización, la CPU tiene 16 registros de control, teniendo cada uno posiciones de 64 bits. Las posiciones de los bits en los registros se asignan a funciones particulares en el sistema, tal como el Registro de eventos de programa (PER – Program Event Recording, en inglés) (que se analiza a continuación), y se utilizan para especificar que una operación puede ser realizada o para proporcionar información especial requerida por la función. En una realización, para la función de transacción, se utilizan CR0 (bits 8 y 9) y CR2 (bits 61 a 63), tal como se describe a continuación.

La CPU tiene, por ejemplo, 16 registros de acceso numerados 0 a 15. Un registro de acceso consta de posiciones de 32 bits que contienen una especificación indirecta de un elemento de control del espacio de direcciones (ASCE – Address Space Control Element, en inglés). Un elemento de control de espacio de direcciones es un parámetro utilizado por el mecanismo de traducción dinámica de direcciones (DAT – Dynamic Address Translation, en inglés) para traducir referencias en un espacio de direcciones correspondiente. Cuando la CPU está en un modo denominado modo de registro de acceso (controlado por bits en la palabra de estado del programa (PSW – Program Status Word, en inglés)) un campo B de instrucción, utilizado para especificar una dirección lógica para una referencia de operando de almacenamiento, designa un registro de acceso, y el elemento de control del espacio de direcciones especificado por el registro de acceso es utilizado por la DAT para la referencia que se está realizando. Para algunas instrucciones, se utiliza un campo R en lugar de un campo B. Se proporcionan instrucciones para cargar y almacenar los contenidos de los registros de acceso y para mover los contenidos de un registro de acceso a otro.

Cada uno de los registros de acceso 1 a 15 puede designar cualquier espacio de direcciones. El registro de acceso 0 designa el espacio de instrucción principal. Cuando se utiliza uno de los registros de acceso 1 a 15 para designar un espacio de direcciones, la CPU determina qué espacio de direcciones se designa traduciendo los contenidos del registro de acceso. Cuando se utiliza el registro de acceso 0 para designar un espacio de direcciones, la CPU trata el registro de acceso como que designa el espacio de instrucción principal, y no examina los contenidos reales del registro de acceso. Por lo tanto, los 16 registros de acceso pueden designar, en cualquier momento, el espacio de instrucción principal y un máximo de 15 espacios adicionales.

En una realización, existen múltiples tipos de espacios de direcciones. Un espacio de direcciones es una secuencia consecutiva de números enteros (direcciones virtuales), junto con los parámetros de transformación que permiten que cada número se asocie con una ubicación de byte en el almacenamiento. La secuencia comienza en cero y continúa de izquierda a derecha.

Por ejemplo, en la Arquitectura/z, cuando una CPU utiliza una dirección virtual para acceder al almacenamiento principal (a.k.a., memoria principal), primero se convierte, por medio de la traducción dinámica de direcciones (DAT), en una dirección real, y a continuación, mediante el prefijo, a una dirección absoluta. La DAT puede utilizar de uno a

cinco niveles de tablas (página, segmento, región tercera, región segunda y región primera) como parámetros de transformación. La designación (origen y longitud) de la tabla de más alto nivel para un espacio de direcciones específico se denomina elemento de control del espacio de direcciones, y se encuentra para ser utilizado por la DAT en un registro de control o según especifica un registro de acceso. De manera alternativa, el elemento de control del espacio de direcciones para un espacio de direcciones puede ser una designación de espacio real, lo que indica que la DAT debe traducir la dirección virtual tratándola simplemente como una dirección real y sin utilizar ninguna tabla.

La DAT utiliza, en diferentes momentos, los elementos de control del espacio de direcciones en diferentes registros de control o especificados por los registros de acceso. La elección está determinada por el modo de traducción especificado en la PSW actual. Hay cuatro modos de traducción disponibles: modo de espacio primario, modo de espacio secundario, modo de registro de acceso y modo de espacio doméstico. Diferentes espacios de direcciones son direccionables dependiendo del modo de traducción.

En cualquier instante, cuando la CPU está en el modo de espacio primario o en el modo de espacio secundario, la CPU puede traducir direcciones virtuales que pertenecen a dos espacios de direcciones - el espacio de direcciones principal y el segundo espacio de direcciones. En cualquier momento cuando la CPU está en el modo de registro de acceso, puede traducir direcciones virtuales de hasta 16 espacios de direcciones - el espacio de direcciones principal y hasta 15 AR- espacios de direcciones especificados. En cualquier momento cuando la CPU se encuentra en el modo de espacio doméstico, puede traducir las direcciones virtuales del espacio de direcciones particular.

El espacio de direcciones principal se identifica de este modo porque consiste en direcciones virtuales principales, que se traducen por medio del elemento de control del espacio de direcciones principal (ASCE). De manera similar, el espacio de direcciones secundario consiste en direcciones virtuales secundarias traducidas por medio del ASCE secundario; los espacios de direcciones del AR especificados constan de direcciones virtuales del AR especificadas traducidas por medio de los ASCE especificados por el AR; y el espacio de direcciones de inicio consiste en direcciones virtuales de inicio traducidas por medio del ASCE de origen. Los ASCE primarios y secundarios están en los registros de control 1 y 7, respectivamente. Los ASCE especificados por el AR están en entradas de segunda tabla ASN que son localizadas por medio de un proceso denominado traducción de registro de acceso (ART – Access Register Translation, en inglés) utilizando los registros de control 2, 5 y 8. El ASCE de origen está en el registro de control 13.

Una realización de un entorno informático para incorporar y utilizar uno o más aspectos de la función de transacción descrita en este documento se describe haciendo referencia a la figura 1.

Haciendo referencia a la figura. 1, en un ejemplo, el entorno informático 100 se basa en la Arquitectura/z, comercializada por International Business Machines (IBM®) Corporation, Armonk, Nueva York. La Arquitectura/z se describe en una publicación de IBM titulada "Arquitectura/z - Principles of Operation", Publicación Nº SA22-7932-08, 9ª Edición, agosto de 2010.

La ARQUITECTURA/Z, IBM, y Z/OS y Z/VM (referenciados a continuación) son marcas registradas de International Business Machines. Corporation, Armonk, Nueva York. Otros nombres utilizados en este documento pueden ser marcas registradas, marcas comerciales o nombres de productos de International Business Machines Corporation o de otras empresas.

Como ejemplo, el entorno informático 100 incluye un complejo de procesador central (CPC) 102 acoplado a uno o más dispositivos de entrada / salida (I/O) 106 por medio de una o más unidades de control 108. El complejo de procesador central 102 incluye, por ejemplo, uno o más procesadores centrales 110, una o más particiones 112 (por ejemplo, particiones lógicas (LP – Logical Partitions, en inglés)), un hipervisor de partición lógica 114, y un subsistema de entrada / salida 115, cada uno de los cuales se describe a continuación.

Los procesadores centrales 110 son recursos físicos del procesador asignados a las particiones lógicas. En concreto, cada partición lógica 112 tiene uno o más procesadores lógicos, cada uno de los cuales representa todo o una parte de un procesador físico 110 asignado a la partición. Los procesadores lógicos de una partición 112 particular pueden estar dedicados a la partición, de modo que el recurso del procesador 110 subyacente está reservado para esa partición; o compartido con otra partición, de modo que el recurso del procesador subyacente esté potencialmente disponible para otra partición.

Una partición lógica funciona como un sistema separado y tiene una o más aplicaciones, y opcionalmente, un sistema operativo residente 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, comercializado por International Business Machines Corporation, Armonk, Nueva York.

Las particiones lógicas 112 se gestionan mediante un hipervisor 114 de partición lógica, que se implementa mediante firmware que se ejecuta en los procesadores 110. Tal como se utiliza en el presente documento, el firmware incluye, por ejemplo, el microcódigo y/o el milicódigo del procesador. Incluye, por ejemplo, las instrucciones de nivel de hardware y/o las estructuras de datos utilizadas en la implementación del código de máquina de nivel superior. En una realización, incluye, por ejemplo, un código propietario que, típicamente, se suministra como un

microcódigo que incluye software de confianza o un microcódigo específico para el hardware subyacente, y controla el acceso del sistema operativo al hardware del sistema.

Cada uno de as particiones lógicas y el hipervisor de partición lógica comprenden cada uno o más programas que residen en las particiones respectivas del almacenamiento central asociado con los procesadores centrales. Un ejemplo de hipervisor 114 de partición lógica es el recurso del procesador / administrador del sistema (PR/SM – Resource Processor / System Manager, en inglés), comercializado por International Business Machines Corporation, Armonk, Nueva York.

El subsistema entrada / salida 115 dirige el flujo de información entre los dispositivos de entrada / salida 106 y el almacenamiento principal (a.k.a., memoria principal). Está acoplado al complejo de procesamiento central, en el sentido de que puede ser una parte del complejo de procesamiento central o estar separado del mismo. El subsistema I/O libera a los procesadores centrales de la tarea de comunicarse directamente con los dispositivos de entrada / salida, y permite que el procesamiento de datos continúe al mismo tiempo que el procesamiento de entrada / salida. Para proporcionar comunicaciones, el subsistema I/O emplea adaptadores de comunicaciones de I/O. Existen diversos tipos de adaptadores de comunicaciones que incluyen, por ejemplo, canales, adaptadores de I/O, tarjetas PCI, tarjetas Ethernet, tarjetas de Interfaz de almacenamiento para pequeños ordenadores (SCSI. Small Computer Storage Interface, en inglés), etc. En el ejemplo particular descrito en el presente documento, los adaptadores de comunicaciones de I/O son canales, y, por lo tanto, el subsistema I/O se denomina en el presente documento subsistema de canales. No obstante, este es solo un ejemplo. Se pueden utilizar otros tipos de subsistemas I/O.

El subsistema I/O utiliza una o más rutas de entrada / salida como enlaces de comunicación para gestionar el flujo de información hacia o desde los dispositivos de entrada / salida 106. En este ejemplo particular, estas rutas se denominan rutas de canales, ya que los adaptadores de comunicación son canales.

El entorno informático descrito anteriormente es solo un ejemplo de un entorno informático que se puede utilizar. Se pueden utilizar otros entornos, incluidos, entre otros, entornos no particionados, otros entornos particionados y/o entornos emulados; las realizaciones no están limitadas a ningún entorno en concreto.

De acuerdo con uno o más aspectos, la función de ejecución de transacción es una mejora de la CPU que proporciona los medios por los cuales la CPU puede ejecutar una secuencia de instrucciones, conocida como transacción, que puede acceder a múltiples ubicaciones de almacenamiento, incluida la actualización de esas ubicaciones. Según lo observado por otras CPU y por el subsistema I/O, la transacción (a) se completa en su totalidad como una única operación atómica, o (b) se cancela, posiblemente sin dejar evidencia de que alguna vez haya sido ejecutada (excepto por ciertas condiciones descritas en el presente documento). Por lo tanto, una transacción completada con éxito puede actualizar numerosas ubicaciones de almacenamiento sin ningún bloqueo especial, que es necesario en el modelo clásico de procesamiento múltiple.

La función de ejecución de transacción incluye, por ejemplo, uno o más controles; una o más instrucciones; procesamiento de transacción, incluida la ejecución restringida y no restringida; y procesamiento de cancelación, cada uno de los cuales se describe con más detalle a continuación.

En una realización, se utilizan tres controles de propósito especial, que incluyen una palabra de estado del programa (PSW) de cancelación de transacción, una dirección del bloque de diagnóstico de la transacción (TDB), y una profundidad de anidamiento de transacción; cinco bits de registro de control; y seis instrucciones generales, que incluyen TRANSACTION BEGIN (restringida y no restringida), TRANSACTION END, EXTRACT TRANSACTION NESTING DEPTH, TRANSACTION ABORT y NONTRANSACTIONAL STORE, para controlar la función de ejecución de transacción. Cuando se instala la función, se instala, por ejemplo, en todas las CPU en la configuración. Una indicación de función, bit 73 en una implementación, cuando existe una, indica que la función de ejecución de transacción está instalada.

Cuando la función de ejecución de transacción está instalada, la configuración proporciona una función de ejecución de transacción no restringida y, opcionalmente, una función de ejecución de transacción restringida, cada uno de los cuales se describe a continuación. Cuando las indicaciones 50 y 73 de la función, como ejemplos, son ambas una, se instala la función de ejecución de transacción restringida. Ambas indicaciones de la función se almacenan en la memoria en ubicaciones especificadas.

Tal como se utiliza en el presente documento, el nombre de instrucción TRANSACTION BEGIN se refiere a las instrucciones que tienen los mnemónicos TBEGIN (inicio de transacción para una transacción no restringida) y TBEGINC (inicio de transacción para una transacción no restringida). Las explicaciones relativas a una instrucción específica se indican mediante el nombre de la instrucción seguido de la tecla de acceso entre paréntesis o corchetes, o simplemente mediante el mnemotécnico.

Una realización de un formato de una instrucción TRANSACTION BEGIN (TBEGIN) está indicada en las figuras 2A a 2B. Como ejemplo, una instrucción TBEGIN 200 incluye un campo de código de operación 202 que incluye un código de operación que especifica una operación de inicio de transacción no restringida; un campo de base (B_1) 204; un campo de desplazamiento (D_1) 206; y un campo inmediato (I_2) 208. Cuando el campo B_1 es distinto de cero,

los contenidos del registro general especificado por B_1 204 son agregados a D_1 206 para obtener la dirección del primer operando.

Cuando el campo B_1 es distinto de cero, se aplica lo siguiente:

- Cuando la profundidad de anidamiento de la transacción es inicialmente cero, la dirección del primer operando designa la ubicación del bloque de diagnóstico de transacción de 256 bytes, denominado TDB especificado por TBEGIN (descrito con más detalle a continuación) en el que se puede almacenar diversa información de diagnóstico si la transacción es cancelada. Cuando la CPU está en el modo de espacio principal, o modo de registro de acceso, la dirección del primer operando designa una ubicación en el espacio de direcciones principal. Cuando la CPU está en el espacio secundario, o modo de espacio doméstico, la dirección del primer operando designa una ubicación en el espacio de direcciones secundario o doméstico, respectivamente. Cuando la DAT está desactivada, la dirección del bloque de diagnóstico de transacción (TDB) (TDBA) designa una ubicación en almacenamiento real.

Se determina la accesibilidad de la función al primer operando. Si es accesible, la dirección lógica del operando se coloca en la dirección del bloque de diagnóstico de la transacción (TDBA), y la TDBA es válida.

- Cuando la CPU ya se encuentra en el modo no restringido de ejecución de transacción, la TDBA no se modifica, y no se puede predecir si el primer operando ha sido probado para accesibilidad.

Cuando el campo B_1 es cero, no se detectan excepciones de acceso para el primer operando y, para la instrucción TBEGIN más exterior, la TDBA no es válida.

Los bits del campo I_2 se definen como sigue, en un ejemplo:

Máscara de Guardar Registro General (GRSM – General Register Save Mask, en inglés) 210 (figura 2B): Los bits 0 a 7 del campo I_2 contienen la máscara de guardar registro general (GRSM). Cada bit de la GRSM representa una pareja par - impar de registros generales, en donde el bit 0 representa los registros 0 y 1, el bit 1 representa los registros 2 y 3, y así sucesivamente. Cuando un bit en la GRSM de la instrucción TBEGIN más exterior es cero, la pareja de registros correspondiente no se guarda. Cuando un bit en el GRSM de la instrucción TBEGIN más exterior es uno, la pareja de registros correspondiente se guarda en una ubicación que depende del modelo a la que el programa no puede acceder directamente.

Si la transacción se cancela, las parejas de registros guardadas se restauran a su contenido cuando la instrucción más avanzada de TBEGIN fue ejecutada. Los contenidos de todos los demás registros generales (no guardados) no se restauran cuando se cancela una transacción.

La máscara de guardar registro general es ignorada en todos los TBEGIN excepto en el más exterior.

Permitir la modificación del AR (A) 212: El control A, bit 12 del campo I_2, controla si la transacción puede modificar un registro de acceso. El control efectivo del permiso de modificación de AR es el AND lógico del control A en la instrucción TBEGIN para el nivel de anidamiento actual y para todos los niveles exteriores.

Si el control A efectivo es cero, la transacción será cancelada con el código de cancelación 11 (instrucción restringida) si se lleva a cabo un intento de modificar cualquier registro de acceso. Si el control A efectivo es uno, la transacción no será cancelada si se modifica un registro de acceso (ausente de cualquier otra condición de cancelación).

Operación de permitir el coma flotante (F) 214: El control F, bit 13 del campo I_2, controla si la transacción está autorizada a ejecutar instrucciones de coma flotante especificadas. El control efectivo de la operación de permitir el coma flotante es el AND lógico del control F en la instrucción TBEGIN para el nivel de anidamiento actual y para todos los niveles exteriores.

Si el control F efectivo es cero, entonces (a) la transacción será cancelada con el código de cancelación 11 (instrucción restringida) si se lleva a cabo un intento de ejecutar una instrucción de coma flotante, y (b) el código de excepción de datos (DXC – Data eXception Code, en inglés) en el byte 2 del registro de control de coma flotante (FPCR – Floating Point Control Register, en inglés) no será configurado debido a ninguna condición de excepción del programa de excepción de datos. Si el control F efectivo es uno, entonces (a) la transacción no se cancelará si se lleva a cabo un intento de ejecutar una instrucción de coma flotante (en ausencia de cualquier otra condición de cancelación), y (b) el DXC en el FPCR puede ser configurado debido a una condición de excepción del programa de excepción de datos.

Control del filtrado de la interrupción del programa (PIFC – Program Interruption Filtering Control, en inglés) 216: Los bits 14 a 15 del campo I_2 son el control del filtrado de la interrupción del programa (PIFC). El PIFC controla si ciertas clases de condiciones de excepción del programa (por ejemplo, excepción en el direccionamiento, excepción en los datos, excepción en la operación, excepción en la protección, etc.) que ocurren mientras la CPU está en el modo de ejecución de transacción resulta en una interrupción.

El PIFC efectivo es el valor más alto del PIFC en la instrucción TBEGIN para el nivel de anidamiento actual y para todos los niveles exteriores. Cuando el PIFC efectivo es cero, todas las condiciones de excepción del programa resultan en una interrupción. Cuando el PIFC efectivo es uno, las condiciones de excepción del programa que tienen una clase de ejecución de transacción de 1 y 2 resultan en una interrupción. (Cada condición de excepción del programa tiene asignada, por lo menos, una clase de ejecución de transacción, dependiendo de la gravedad de la excepción. La gravedad se basa en la probabilidad de búsqueda durante una ejecución repetida de la ejecución de transacción, y, si el sistema operativo necesita ver la interrupción.) Cuando el PIFC efectivo es dos, las condiciones de excepción del programa que tienen una clase de ejecución de transacción de 1 resultan en una interrupción. Se reserva un PIFC de 3.

Los bits 8 a 11 del campo I_2 (bits 40 a 43 de la instrucción) están reservados y deben contener ceros; de lo contrario, el programa puede no funcionar de manera compatible en el futuro.

Se describe una realización de un formato de una instrucción de restricción de inicio de transacción (TBEGINC) haciendo referencia a las figuras 3A a 3B. En un ejemplo, TBEGINC 300 incluye un campo de código de operación 302 que incluye un código de operación que especifica una operación de inicio de transacción restringida; un campo de base (B_1) 304; un campo de desplazamiento (D_1) 306; y un campo inmediato (I_2) 308. Los contenidos del registro general especificados por B_1 304 son agregados a D_1 306 para obtener la dirección del primer operando. No obstante, con la instrucción restringida de inicio de transacción, la dirección del primer operando no se utiliza para acceder al almacenamiento. Por el contrario, el campo B_1 de la instrucción incluye ceros; de lo contrario, se reconoce una excepción de especificación.

En una realización, el campo I_2 incluye diversos controles, un ejemplo de los cuales está representado en la figura 3B.

Los bits del campo I_2 se definen de la siguiente manera, en un ejemplo:

Máscara de guardar registro general (GRSM) 310: Los bits 0 a 7 del campo I_2 contienen la máscara de guardar registro general (GRSM). Cada bit de la GRSM representa una pareja par - impar de registros generales, en la que el bit 0 representa los registros 0 y 1, el bit 1 representa los registros 2 y 3, y así sucesivamente. Cuando un bit en la GRSM es cero, la pareja de registros correspondiente no se guarda. Cuando un bit en la GRSM es uno, la pareja de registros correspondiente se guarda en una ubicación que depende del modelo, a la que el programa no puede acceder directamente.

Si la transacción es cancelada, las parejas de registros guardadas se restauran a su contenido cuando se ejecuta la instrucción TRANSACTION BEGIN más exterior. Los contenidos de todos los demás registros generales (no guardados) no se restauran cuando una transacción restringida es cancelada.

Cuando se utiliza TBEGINC para continuar la ejecución en el modo no restringido de ejecución de transacción, se ignora la máscara de guardar registro general.

Permitir la modificación de AR (A) 312: El control A, bit 12 del campo I_2, controla si la transacción está autorizada a modificar un registro de acceso. El control efectivo de permitir modificación de AR es el AND lógico del control A en la instrucción TBEGINC para el nivel de anidamiento actual y para cualquier instrucción exterior de TBEGIN o TBEGINC.

Si el control A efectivo es cero, la transacción se cancelará con el código de cancelación 11 (instrucción restringida) si se lleva a cabo un intento de modificar cualquier registro de acceso. Si el control A efectivo es uno, la transacción no se cancelará si se modifica un registro de acceso (en ausencia de cualquier otra condición de cancelación).

Los bits 8 a 11 y 13 a 15 del campo I_2 (bits 40 a 43 y 45 a 47 de la instrucción) están reservados y deberían contener ceros.

El final de una instrucción de inicio de transacción se especifica mediante una instrucción de FIN DE TRANSACCIÓN (TEND - TRANSACTION END, en inglés), cuyo formato está representado en la figura 4. Como ejemplo, una instrucción TEND 400 incluye un campo de código de operación 402 que incluye un código de operación que especifica una operación de fin de transacción.

Se utilizan varios términos con respecto a la facilidad de ejecución de una transacción y, por lo tanto, únicamente por comodidad, a continuación, se proporciona una lista de términos en orden alfabético. En una realización, estos términos tienen la siguiente definición:

- Cancelar: una transacción es cancelada cuando finaliza antes de una instrucción TRANSACTION END que resulta en una profundidad de anidamiento de la transacción de cero. Cuando una transacción es cancelada, ocurre lo siguiente, en una realización:

- Los accesos al almacenamiento de transacción realizados por cualquiera y todos los niveles de la transacción son descartados (es decir, no están comprometidos).

- Los accesos al almacenamiento no de transacción realizados por cualquiera y todos los niveles de la transacción están comprometidos.

- Los registros designados por la máscara de guardar registro general (GRSM) de la instrucción TRANSACTION BEGIN más exterior son restaurados a sus contenidos antes de la ejecución de transacción (es decir, a sus contenidos en la ejecución de la instrucción TRANSACTION BEGIN más exterior). Los registros generales no designados por la máscara de guardar registro general de la instrucción TRANSACTION BEGIN más exterior no se restauran.

- Los registros de acceso, los registros de coma flotante y el registro de control de coma flotante no se restauran. Cualquier cambio realizado en estos registros durante la ejecución de la transacción se conserva cuando la transacción es cancelada.

Una transacción puede ser cancelada debido a una variedad de motivos, incluyendo el intento de ejecución de una instrucción restringida, el intento de modificación de un recurso restringido, un conflicto de transacción, exceder diversos recursos de la CPU, cualquier condición de interceptación de interpretación - ejecución, cualquier interrupción, una instrucción TRANSACTION ABORT, y otros motivos. Un código de cancelación de transacción proporciona motivos específicos por los cuales se puede cancelar una transacción.

Un ejemplo de un formato de una instrucción TRANSACTION ABORT (TABORT) se describe haciendo referencia a la figura 5. Como ejemplo, una instrucción TABORT 500 incluye un campo de código de operación 502 que incluye un código de operación que especifica una operación de cancelación de transacción; un campo de base (B_2) 504; y un campo de desplazamiento (D_2) 506. Cuando el campo B_2 es distinto de cero, los contenidos del registro general especificado por B_2 504 son agregados a D_2 506 para obtener una dirección del segundo operando; de lo contrario, la dirección del segundo operando se forma únicamente a partir del campo D_2, y el campo B_2 es ignorado. La dirección del segundo operando no se utiliza para direccionar datos; en cambio, la dirección forma el código de cancelación de transacción que se coloca en un bloque de diagnóstico de transacción durante el procesamiento de cancelación. El cálculo de dirección para la dirección del segundo operando sigue las reglas de la aritmética de direcciones: en el modo de direccionamiento de 24 bits, los bits 0 a 29 son configurados en ceros; en el modo de direccionamiento de 31 bits, los bits 0 a 32 son configurados en ceros.

Comprometer: cuando finaliza una instrucción TRANSACTION END más exterior, la CPU confirma los accesos al almacenamiento realizados por la transacción (es decir, la transacción más exterior y cualquier nivel anidado) de manera que sean visibles para otras CPU y para el subsistema I/O. Según lo observado por otras CPU y por el subsistema I/O, todos los accesos de búsqueda y almacenamiento realizados por todos los niveles anidados de la transacción parecen ocurrir como una única operación simultánea cuando se produce el compromiso.

Los contenidos de los registros generales, los registros de acceso, los registros de coma flotante y el registro de control de coma flotante no se modifican mediante el proceso de confirmación. Cualquier cambio realizado en estos registros durante la ejecución de transacción se conserva cuando se comprometen los almacenamientos de la transacción.

Conflicto: un acceso de transacción realizado por una CPU entra en conflicto con (a) un acceso de transacción o acceso no de transacción realizado por otra CPU, o (b) el acceso no de transacción realizado por el subsistema I/O, si ambos accesos son a cualquier ubicación dentro de la misma línea de memoria caché, y uno o más de los accesos es un almacenamiento.

Un conflicto puede ser detectado por la ejecución especulativa de instrucciones de la CPU, incluso aunque el conflicto no se detecte en la secuencia conceptual.

Transacción restringida: una transacción restringida es una transacción que se ejecuta en el modo restringido de ejecución de transacción y está sujeta a las siguientes limitaciones:

- un subconjunto de las instrucciones generales está disponible.

- Es posible ejecutar un número limitado de instrucciones.

- Se puede acceder a un número limitado de ubicaciones del operando de almacenamiento.

- La transacción está limitada a un único nivel de anidamiento.

En ausencia de interrupciones repetidas o conflictos con otras CPU o con el subsistema I/O, una transacción restringida finalmente se completa, por lo tanto, no se requiere una rutina de gestión de la cancelación. Las transacciones restringidas se describen en detalle a continuación.

Cuando se ejecuta una instrucción restringida TRANSACTION BEGIN (TBEGINC) mientras la CPU ya está en el modo no restringido de ejecución de transacción, la ejecución continúa como una transacción anidada no restringida.

Modo restringido de ejecución de transacción: cuando la profundidad de anidamiento de la transacción es cero y una transacción se inicia mediante una instrucción TBEGINC, la CPU entra en el modo restringido de ejecución de transacción. Mientras que la CPU está en el modo restringido de ejecución de transacción, la profundidad de anidamiento de la transacción es uno.

Transacción anidada: cuando se emite la instrucción TRANSACTION BEGIN mientras la CPU está en el modo no restringido de ejecución de transacción, la transacción es anidada.

La función de ejecución de transacción utiliza un modelo denominado anidamiento aplanado. En el modo de anidamiento aplanado, los almacenamientos creados por una transacción interior no son observables por otras CPU y por el subsistema I/O hasta que la transacción más exterior compromete sus almacenamientos. De manera similar, si una transacción es cancelada, todas las transacciones anidadas se cancelan, y todos los almacenamientos de la transacción de todas las transacciones anidadas son descartados.

Un ejemplo de transacciones anidadas se representa en la figura 6. Tal como se muestra, un primer TBEGIN 600 inicia una transacción más exterior 601, TBEGIN 602 inicia una primera transacción anidada, y TBEGIN 604 inicia una segunda transacción anidada. En este ejemplo, TBEGIN 604 y TEND 606 definen una transacción más interior 608. Cuando se ejecuta TEND 610, los almacenamientos de la transacción se comprometen 612 para la transacción más exterior y para todas las transacciones interiores.

Transacción no restringida: una transacción no restringida es una transacción que se ejecuta en el modo no restringido de ejecución de transacción. Aunque una transacción no restringida no está limitada en la manera de una transacción restringida, aún puede ser cancelada debido a una variedad de causas.

Modo no restringido de ejecución de transacción: cuando la instrucción TBEGIN inicia una transacción, la CPU entra en el modo no restringido de ejecución de transacción. Mientras que la CPU se encuentra en el modo no restringido de ejecución de transacción, la profundidad de anidamiento de la transacción puede variar desde una a la máxima profundidad de anidamiento de la transacción.

Acceso no de transacción: los accesos no de transacción son accesos a los operandos de almacenamiento realizados por la CPU cuando no está en el modo de ejecución de transacción (es decir, los clásicos accesos de almacenamiento fuera de una transacción). Además, los accesos realizados por el subsistema I/O son accesos no de transacción. Adicionalmente, la instrucción NONTRANSACTIONAL STORE se puede utilizar para provocar un acceso al almacenamiento no de transacción mientras la CPU está en el modo no restringido de ejecución de transacción.

Una realización de un formato de una instrucción NONTRANSACTIONAL STORE se describe haciendo referencia a la figura 7. Como ejemplo, una instrucción NONTRANSACTIONAL STORE 700 incluye una pluralidad de campos de código de operación 702a, 702b que especifican un código de operación que designa una operación de almacenamiento no de transacción; un campo de registro (R_1) 704 que especifica un registro, cuyos contenidos se denominan primer operando; un campo de índice (X_2) 706; un campo de base (B_2) 708; un primer campo de desplazamiento (DL_2) 710; y un segundo campo de desplazamiento (DH_2) 712. Los contenidos de los registros generales designados por los campos X_2 y B_2 son agregados a los contenidos de una concatenación de contenidos de los campos DH_2 y DL_2 para formar la dirección del segundo operando. Cuando uno o ambos campos X_2 o B_2 son cero, el registro correspondiente no toma parte en la agregación.

El primer operando de 64 bits no está colocado a la manera no de transacción en la segunda ubicación del operando.

El desplazamiento, formado por la concatenación de los campos DH_2 y DL_2, es tratado como un entero binario con signo de 20 bits.

El segundo operando debe ser alineado en un límite de doble palabra; de lo contrario, se reconoce la excepción de la especificación y se suprime la operación.

Transacción exterior / más exterior: una transacción con una profundidad de anidamiento de transacción de número inferior es una transacción exterior. Una transacción con un valor de uno de la profundidad de anidamiento de la transacción es la transacción más exterior.

Una instrucción TRANSACTION BEGIN más exterior es aquella que se ejecuta cuando la profundidad de anidamiento de la transacción es inicialmente cero. Una instrucción TRANSACTION END más exterior es la que hace que la profundidad de anidamiento de la transacción pase de uno a cero. Una transacción restringida es la transacción más exterior, en esta realización.

Filtrado de interrupción del programa: cuando se cancela una transacción debido a ciertas condiciones de excepción del programa, el programa puede evitar, opcionalmente, que se produzca la interrupción. Esta técnica se denomina filtrado de interrupción del programa. El filtrado de interrupción del programa está sujeto a la clase de transacción de

la interrupción, al control efectivo del filtrado de interrupción del programa de la instrucción TRANSACTION BEGIN y al reemplazo del filtro de interrupción del programa de ejecución de transacción en el registro de control 0.

Transacción: una transacción incluye los accesos al operando de almacenamiento realizados, y los registros generales seleccionados alterados, mientras que la CPU se encuentra en el modo de ejecución de la transacción. Para una transacción no restringida, los accesos al operando de almacenamiento pueden incluir tanto accesos de transacción como accesos no de transacción. Para una transacción restringida, los accesos de los operandos de almacenamiento están limitados a los accesos de transacción. Según lo observado por otras CPU y por el subsistema I/O, todos los accesos del operando de almacenamiento son realizados por la CPU, mientras que en el modo de ejecución de la transacción parece ocurrir como una sola operación simultánea. Si se cancela una transacción, se descartan los accesos al almacenamiento de transacción, y todos los registros designados por la máscara de guardar registro general de la instrucción TRANSACTION BEGIN más exterior son restaurados a sus contenidos antes de la ejecución de transacción.

Accesos de transacción: los accesos de transacción son accesos a los operandos de almacenamiento realizados mientras la CPU está en el modo de ejecución de transacción, con la excepción de los accesos realizados por la instrucción NONTRANSACTIONAL STORE.

Modo de ejecución de transacción: el término modo de ejecución de transacción (a.k.a., modo de ejecución de transacción) describe la operación común de los modos de ejecución de transacción no restringida y restringida. Por lo tanto, cuando se describe la operación, los términos no restringida y restringida se utilizan para calificar el modo de ejecución de transacción.

Cuando la profundidad de anidamiento de la transacción es cero, la CPU no está en el modo de ejecución de transacción (también denominado modo de ejecución no de transacción).

Según lo observado por la CPU, las búsquedas y los almacenamientos realizados en el modo de ejecución de transacción no son diferentes a los realizados mientras no está en el modo de ejecución de transacción.

En una realización de la Arquitectura/z, la función de ejecución de transacción está bajo el control de los bits 8 a 9 del registro de control 0, bits 61 a 63 del registro de control 2, la profundidad de anidamiento de la transacción, la dirección del bloque de diagnóstico de transacción y la palabra de estado del programa de cancelar transacción (PSW).

Después de un restablecimiento inicial de la CPU, el contenido de las posiciones de bit 8 a 9 del registro de control 0, las posiciones de bit 62 a 63 del registro de control 2 y la profundidad de anidamiento de la transacción son configuradas en cero. Cuando el control de ejecución de la transacción, bit 8 del registro de control 0, es cero, la CPU no puede colocarse en el modo de ejecución de la transacción.

A continuación, se describen más detalles con respecto a los diversos controles.

Tal como se indicó, la función de ejecución de transacción está controlado por dos bits en el registro de control cero y tres bits en el registro de control dos. Por ejemplo:

bits 0 del registro de control: las asignaciones de bits son las siguientes, en una realización:

control de ejecución de transacción (TXC): el bit 8 del registro de control cero es el control de ejecución de transacción. Este bit proporciona un mecanismo mediante el cual el programa de control (por ejemplo, el sistema operativo) puede indicar si el programa puede o no utilizar la función de ejecución de transacción. El bit 8 debe ser uno para entrar con éxito en el modo de ejecución de transacción.

Cuando el bit 8 del registro de control 0 es cero, el intento de ejecución de las instrucciones EXTRACT TRANSACTION NESTING DEPTH, TRANSACTION BEGIN y TRANSACTION END resulta en una ejecución de operación especial.

Una realización de un formato de una instrucción EXTRACT TRANSACTION NESTING DEPTH se describe haciendo referencia a la figura 8. Como ejemplo, una instrucción EXTRACT TRANSACTION NESTING DEPTH 800 incluye un campo de código de operación 802 que especifica un código de operación que indica la operación de extracción de la profundidad de anidamiento de la transacción; y un campo de registro R_1 804 que designa un registro general.

La profundidad de anidamiento de la transacción actual se coloca en los bits 48 a 63 del registro general R_1. Los bits 0 a 31 del registro permanecen sin cambios, y los bits 32 a 47 del registro son configurados en cero.

En otra realización, la profundidad máxima de anidamiento de la transacción se coloca asimismo en el registro general R_1, tal como en los bits 16 a 31.

Anulación del filtrado de interrupción del programa de ejecución de transacción (PIFO – transaction execution Program Interruption Filtering Override, en inglés): el bit 9 del registro de control cero es la anulación del filtrado de

interrupción del programa de ejecución de transacción. Este bit proporciona un mecanismo mediante el cual el programa de control puede garantizar que cualquier condición de excepción de programa que ocurra mientras la CPU está en el modo de ejecución de transacción produce una interrupción, independientemente del control efectivo del filtrado de interrupción del programa especificado o implícito por la instrucción o instrucciones TRANSACTION BEGIN.

Bits del registro de control 2: las asignaciones son las siguientes, en una realización:

Alcance del diagnóstico de transacción (TDS – Transaction Diagnostic Scope, en inglés): el bit 61 del registro de control 2 controla la aplicabilidad del control del diagnóstico de transacción (TDC) en los bits 62 a 63 del registro, como sigue:

TDS

Significado del valor

0 El TDC se aplica independientemente de si la CPU está en el estado de problema o de supervisor.

1 El TDC se aplica solo cuando la CPU está en el estado de problema. Cuando la CPU está en el estado de supervisor, el procesamiento es como si el TDC contuviera el cero.

Control del diagnóstico de transacción (TDC): los bits 62 a 63 del registro de control 2 son un entero de 2 bits sin signo que puede ser utilizado para provocar la cancelación aleatoria de las transacciones con fines de diagnóstico. La codificación del TDC es la siguiente, en un ejemplo:

TDC

Significado del valor

0 Operación normal; las transacciones no se cancelan como resultado del TDC.

1 Cancelar cada transacción en una instrucción aleatoria, pero antes de la ejecución de la instrucción TRANSACTION END más exterior.

2 Cancelar las transacciones aleatorias en una instrucción aleatoria.

3 Reservado

Cuando una transacción se cancela debido a un TDC distinto de cero, cualquiera de lo siguiente puede ocurrir:

- El código de cancelación es configurado en cualquiera de los códigos 7 a 11,13 a 16, o 255, siendo elegido el valor del código al azar por la CPU; el código de condición se configura de manera correspondiente al código de cancelación. Los códigos de cancelación se describen con más detalle a continuación.

- Para una transacción no restringida, el código de condición es configurado en uno. En este caso, el código de cancelación no es aplicable.

Es que depende del modelo si se implementa el valor 1 de TDC. Si no se implementa, un valor de 1 actúa como si se hubiera especificado un valor de 2.

Para una transacción restringida, un valor de TDC de 1 se trata como si se hubiera especificado un valor de TDC de 2.

Si se especifica un valor de TDC de 3, los resultados son impredecibles.

Dirección del bloque de diagnóstico de la transacción (TDBA)

Una dirección válida del bloque de diagnóstico de la transacción (TDBA) es configurada a partir de la dirección del primer operando de la instrucción TRANSACTION BEGIN (TBEGIN) más exterior cuando el campo B_1 de la instrucción es distinto de cero. Cuando la CPU está en el espacio primario o modo de registro de acceso, la TDBA designa una ubicación en el espacio de direcciones principal. Cuando la CPU está en el espacio secundario o en el modo de espacio doméstico, la TDBA designa una ubicación en el espacio de direcciones secundario o doméstico, respectivamente. Cuando la DAT (Traducción dinámica de direcciones) está desactivada, la TDBA designa una ubicación en el almacenamiento real.

La TDBA es utilizada por la CPU para localizar el bloque de diagnóstico de la transacción - denominado TDB especificado por TBEGIN - si la transacción es cancelada posteriormente. Los tres bits más a la derecha de la TDBA son cero, lo que significa que el TDB especificado por TBEGIN está en un límite de doble palabra.

Cuando el campo B_1 de una instrucción TRANSACTION BEGIN (TBEGIN) más exterior es cero, la dirección del bloque de diagnóstico de transacción no es válida y no se almacena ningún TDB especificado mediante TBEGIN si la transacción se cancela posteriormente.

PSW de cancelación de la transacción PSW (TAPSW – Transaction Abort PSW, en inglés)

Durante la ejecución de la instrucción TRANSACTION BEGIN (TBEGIN) cuando la profundidad de anidamiento es inicialmente cero, la PSW de la cancelación de la transacción es configurada en los contenidos de la PSW actual; y la dirección de la instrucción de la PSW de la cancelación de la transacción designa la siguiente instrucción secuencial (es decir, la instrucción que sigue a la TBEGIN más exterior). Durante la ejecución de la instrucción TRANSACCIÓN BEGIN restringida (TBEGINC) cuando la profundidad de anidamiento es inicialmente cero, la PSW de la cancelación de la transacción es configurada en los contenidos de la PSW actual, excepto por que la dirección de la instrucción de la PSW de la cancelación de la transacción designa la instrucción TBEGINC (en lugar de la siguiente instrucción secuencial posterior a la TBEGINC).

Cuando se cancela una transacción, el código de la condición en la PSW de la cancelación de la transacción es reemplazado con un código que indica la gravedad de la condición de la cancelación. Posteriormente, si la transacción fue cancelada debido a causas que no resultan en una interrupción, la PSW se carga desde la PSW de la cancelación de la transacción; si la transacción fue cancelada debido a causas que resultan en una interrupción, la PSW de la cancelación de la transacción es almacenada como la PSW anterior de la interrupción.

La PSW de la cancelación de la transacción no se altera durante la ejecución de ninguna instrucción TRANSACTION BEGIN interior.

Profundidad de anidamiento de la transacción (TND)

La profundidad de anidamiento de la transacción es, por ejemplo, un valor sin signo de 16 bits que se incrementa cada vez que se completa una instrucción TRANSACTION BEGIN con el código de condición 0, y disminuye cada vez que se completa una instrucción TRANSACTION END. La profundidad de anidamiento de la transacción es reiniciada a cero cuando se cancela una transacción o mediante el reinicio de la CPU.

En una realización, se implementa una TND máxima de 15.

En una implementación, cuando la CPU está en el modo restringido de ejecución de transacción, la profundidad de anidamiento de la transacción es uno. Además, aunque la TND máxima se puede representar como un valor de 4 bits, la TND se define como un valor de 16 bits para facilitar su inspección en el bloque de diagnóstico de la transacción.

Bloque de diagnóstico de transacción (TDB)

Cuando se cancela una transacción, se puede guardar diversa información de estado en un bloque de diagnóstico de transacción (TDB), como sigue:

1. TDB especificado por TBEGIN: para una transacción no restringida, cuando el campo B_1 de la instrucción TBEGIN más exterior es distinto de cero, la dirección del primer operando de la instrucción designa el TDB especificado por TBEGIN. Esta es una ubicación especificada del programa de aplicación que puede ser examinada por el gestor de la interrupción de la aplicación.

2. TDB de interrupción de programa (PI): si se cancela una transacción no restringida debido a una condición no filtrada de excepción del programa, o si se cancela una transacción restringida debido a cualquier condición de excepción del programa (es decir, cualquier condición que resulte en una interrupción del programa que se reconoce), el TDB de PI es almacenado en ubicaciones en el área del prefijo. Esto está disponible para inspección por parte del sistema operativo y posterior desconexión cualquier informe de diagnóstico que pueda proporcionar.

3. TDB de interceptación: si la transacción es cancelada debido a cualquier condición de excepción del programa que resulta en una interceptación (es decir, la condición hace que la ejecución interpretativa finalice y el control regrese al programa anfitrión), una TDB es almacenada en una ubicación especificada en el bloque de descripción de estado para el sistema operativo invitado.

El TDB especificada por TBEGIN solo se almacena, en una realización, cuando la dirección de TDB es válida (es decir, cuando el campo B_1 de la instrucción TBEGIN más exterior es distinto de cero).

Para cancelaciones debidas a condiciones no filtradas de excepción del programa, solo se almacenará uno de los TDB de PI o los TDB de interceptación. Por lo tanto, pueden existir cero, uno o dos TDB almacenados para una cancelación.

Más detalles con respecto a un ejemplo de cada uno de los TDB se describen a continuación:

TDB especificado por TBEGIN: la ubicación de 256 bytes especificada por una dirección válida del bloque de diagnóstico de transacción. Cuando la dirección del bloque de diagnóstico de la transacción es válida, el TDB especificado por TBEGIN es almacenado en una cancelación de transacción. El TDB especificado por TBEGIN está sujeto a todos los mecanismos de protección de almacenamiento que están en vigencia en la ejecución de la instrucción TRANSACTION BEGIN más exterior. Se detecta un evento de alteración del almacenamiento PER (Registro de Evento de Programa – Program Event Recording, en inglés) para cualquier porción del TDB especificado por TBEGIN durante la ejecución de la TBEGIN más exterior, no durante el procesamiento de cancelación de la transacción.

Uno de los propósitos de PER es ayudar en la depuración de programas. Permite alertar al programa sobre los siguientes tipos de eventos, por ejemplo:

- Ejecución con éxito de una instrucción de bifurcación. Se proporciona la opción de que un evento ocurra solo cuando la ubicación del objetivo de bifurcación se encuentra dentro del área de almacenamiento designada.
- Obtención de una instrucción del área de almacenamiento designada.
- Alteración de los contenidos del área de almacenamiento designada. Se proporciona la opción de que un evento ocurra solo cuando el área de almacenamiento está dentro de los espacios de direcciones designados.
- Ejecución de una instrucción STORE USING REAL ADDRESS.
- Ejecución de la instrucción TRANSACTION END.

El programa puede especificar selectivamente que uno o más de los tipos de eventos mencionados anteriormente sean reconocidos, excepto que el evento para STORE USING REAL ADDRESS puede especificarse solo junto con el evento de alteración de almacenamiento. La información relativa a un evento PER es proporcionada al programa por medio de una interrupción del programa, identificándose la causa de la interrupción en el código de interrupción.

Cuando la dirección del bloque de diagnóstico de la transacción no es válida, un TDB especificado por TBEGIN no se almacena.

TDB de interrupción de programa: ubicaciones reales 6,144 a 6,399 (1800 a 18FF hex). El TDB de interrupción del programa se almacena cuando una transacción se interrumpe debido a la interrupción del programa. Cuando una transacción se interrumpe debido a otras causas, los contenidos del TDB de interrupción del programa son impredecibles.

El TDB de interrupción del programa no está sujeto a ningún mecanismo de protección. Los eventos de alteración del almacenamiento no son detectados para el TDB de interrupción del programa cuando es almacenado durante una interrupción de programa.

TDB de interceptación: la ubicación real del anfitrión de 256 bytes especificada por las ubicaciones 488 a 495 de la descripción del estado. El TDB de interceptación se almacena cuando una transacción cancelada resulta en una interceptación de interrupción del programa invitado (es decir, código de interceptación 8). Cuando se cancela una transacción debido a otras causas, los contenidos del TDB de interceptación son impredecibles. El TDB de interceptación no está sujeto a ningún mecanismo de protección.

Tal como se representa en la figura 9, los campos de un bloque de diagnóstico de transacción 900 son los siguientes, en una realización:

Formato 902: El byte 0 contiene una indicación de validez y de formato, como sigue:

Significado del valor

0 Los campos restantes del TDB son impredecibles.

1 Un TDB de formato 1, cuyos campos restantes se describen a continuación.

2 a 255 Reservado

Un TDB en el que el campo de formato es cero se denomina TDB nulo.

Indicadores 904: el byte 1 contiene diversas indicaciones, como sigue:

Validez del testigo de conflicto (CTV – conflicto Testigo Validity, en inglés): cuando una transacción se cancela debido a un conflicto de búsqueda o de almacenamiento (es decir, códigos de cancelación 9 o 10, respectivamente), el bit 0 del byte 1 es la indicación de validez del testigo de conflicto. Cuando la indicación de CTV es uno, el testigo del conflicto 910 en los bytes 16 a 23 del TDB contiene la dirección lógica en la que fue detectado el conflicto. Cuando la indicación de CTV es cero, los bytes 16 a 23 del TDB son impredecibles.

Cuando una transacción se cancela debido a cualquier motivo que no sea un conflicto de búsqueda o de almacenamiento, el bit 0 del byte 1 se almacena como cero.

Indicación de transacción restringida (CTI – Constrained – Transaction Indication, en inglés): cuando la CPU se encuentra en el modo restringido de ejecución de transacción, el bit 1 del byte 1 es configurado en uno. Cuando la CPU está en modo restringido de ejecución de transacción, el bit 1 del byte 1 es configurado en cero.

Reservado: los bits 2 a 7 del byte 1 están reservados y almacenados como ceros.

Profundidad de anidamiento de la transacción (TND) 906: los bytes 6 a 7 contienen la profundidad de anidamiento de la transacción cuando la transacción fue cancelada.

Código de cancelación de transacción (TAC – Transaction Abort Code, en inglés) 908: los bytes 8 a 15 contienen un código de transacción sin signo de 64 bits. Cada punto de código indica un motivo por el cual se cancela una transacción.

Depende del modelo si el código de cancelación de la transacción se almacena en el TDB de interrupción del programa cuando se cancela una transacción debido a condiciones distintas de una interrupción de programa.

Testigo de conflicto 910: para las transacciones que son canceladas debido a un conflicto de búsqueda o de almacenamiento (es decir, códigos de cancelación 9 y 10, respectivamente), los bytes 16 a 23 contienen la dirección lógica de la ubicación de almacenamiento en la que se detectó el conflicto. El testigo del conflicto es significativo cuando el bit de CTV, el bit 0 del byte 1, es uno.

Cuando el bit de CTV es cero, los bytes 16 a 23 son impredecibles.

Debido a la ejecución especulativa de la CPU, el testigo del conflicto puede designar una ubicación de almacenamiento a la que no se accederá necesariamente mediante la secuencia de ejecución conceptual de la transacción.

Dirección de la instrucción de transacción abortada (ATIA) 912: los bytes 24 a 31 contienen una instrucción que identifica la instrucción que se estaba ejecutando cuando se detectó una cancelación. Cuando se cancela una transacción debido a los códigos de cancelación 2, 5, 6, 11, 13 o 256 o más, o cuando se cancela una transacción debido a los códigos de cancelación 4 o 13 y la condición de excepción del programa es de cancelación, la ATIA apunta directamente a la instrucción que se estaba ejecutando. Cuando se cancela una transacción debido a los códigos de cancelación 4 o 12, y la condición de excepción del programa no es de cancelación, la ATIA señala más allá de la instrucción que se estaba ejecutando.

Cuando se cancela una transacción debido a los códigos de cancelación 7 a 10, 14 a 16 o 255, la ATIA no necesariamente indica la instrucción exacta que provoca la cancelación, sino que puede apuntar a una instrucción anterior o posterior dentro de la transacción.

Si una transacción se cancela debido a una instrucción que es el objetivo de una instrucción del tipo de ejecución, la ATIA identifica la instrucción del tipo de ejecución, apuntando a la instrucción o más allá de ella, dependiendo del código de cancelación tal como se describió anteriormente. La ATIA no indica el objetivo de la instrucción del tipo de ejecución.

La ATIA está sujeta al modo de direccionamiento cuando se cancela la transacción. En el modo de direccionamiento de 24 bits, los bits 0 a 40 del campo contienen ceros. En el modo de direccionamiento de 31 bits, los bits 0 a 32 del campo contienen ceros.

Depende del modelo si la dirección de la instrucción de transacción cancelada se almacena Enel TDB de interrupción de programa cuando se cancela una transacción debido a condiciones distintas de una interrupción de programa.

Cuando una transacción se cancela debido al código de cancelación 4 o 12, y la condición de excepción del programa no es de cancelación, la ATIA no apunta a la instrucción que causa la cancelación. Restando el número de medias palabras indicado por el código de longitud de la interrupción (ILC – Interruption Length Code, en inglés) de la ATIA, la instrucción que causa la cancelación se puede identificar en condiciones que son de supresión o de terminación, o para eventos que no son PER que son de finalización. Cuando una transacción se cancela debido a un evento PER y no está presente ninguna otra condición de excepción del programa, la ATIA es impredecible.

Cuando la dirección de bloque de diagnóstico de transacción es válida, el ILC puede ser examinado en la identificación de la interrupción del programa (PIID) en los bytes 36 a 39 del TDB especificado por TBEGIN. Cuando no se aplica el filtrado, el ILC puede ser examinado en la PIID en la ubicación 140 a 143 en almacenamiento real.

Identificación de acceso a excepción (EAID) 914: para las transacciones que se cancelan debido a ciertas condiciones filtradas de excepción del programa, el byte 32 del TDB especificado por TBEGIN contiene la identificación de acceso a la excepción. En un ejemplo de la Arquitectura/z, el formato de la EAID y los casos para

los que se almacena son los mismos que los descritos en la ubicación real 160, cuando la condición de excepción resulta en una interrupción, tal como se describe en los principios de funcionamiento mencionados anteriormente.

Para transacciones que se cancelan por otros motivos, incluyendo cualquier condición de excepción que resulta en una interrupción del programa, el byte 32 es impredecible. El byte 32 es impredecible en el TDB de interrupción del programa.

Este campo se almacena solo en el TDB designado por la dirección del bloque de diagnóstico de la transacción; de lo contrario, el campo está reservado. La EAID se almacena solo para la lista de acceso controlado o la protección DAT, tipo ASCE, traducción de página, primera traducción de región, segunda traducción de región, tercera traducción de región y condiciones de excepción de programa de traducción de segmento.

Código de excepción de datos (DXC) 916: para las transacciones que se cancelan debido a condiciones filtradas de excepción del programa de excepción de datos, el byte 33 del TDB especificado por TBEGIN contiene el código de excepción de datos. En un ejemplo de la Arquitectura/z, el formato de DXC y los casos para los que se almacena son los mismos que los descritos en la ubicación real 147 cuando la condición de excepción resulta en una interrupción, tal como se describe en los principios de funcionamiento mencionados anteriormente. En un ejemplo, la ubicación 147 incluye el DXC.

Para las transacciones que se cancelan por otros motivos, incluidas las condiciones de excepción que resultan en una interrupción del programa, el byte 33 es impredecible. El byte 33 es impredecible en el TDB de interrupción del programa.

Este campo se almacena solo en el TDB designado por la dirección del bloque de diagnóstico de la transacción; de lo contrario, el campo está reservado. El DXC se almacena solo para las condiciones de excepción del programa de datos.

Identificación de Interrupción de Programa (PIID) 918: Para transacciones que se cancelan debido a condiciones filtradas de excepción del programa, los bytes 36 a 39 del TDB especificado por TBEGIN contienen la identificación de la interrupción del programa. En un ejemplo de la Arquitectura/z, el formato del PIID es el mismo que el descrito en las ubicaciones reales 140 a 143 cuando la condición resulta en una interrupción (tal como se describe en los principios de funcionamiento mencionados anteriormente), excepto por que el código de la longitud de la instrucción en los bits 13 a 14 del PIID es relativo a la instrucción en la que se detectó la condición de excepción.

Para las transacciones que se cancelan por otros motivos, incluidas las condiciones de excepción que resultan en una interrupción del programa, los bytes 36 a 39 son impredecibles. Los bytes 36 a 39 son impredecibles en el TDB de la interrupción del programa.

Este campo se almacena solo en el TDB designado por la dirección del bloque de diagnóstico de la transacción; de lo contrario, el campo está reservado. La identificación de la interrupción del programa solo se almacena para las condiciones de excepción del programa.

Identificación de excepción de traducción (TEID – Translation Exception IDentification, en inglés) 920: para transacciones que se cancelan debido a cualquiera de las siguientes condiciones filtradas de excepción del programa, los bytes 40 a 47 del TDB especificado por TBEGIN contienen la identificación de la excepción de la traducción.

- Lista de accesos controlada o protección de DAT
- Tipo de ASCE
- Traducción de página
- Primera traducción de región
- Segunda traducción de región
- Tercera traducción de región
- Excepción de la traducción del segmento

En un ejemplo de la Arquitectura/z, el formato de la TEID es el mismo que el descrito en las ubicaciones reales 168 a 175 cuando la condición resulta en una interrupción, tal como se describe en los principios de funcionamiento referenciados anteriormente.

Para las transacciones que se cancelan por otros motivos, incluidas las condiciones de excepción que resultan en una interrupción del programa, los bytes 40 a 47 son impredecibles. Los bytes 40 a 47 son impredecibles en el TDB de interrupción del programa.

Este campo se almacena solo en el TDB designado por la dirección del bloque de diagnóstico de la transacción; de lo contrario, el campo está reservado.

Dirección de evento de ruptura 922: para las transacciones que se cancelan debido a condiciones filtradas de excepción del programa, los bytes 48 a 55 del TDB especificado por TBEGIN contienen la dirección del evento de ruptura. En un ejemplo de la Arquitectura/z, el formato de la dirección del evento de ruptura es el mismo que el descrito en las ubicaciones reales 272 a 279 cuando la condición resulta en una interrupción, tal como se describe en los principios de funcionamiento antes referenciados.

Para las transacciones que se cancelan por otros motivos, incluidas las condiciones de excepción que resultan en una interrupción del programa, los bytes 48 a 55 son impredecibles. Los bytes 48 a 55 son impredecibles en el TDB de interrupción del programa.

Este campo se almacena solo en el TDB designado por la dirección del bloque de diagnóstico de la transacción; de lo contrario, el campo está reservado.

A continuación, se describen más detalles relacionados con los eventos de ruptura.

En una realización de la Arquitectura/z, cuando la función PER-3 está instalada, proporciona al programa la dirección de la última instrucción para causar una ruptura en la ejecución secuencial de la CPU. La grabación de la dirección del evento de ruptura se puede utilizar como una asistencia de depuración para la detección de saltos salvajes (wild branch, en inglés). Esta función proporciona, por ejemplo, un registro de 64 bits en la CPU, denominado registro de dirección del evento de ruptura. Cada vez que una instrucción distinta de TRANSACTION ABORT causa una ruptura en la ejecución de la instrucción secuencial (es decir, la dirección de la instrucción en la PSW es reemplazada, en lugar de ser incrementada en la duración de la instrucción), la dirección de esa instrucción se coloca en el registro de la dirección del evento de ruptura. Cuando se produce una interrupción del programa, independientemente de si se indica PER, los contenidos actuales del registro de la dirección del evento de ruptura se colocan en ubicaciones de almacenamiento reales 272 a 279.

Si la instrucción que causa el evento de ruptura es el objetivo de una instrucción del tipo de ejecutar (EXECUTE o EXECUTE RELATIVE LONG), entonces la dirección de la instrucción utilizada para recuperar la instrucción del tipo de ejecución es colocada en el registro de dirección del evento de ruptura.

En una realización de la Arquitectura/z, se considera que ocurre un evento de ruptura siempre que una de las siguientes instrucciones provoque la bifurcación: bifurcación y enlazado, registro de bifurcación y enlazado (BAL - BRANCH AND LINK, BALR - BRANCH AND LINK REGISTER, en inglés); bifurcación y guardado, registro de bifurcación y guardado (BAS - BRANCH AND SAVE, BASR - BRANCH AND SAVE REGISTER, en inglés); modo de bifurcación, guardado y ajuste (BASSM - BRANCH AND SAVE AND SET MODE, en inglés); modo de bifurcación y ajuste (BSM - BRANCH AND SET MODE, en inglés); registro de bifurcación y apilado (BAKR - BRANCH AND STACK REGISTER, en inglés); bifurcación por condición, registro de bifurcación por condición (BC - BRANCH ON CONDITION, BCR - BRANCH ON CONDITION REGISTER, en inglés); bifurcación por recuento, registro de bifurcación por recuento, bifurcación por recuento grande, registro de bifurcación por recuento grande (BCT - BRANCH ON COUNT, BCTR - BRANCH ON COUNT REGISTER, BCTG - BRANCH ON COUNT GRANDE, BCTGR - BRANCH ON COUNT GRANDE REGISTER, en inglés); bifurcación por índice alto, bifurcación por índice alto grande (BXH - BRANCH ON INDEX HIGH, BXHG - BRANCH ON INDEX HIGH GRANDE, en inglés); bifurcación por índice bajo o igual, bifurcación por índice bajo o igual (BXLE - BRANCH ON INDEX LOW OR EQUAL, BXLEG - BRANCH ON INDEX LOW OR EQUAL GRANDE, en inglés); bifurcación relativa por condición (BRC - BRANCH RELATIVE ON CONDITION, en inglés); bifurcación relativa por condición larga (BRCL - BRANCH RELATIVE ON CONDITION LONG, en inglés); bifurcación relativa por recuento, bifurcación relativa por recuento grande (BRCT - BRANCH RELATIVE ON COUNT, BRCTG - BRANCH RELATIVE ON COUNT GRANDE, en inglés); bifurcación relativa por índice alto, bifurcación relativa por índice alto grande (BRXH - BRANCH RELATIVE ON INDEX HIGH, BRXHG - BRANCH RELATIVE ON INDEX HIGH GRANDE, en inglés); bifurcación relativa por índice bajo o igual, bifurcación relativa por índice bajo o igual grande (BRXLE - BRANCH RELATIVE ON INDEX LOW OR EQUAL, BRXLG - BRANCH RELATIVE ON INDEX LOW OR EQUAL GRANDE, en inglés); comparar y bifurcar (CRB, CGRB - COMPARE AND BRANCH, en inglés); comparación y bifurcación relativa (CRJ, CGRJ - COMPARE AND BRANCH RELATIVE, en inglés); comparación inmediata y bifurcación (CIB, CGIB - COMPARE IMMEDIATE AND BRANCH, en inglés); comparación inmediata y bifurcación relativa (CIJ, CGIJ - COMPARE IMMEDIATE AND BRANCH RELATIVE, en inglés); comparación de lógica y bifurcación (CLRB, CLGRB - COMPARE LOGICAL AND BRANCH, en inglés); comparación de lógica y bifurcación relativa (CLRJ, CLGRJ - COMPARE LOGICAL AND BRANCH RELATIVE, en inglés); comparación inmediata de lógica y bifurcación (CLIB, CLGIB - COMPARE LOGICAL IMMEDIATE AND BRANCH, en inglés); y comparación inmediata de lógica y bifurcación relativa (CLIJ, CLGIJ - COMPARE LOGICAL IMMEDIATE AND BRANCH RELATIVE, en inglés).

Asimismo, se considera que ocurre un evento de ruptura cuando se completa una de las siguientes instrucciones: autoridad de bifurcación y ajuste (BSA - BRANCH AND SET AUTHORITY, en inglés); bifurcación en grupo de subespacio (BSG - BRANCH IN SUBSPACE GROUP, en inglés); bifurcación relativa y guardado (BRAS – BRANCH RELATIVE AND SAVE, en inglés); bifurcación relativa y guardado largo (BRASL - BRANCH RELATIVE AND SAVE

LARGO, en inglés); PSW de carga (LPSW – LOAD PSW, en inglés); PSW de carga extendida (LPSWE - LOAD PSW EXTENDED, en inglés); llamada al programa (PC - PROGRAM CALL, en inglés); retorno del programa (PR – PROGRAM RETURN, en inglés); transferencia de programa (PT – PROGRAM TRANSFER, en inglés); transferencia de programa con instancia (PTI – PROGRAM TRANSFER WITH INSTANCE, en inglés); reanudación de programa (RP - RESUME PROGRAM, en inglés); y TRAP (TRAP2, TRAP4).

No se considera que ocurre un evento de ruptura como resultado de una transacción que se ha cancelado (implícitamente o como resultado de la instrucción TRANSACTION ABORT).

Información de diagnóstico que depende del modelo 924: los bytes 112 a 127 contienen información de diagnóstico que depende del modelo.

Para todos los códigos de cancelación excepto 12 (interrupción filtrada del programa), la información de diagnóstico que depende del modelo está guardada en cada TDB que se almacena.

En una realización, la información de diagnóstico que depende del modelo incluye lo siguiente:

- Los bytes 112 a 119 contienen un vector de 64 bits denominado indicaciones de bifurcación de ejecución de transacción (TXBI – Transactional eXecution Branch Indications, en inglés). Cada uno de los primeros 63 bits del vector indica los resultados de la ejecución de una instrucción de bifurcación mientras la CPU estaba en el modo de ejecución de transacción, como sigue:

Significado del Valor

0 La instrucción se completó sin bifurcación.

1 La instrucción se completó con bifurcación.

El bit 0 representa el resultado de la primera de dichas instrucciones de bifurcación de este tipo, el bit 1 representa el resultado de la segunda de dichas instrucciones de este tipo, y así sucesivamente.

Si se ejecutaron menos de 63 instrucciones de bifurcación mientras la CPU estaba en el modo de ejecución de transacción, los bits de más a la derecha que no corresponden a las instrucciones de bifurcación se configuran en ceros (incluido el bit 63). Cuando se ejecutaron más de 63 instrucciones de bifurcación, el bit 63 de la TXBI se configura en uno.

Los bits en la TXBI se configuran mediante instrucciones que son capaces de provocar un evento de ruptura, tal como se indicó anteriormente, excepto por lo siguiente:

- Cualquier instrucción restringida no provoca que se ajuste un bit en la TXBI.

- Para instrucciones, por ejemplo, de la Arquitectura/z, cuando el campo M_1 de la instrucción BRANCH ON CONDITION, BRANCH RELATIVE ON CONDITION, o BRANCH RELATIVE ON CONDITION LONG es cero, o cuando el campo R_2 de las siguientes instrucciones es cero, es que depende del modelo si la ejecución de la instrucción hace que se establezca un bit en la TXBI.

- BRANCH AND LINK (BALR); BRANCH AND SAVE (BASR); BRANCH AND SAVE AND SET MODE (BASSM); BRANCH AND SET MODE (BSM); BRANCH ON CONDITION (BCR); y BRANCH ON COUNT (BCTR, BCTGR)

- Para las condiciones de cancelación causadas por una excepción del acceso al anfitrión, la posición del bit 0 del byte 127 se configura en uno. Para todas las demás condiciones de cancelación, la posición del bit 0 del byte 127 se configura en cero.

- Para condiciones de cancelación que fueron detectadas por la unidad de carga / almacenamiento (LSU – Load / Store Unit, en inglés), los bits de más a la derecha del byte 127 contienen una indicación de la causa. Para condiciones de cancelación que no fueron detectadas por la LSU, se reserva el byte 127.

Registros generales 930: los bytes 128 a 255 contienen los contenidos de los registros generales 0 a 15 en el momento en que se canceló la transacción. Los registros están almacenados en orden ascendente, empezando con el registro general 0 en los bytes 128 a 135, el registro general 1 en los bytes 136 a 143, y así sucesivamente.

Reservado: Todos los demás campos están reservados. A menos que se indique lo contrario, los contenidos de los campos reservados son impredecibles.

Según lo observado por otras CPU y por el subsistema I/O, el almacenamiento del TDB o los TDB durante una cancelación de transacción es una referencia de acceso múltiple que ocurre después de cualquier almacenamiento no de transacción.

Una transacción puede ser cancelada debido a causas que están fuera del alcance de la configuración inmediata en la que se ejecuta. Por ejemplo, los eventos transitorios reconocidos por un hipervisor (Tal como LPAR o z/VM) pueden provocar la cancelación de una transacción.

La información proporcionada en el bloque de diagnóstico de la transacción está destinada a fines de diagnóstico y es sustancialmente correcta. No obstante, debido a que una cancelación puede haber sido causada por un evento fuera del alcance de la configuración inmediata, la información tal como el código de cancelación o la identificación de interrupción del programa puede no reflejar con precisión las condiciones dentro de la configuración y, por lo tanto, no debe ser utilizada para determinar la acción del programa.

Además de la información de diagnóstico guardada en el TDB, cuando una transacción es cancelada debido a cualquier condición de excepción de programa de excepción de datos y tanto el control de registro AFP, bit 45 del registro de control 0, como el efectivo permiten el control de operación de coma flotante (F) son uno, el código de excepción de datos (DXC) es colocado en el byte 2 del registro de control de coma flotante (FPCR), independientemente de si el filtrado se aplica a la condición de excepción del programa. Cuando una transacción es cancelada, y ya sea el control del registro AFP o el control de operación de coma flotante de permiso efectivo, o ambos, es cero, el DXC no está colocado en el FPCR.

En una realización, tal como se indica en el presente documento, cuando la función de ejecución de transacción está instalada, se proporcionan las siguientes instrucciones generales.

- EXTRAER PROFUNDIDAD DE ANIDAMIENTO DE LA TRANSACCIÓN
- ALMACENAMIENTO NO DE TRANSACCIÓN
- CANCELACIÓN DE LA TRANSACCIÓN
- INICIO DE LA TRANSACCIÓN
- FIN DE LA TRANSACCIÓN

Cuando la CPU está en el modo de ejecución de transacción, el intento de ejecución de ciertas instrucciones está restringido y causa la cancelación de la transacción.

Cuando se emite en el modo restringido de ejecución de transacción, el intento de ejecución de instrucciones restringidas puede resultar asimismo en una interrupción del programa de restricción de transacción, o puede resultar en la ejecución de procedimientos como si la transacción no estuviera restringida.

En un ejemplo de la Arquitectura/z, las instrucciones restringidas incluyen, como ejemplos, las siguientes instrucciones sin privilegios: COMPARAR E INTERCAMBIAR Y ALMACENAR; MODIFICAR CONTROLES DE INSTRUMENTACIÓN DE TIEMPO DE EJECUCIÓN; REALIZAR OPERACIÓN DE BLOQUEO; DATOS DE BÚSQUEDA ADELANTADA (RELATIVA LARGA), cuando el código en el campo M_1 es 6 o 7; ALMACENAR LOS CARACTERES BAJO MÁSCARA ALTA, cuando el campo M_3 es cero y el código en el campo R_1 es 6 o 7; ALMACENAR LISTA DE FUNCIONES EXTENDIDA; ALMACENAR CONTROLES DE INSTRUMENTACIÓN DEL TIEMPO DE EJECUCIÓN; LLAMADA DEL SUPERVISOR; y CONTROLES DE INSTRUMENTACIÓN DEL TIEMPO DE EJECUCIÓN DE LA PRUEBA.

En la lista anterior, COMPARAR E INTERCAMBIAR Y ALMACENAR y REALIZAR OPERACIÓN BLOQUEADA son instrucciones complejas que pueden ser implementadas de manera más eficiente mediante la utilización de instrucciones básicas en el modo TX. Los casos de BÚSQUEDA ADELANTADA DE DATOS y BÚSQUEDA ADELANTADA DE DATOS RELATIVA LARGA están restringidos, ya que los códigos de 6 y 7 liberan una línea de la memoria caché, que exige el compromiso de los datos potencialmente antes de la finalización de una transacción. La LLAMADA DEL SUPERVISOR está restringida, ya que causa una interrupción (lo que hace que una transacción se cancele).

En las condiciones enumeradas a continuación, las siguientes instrucciones están restringidas:

- BIFURCACIÓN Y ENLAZADO (BALR), BIFURCACIÓN Y GUARDADO (BASR) y MODO DE BIFURCACIÓN Y GUARDADO Y AJUSTE, cuando el campo R_2 de la instrucción es distinto de cero y el seguimiento de la bifurcación está habilitado.

- MODO DE BIFURCACIÓN, GUARDADO Y AJUSTE y MODO DE BIFURCACIÓN Y AJUSTE, cuando el campo R_2 es distinto de cero y el seguimiento de modo está habilitado; MODO DE AJUSTE DEL DIRECCIONAMIENTO, cuando el seguimiento de modo está habilitado.

- MONITORIZAR LLAMADA, cuando se reconoce una condición de evento de monitorización.

La lista anterior incluye instrucciones que pueden formar entradas de rastreo. Si se permitiera la ejecución de estas instrucciones a modo de transacción y la formación de entradas de rastreo, y la transacción posteriormente fuese cancelada, se avanzaría el puntero de la tabla de rastreo en el registro de control 12, pero los almacenamientos en la tabla de rastreo serían descartados. Esto dejaría un vacío incoherente en la tabla de seguimiento. De este modo, las instrucciones están restringidas en los casos en que formarían entradas de rastreo.

Cuando la CPU está en el modo de ejecución de transacción, depende del modelo si las siguientes instrucciones están restringidas: CIFRAR MENSAJE; CIFRAR MENSAJE CON CFB; CIFRAR MENSAJE CON ENSECUENCIAMIENTO; CIFRAR MENSAJE CON CONTADOR; CIFRAR MENSAJE CON OFB; LLAMADA DE COMPRESIÓN; CALCULAR RESUMEN DE MENSAJES INTERMEDIOS; CALCULAR RESUMEN DEL ÚLTIMO MENSAJE; CALCULAR CÓDIGO DE AUTENTICACIÓN DE MENSAJE; CONVERTIR UNICODE-16 A UNICODE-32; CONVERTIR UNICODE-16 A UNICODE-8; CONVERTIR UNICODE-32 A UNICODE-16; CONVERTIR UNICODE-32 A UNICODE-8; CONVERTIR UNICODE-8 A UNICODE-16; CONVERTIR UNICODE-8 A UNICODE-32; REALIZAR EL CÁLCULO CRIPTOGRÁFICO; INSTRUMENTACIÓN DE TIEMPO DE EJECUCIÓN DESCONECTADA; e INSTRUMENTACIÓN DE TIEMPO DE EJECUCIÓN CONECTADA.

Cada una de las instrucciones anteriores está actualmente implementada por el coprocesador de hardware, o lo ha estado en máquinas anteriores, y, por lo tanto, se considera restringida.

Cuando el control efectivo de modificación de AR (A) es cero, las siguientes instrucciones están restringidas: COPIAR ACCESO; CARGAR ACCESO MÚLTIPLE; CARGAR DIRECCIÓN EXTENDIDA; y CONFIGURAR ACCESO.

Cada una de las instrucciones anteriores hace que se modifiquen los contenidos de un registro de acceso. Si el control A en la instrucción TRANSACTION BEGIN es cero, entonces el programa ha indicado explícitamente que no se debe permitir la modificación del registro de acceso.

Cuando el control efectivo de la operación en coma flotante (F) es cero, las instrucciones de coma flotante están restringidas.

Bajo ciertas circunstancias, las siguientes instrucciones pueden estar restringidas: EXTRAER TIEMPO DE LA CPU; EXTRAER PSW; ALMACENAR RELOJ; ALMACENAR RELOJ EXTENDIDO; y ALMACENAR RELOJ RÁPIDO.

Cada una de las instrucciones anteriores está sujeta a un control de interceptación en la descripción del estado de ejecución interpretativa. Si el hipervisor ha configurado el control de interceptación para estas instrucciones, entonces su ejecución se puede prolongar debido a la implementación del hipervisor; por lo tanto, se consideran restringidas si se produce una interceptación.

Cuando una transacción no restringida se cancela debido al intento de ejecución de una instrucción restringida, el código de cancelación de la transacción en el bloque de diagnóstico de la transacción se configura en 11 (instrucción restringida) y el código de condición se configura en 3, excepto de la siguiente manera: cuando se cancela una transacción no restringida debido al intento de ejecución de una instrucción que de lo contrario daría lugar a una excepción de operación privilegiada, es impredecible si el código de cancelación se configura en 11 (instrucción restringida) o a 4 (interrupción del programa no filtrado resultante del reconocimiento de la interrupción del programa de la operación privilegiada). Cuando una transacción no restringida se cancela debido al intento de ejecución de PREFETCH DATA (RELATIVE LONG) cuando el código en el campo M_1 es 6 o 7 o STORE CHARACTERS UNDER MASK HIGH cuando el campo M_3 es cero y el código en el campo R_1 es 6 o 7, es impredecible si el código de cancelación se configura en 11 (instrucción restringida) o a 16 (en la otra memoria caché). Cuando una transacción no restringida se cancela debido al intento de ejecución de MONITOR CALL, y están presentes una condición de evento de monitorización y una condición de excepción de especificación, es impredecible si el código de cancelación se configura en 11 o a 4, o si se la interrupción del programa está filtrada, 12.

Las instrucciones adicionales pueden estar restringidas en una transacción restringida. Aunque estas instrucciones no están definidas actualmente como restringidas en una transacción no restringida, pueden estar restringidas en ciertas circunstancias en una transacción no restringida en procesadores futuros.

Se pueden permitir ciertas instrucciones restringidas en el modo de ejecución de transacción en futuros procesadores. Por lo tanto, el programa no debe estar basado en que la transacción se cancele debido al intento de ejecución de una instrucción restringida. La instrucción TRANSACTION ABORT se debe utilizar para cancelar una transacción de manera fiable.

En una transacción no restringida, el programa debe proporcionar una ruta de código no de transacción alternativa para soportar una transacción que es cancelada debido a una instrucción restringida.

En funcionamiento, cuando la profundidad de anidamiento de la transacción es cero, la ejecución de la instrucción TRANSACTION BEGIN (TBEGIN) que resulta en el código de condición cero hace que la CPU entre en el modo no restringido de ejecución de transacción. Cuando la profundidad de anidamiento de la transacción es cero, la

ejecución de la instrucción restringida TRANSACTION BEGIN (TBEGINC) que resulta en el código de condición cero hace que la CPU entre en el modo restringido de ejecución de transacción.

Excepto cuando se indique explícitamente lo contrario, todas las reglas que se aplican para la ejecución no de transacción también se aplican a la ejecución de transacción. A continuación, se presentan otras características de procesamiento mientras la CPU está en el modo de ejecución de transacción.

Cuando la CPU está en el modo no restringido de ejecución de transacción, la ejecución de la instrucción TRANSACTION BEGIN que resulta en el código de condición cero hace que la CPU permanezca en el modo no restringido de ejecución de transacción.

Según lo observado por la CPU, las búsquedas y los almacenamientos realizados en el modo de ejecución de la transacción no son diferentes a los realizados mientras no se está en el modo de ejecución de transacción. Según lo observado por otras CPU y por el subsistema I/O, todos los accesos de operandos de almacenamiento realizados mientras una CPU está en el modo de ejecución de transacción parecen ser un acceso simultáneo de bloque único. Es decir, los accesos a todos los bytes dentro de una media palabra, palabra, palabra doble o palabra cuádruple se especifican para que parezcan bloques simultáneos, según lo observado por otras CPU y por programas de I/O (por ejemplo, canales). La media palabra, palabra, doble palabra o palabra cuádruple se denomina bloque, en esta sección. Cuando se especifica que una referencia del tipo de búsqueda es simultánea dentro de un bloque, no se permite el acceso del almacenamiento al bloque por parte de otra CPU o programa de I/O durante el tiempo en que se buscan los bytes contenidos en el bloque. Cuando se especifica que una referencia del tipo de almacenamiento parece ser simultánea dentro de un bloque, ninguna otra CPU o programa de I/O está autorizada a acceder al bloque, ya sea para buscar o almacenar, durante el tiempo en que se almacenan los bytes dentro del bloque.

Los accesos de almacenamiento para la instrucción y las búsquedas de tablas de DAT y ART (Access Register Table) siguen las reglas no de transacción. La CPU sale normalmente del modo de ejecución de transacción por medio de una instrucción FINALIZACIÓN DE TRANSACCIÓN que hace que la profundidad de anidamiento de la transacción pase a cero, en cuyo caso, la transacción se completa.

Cuando la CPU abandona el modo de ejecución de transacción mediante la finalización de una instrucción TRANSACTION END, todos los almacenamientos realizadas en el modo de ejecución de transacción se comprometen; es decir, los almacenamientos parecen producirse como una operación simultánea de bloque único tal como la observada por otras CPU y por el subsistema I/O.

Una transacción puede ser cancelada implícitamente por una variedad de causas, o puede ser cancelada explícitamente por la instrucción TRANSACTION CANCELATION. Ejemplo de posibles causas de una cancelación de transacción, del código de cancelación correspondiente y del código de condición que se coloca en la PSW de cancelación de transacción, se describen a continuación.

Interrupción externa: el código de cancelación de transacción se configura en 2, y el código de condición en la PSW de cancelación de la transacción se configura en 2. La PSW de cancelación de transacción se almacena como la PSW anterior externa como parte del procesamiento de interrupción externa.

Interrupción del programa (no filtrado): Una condición de excepción del programa que resulta en una interrupción (es decir, una condición no filtrada) hace que la transacción se cancele con el código 4. El código de condición en la PSW de cancelación de transacción se configura específicamente para el código de interrupción del programa. La PSW de cancelación de transacción se almacena como el programa viejo PSW como parte del procesamiento de interrupción del programa.

Una instrucción que de lo contrario resultaría en la cancelación de una transacción debido a una excepción de operación puede arrojar resultados alternativos: para una transacción no restringida, la transacción puede ser cancelada con el código de cancelación 11 (instrucción restringida); para una transacción restringida, se puede reconocer una interrupción del programa de restricción de transacción en lugar de la excepción de operación.

Cuando se reconoce un evento PER (Grabación de eventos de programa) junto con cualquier otra condición de excepción de programa no filtrada, el código de condición se configura en 3.

Interrupción de verificación de la máquina: el código de cancelación de transacción se configura en 5 y el código de condición en la PSW de cancelación de la transacción se configura en 2. La PSW de cancelación de la transacción se almacena cuando la máquina verifica la PSW anterior como parte del procesamiento de interrupción de verificación de la máquina.

Interrupción de I/O: el código de cancelación de la transacción se configura en 6 y el código de condición en la PSW de cancelación de la transacción se configura en 2. La PSW de cancelación de la transacción se almacena como la PSW anterior de I/O como parte del proceso de interrupción de I/O.

Anulación de la búsqueda: se detecta una condición de desbordamiento de búsqueda cuando la transacción intenta realizar una búsqueda desde más ubicaciones de las que soporta la CPU. El código de cancelación de la transacción se configura en 7, y el código de condición se configura en 2 o 3.

Anulación del almacenamiento: se detecta una condición de anulación del almacenamiento cuando la transacción intenta almacenar en más ubicaciones de las que soporta la CPU. El código de cancelación de transacción se configura en 8, y el código de condición se configura en 2 o 3.

Permitir que el código de condición sea 2 o 3 en respuesta a una anulación de la búsqueda o el almacenamiento permite a la CPU indicar condiciones potencialmente recuperables (por ejemplo, el código de condición 2 indica que la re ejecución de la transacción puede ser productiva, mientras que el código de condición 3 no recomienda re-ejecución).

Conflicto de búsqueda: una condición de conflicto de búsqueda se detecta cuando otra CPU o subsistema I/O intenta realizar un almacenamiento en una ubicación que se ha realizado una búsqueda de transacción por parte de esta CPU. El código de cancelación de transacción se configura en 9 y el código de condición se configura en 2.

Conflicto de almacenamiento: una condición de conflicto de almacenamiento se detecta cuando otra CPU o el sistema I/O intenta acceder a una ubicación que se ha almacenado durante la transacción de esta CPU. El código de cancelación de la transacción se configura en 10 y el código de condición se configura en 2.

Instrucción restringida: cuando la CPU se encuentra en el modo de ejecución de transacción, un intento de ejecución de una instrucción restringida hace que se cancele la transacción. El código de cancelación de la transacción se configura en 11 y el código de condición se configura en 3.

Cuando la CPU está en el modo restringido de ejecución de transacción, es impredecible si el intento de ejecución de una instrucción restringida resulta en una interrupción del programa de restricción de la transacción o una cancelación debido a una instrucción restringida. La transacción aún se cancela, pero el código de cancelación puede indicar cualquier causa.

Condición de excepción de programa (filtrada): Una condición de excepción de programa que no resulta en una interrupción (es decir, una condición filtrada) hace que la transacción se cancele con un código de cancelación de transacción de 12. El código de condición se configura en 3.

Profundidad de anidamiento excedida: la condición excedida de la profundidad de anidamiento se detecta cuando la profundidad de anidamiento de la transacción está en el valor máximo permitido para la configuración, y se ejecuta una instrucción TRANSACTION BEGIN. La transacción se cancela con un código de cancelación de transacción de 13, y el código de condición se configura en 3.

Condición relacionada con la búsqueda en la memoria caché: los circuitos de la memoria caché de la CPU detectan una condición relacionada con las ubicaciones de almacenamiento captadas por la transacción. La transacción se cancela con un código de cancelación de transacción de 14, y el código de condición se configura en 2 o 3.

Condición relacionada con el almacenamiento en la memoria caché: los circuitos de la memoria caché de la CPU detectan una condición relacionada con las ubicaciones de almacenamiento almacenadas por la transacción. La transacción se cancela con un código de cancelación de transacción de 15, y el código de condición se configura en 2 o 3.

Condición adicional de la memoria caché: los circuitos de la memoria caché de la CPU detectan una condición adicional de la memoria caché. La transacción se cancela con un código de cancelación de transacción de 16, y el código de condición se configura en 2 o 3.

Durante la ejecución de transacción, si la CPU accede a instrucciones u operandos de almacenamiento utilizando diferentes direcciones lógicas asignadas a la misma dirección absoluta, depende del modelo si la transacción se interrumpe. Si la transacción se cancela debido a que los accesos utilizan direcciones lógicas diferentes asignadas a la misma dirección absoluta, se configura el código de cancelación 14, 15 o 16, dependiendo de la condición.

Condición miscelánea: Una condición miscelánea es cualquier otra condición reconocida por la CPU que causa la cancelación de la transacción. El código de cancelación de la transacción se configura en 255 y el código de condición se configura en 2 o 3.

Cuando se ejecutan varias configuraciones en la misma máquina (por ejemplo, particiones lógicas o máquinas virtuales), una transacción puede ser cancelada debido a una verificación de máquina externa o a una interrupción de I/O que ocurrió en una configuración diferente.

Aunque anteriormente se han proporcionado ejemplos, se pueden proporcionar otras causas de una cancelación de transacción con códigos de cancelación y códigos de condición correspondientes. Por ejemplo, una causa puede ser una interrupción del reinicio, en la que el código de cancelación de la transacción se configura en 1, y el código de

condición en la PSW de cancelación de la transacción se configura en 2. La PSW de cancelación de la transacción se almacena como la PSW de reinicio antigua como una parte del proceso de reinicio. Como un ejemplo adicional, una causa puede ser una condición de llamada del supervisor, en la que el código de cancelación se configura en 3, y el código de condición en la PSW de cancelación de la transacción se configura en 3. Asimismo. son posibles otros ejemplos diferentes.

Notas:

1. La condición miscelánea puede resultar de cualquiera de lo siguiente:

- Instrucciones, tales como, en la Arquitectura/z, COMPARAR Y REEMPLAZAR ENTRADA DE LA TABLA DE DAT, COMPARAR E INTERCAMBIAR Y PURGAR, INVALIDAR ENTRADA DE LA TABLA DE DAT, INVALIDAR ENTRADA DE LA PÁGINA DE PÁGINA, REALIZAR LA FUNCIÓN DE ADMINISTRACIÓN DE TRAMAS en la que el control NQ es cero y el control SK es uno, DE ALMACENAMIENTO EXTENDIDA en la que el control NQ es cero, realizada por otra CPU en la configuración; el código de condición se configura en 2.

- Una función de operador, tal como configurar, reiniciar o detener, o la orden SIGNAL PROCESSOR equivalente se realiza en la CPU.

- Cualquier otra condición no enumerada arriba; el código de condición se configura en 2 o 3.

2. La ubicación en la que se detectan los conflictos de búsqueda y almacenamiento puede estar en cualquier lugar dentro de la misma línea de la memoria caché.

3. Bajo ciertas condiciones, la CPU puede no ser capaz de distinguir entre condiciones similares de cancelación. Por ejemplo, una anulación de búsqueda o almacenamiento puede ser indistinguible de un conflicto de búsqueda o almacenamiento respectivo.

4. La ejecución especulativa de múltiples rutas de instrucción por parte de la CPU puede ocasionar que la transacción se cancele debido a condiciones de conflicto o de anulación, incluso si dichas condiciones no ocurren en la secuencia conceptual. Mientras esté en el modo restringido de ejecución de transacción, la CPU puede inhibir temporalmente la ejecución especulativa, lo que permite que se intente completar la transacción sin detectar dichos conflictos o anulaciones especulativos.

La ejecución de una instrucción TRANSACTION ABORT hace que la transacción se cancele. El código de cancelación de la transacción se configura a partir de la dirección del segundo operando. El código de condición se configura en 2 o 3, dependiendo de si el bit 63 de la dirección del segundo operando es cero o uno, respectivamente.

La figura 10 resume los códigos de cancelación de ejemplo almacenados en un bloque de diagnóstico de transacción, y el código de condición correspondiente (CC). La descripción en la figura 10 ilustra una implementación particular. Son posibles otras implementaciones y codificaciones de valores.

En una realización tal como la mencionada anteriormente, la función de transacción proporciona tanto transacciones restringidas como transacciones no restringidas, así como el procesamiento asociado con las mismas. Inicialmente, se explican las transacciones restringidas y, a continuación, las transacciones no restringidas.

Una transacción restringida se ejecuta en modo de transacción sin una ruta de retroceso. Es un modo de procesamiento útil para funciones compactas. En ausencia de interrupciones o conflictos repetidos con otras CPU o con el subsistema I/O (es decir, causados por condiciones que no permitirán que la transacción se complete con éxito), una transacción restringida, finalmente, se completará; por lo tanto, no se requiere una rutina de controlador de cancelación y no se especifica. Por ejemplo, en ausencia de una violación de una condición que no puede ser solucionada (por ejemplo, dividir por 0); una condición que no permite completar la transacción (por ejemplo, una interrupción del temporizador que no permite que se ejecute una instrucción, una I/O en caliente; etc.); o una violación de una limitación (restriction, en inglés) o restricción (constraint, en inglés) asociada con una transacción restringida, la transacción, finalmente, se completará.

Una transacción restringida se inicia mediante una instrucción restringida TRANSACTION BEGIN (TBEGINC) cuando la profundidad de anidamiento de la transacción es inicialmente cero. Una transacción restringida está sujeta a las siguientes restricciones, en una realización.

1. La transacción ejecuta no más de 32 instrucciones, sin incluir las instrucciones TRANSACCIÓN BEGIN restringida (TBEGINC) y TRANSACTION END.

2. Todas las instrucciones en el arco de la transacción deben estar dentro de 256 bytes contiguos de almacenamiento, incluyendo la instrucción TRANSACCIÓN BEGIN restringida (TBEGINC) y cualquier instrucción TRANSACTION END.

3. Además de las instrucciones restringidas, a una transacción restringida se le aplican las siguientes restricciones.

   a. Las instrucciones se limitan a las denominadas Instrucciones generales, que incluyen, por ejemplo, sumar, restar, multiplicar, dividir, desplazar, rotar, etc.

   b. Las instrucciones de bifurcación están limitadas a lo siguiente (las instrucciones enumeradas son de la Arquitectura/z en un ejemplo):

      - BRANCH RELATIVE ON CONDITION, en la que M_1 es distinto de cero y el campo RI_2 contiene un valor positivo.
      - BRANCH RELATIVE ON CONDITION LONG en la que el campo M_1 es distinto de cero, y el campo RI_2 contiene un valor positivo que no causa una dirección envolvente.
      - COMPARE AND BRANCH RELATIVE, COMPARE IMMEDIATE AND BRANCH RELATIVE, COMPARE LOGICAL AND BRANCH RELATIVE, y COMPARE LOGICAL IMMEDIATE AND BRANCH RELATIVE, en la que el campo M_3 es distinto de cero y el campo RI_4 contiene un valor positivo. (Es decir, solo bifurcaciones hacia adelante con máscaras de bifurcación distintas de cero.)

   c. Excepto por el TRANSACTION END y las instrucciones que causan una serialización específica del operando, las instrucciones que causan una función de serialización están restringidas.

   d. Las instrucciones de almacenamiento y almacenamiento de operaciones (SS- - Storage and Storage operations, en inglés) y almacenamiento y almacenamiento de operaciones con código de operación extendido (SSE- - Storage and Storage operations with an Extended opcode, en inglés) están restringidas.

   e. Todas las siguientes instrucciones generales (que son de la Arquitectura/z en este ejemplo) están restringidas: CHECKSUM; CIPHER MESSAGE; CIPHER MESSAGE WITH CFB; CIPHER MESSAGE WITH CHAINING; CIPHER MESSAGE WITH COUNTER; CIPHER MESSAGE WITH OFB; COMPARE AND FORM CODEWORD; COMPARE LOGICAL LONG; COMPARE LOGICAL LONG EXTENDED; COMPARE LOGICAL LONG UNICODE; COMPARE LOGICAL STRING; COMPARE UNTIL SUBSTRING EQUAL; COMPRESSION CALL; COMPUTE INTERMEDIATE MESSAGE DIGEST; COMPUTE LAST MESSAGE DIGEST; COMPUTE MESSAGE AUTHENTICATION CODE; CONVERT TO BINARY; CONVERT TO DECIMAL; CONVERT UNICODE-16 TO UNICODE-32; CONVERT UNICODE-16 TO UNICODE-8; CONVERT UNICODE-32 TO UNICODE-16; CONVERT UNICODE-32 TO UNICODE-8; CONVERT UNICODE-8 TO UNICODE-16; CONVERT UNICODE-8 TO UNICODE-32; DIVIDE; DIVIDE LOGICAL; DIVIDE SINGLE; EXECUTE; EXECUTE RELATIVE LONG; EXTRACT CACHE ATTRIBUTE; EXTRACT CPU TIME; EXTRACT PSW; EXTRACT TRANSACTION NESTING DEPTH; LOAD AND ADD; LOAD AND ADD LOGICAL; LOAD AND; LOAD AND EXCLUSIVE OR; LOAD AND OR; LOAD PAIR DISJOINT; LOAD PAIR FROM QUADWORD; MONITOR CALL; MOVE LONG; MOVE LONG EXTENDED; MOVE LONG UNICODE; MOVE STRING; NON-TRANSACTIONAL STORE; PERFORM CRYPTOGRAPHIC COMPUTATION; PREFETCH DATA; PREFETCH DATA RELATIVE LONG; RUNTIME INSTRUMENTATION EMIT; RUNTIME INSTRUMENTATION NEXT; RUNTIME INSTRUMENTATION OFF; RUNTIME INSTRUMENTATION ON; SEARCH STRING; SEARCH; STRING UNICODE; SET ADDRESSING MODE; STORE CHARACTERS UNDER MASK HIGH, cuando el campo M_3 es cero y el código en el campo R_1 es 6 o 7; STORE CLOCK; STORE CLOCK EXTENDED; STORE CLOCK FAST; STORE FACILITY LIST EXTENDED; STORE PAIR TO QUADWORD; TEST ADDRESSING MODE; TRANSACTION ABORT; TRANSACTION BEGIN (tanto TBEGIN como TBEGINC); TRANSLATE AND TEST EXTENDED; TRANSLATE AND TEST REVERSE EXTENDED; TRANSLATE EXTENDED; TRANSLATE ONE TO ONE; TRANSLATE ONE TO TWO TRANSLATE TWO TO ONE; y TRANSLATE TWO TO TWO.

4. Los operandos de almacenamiento de la transacción tienen acceso a no más de cuatro palabras óptuples. Nota: LOAD ON CONDITION y STORE ON CONDITION se consideran referencia de almacenamiento independientemente del código de condición. Una palabra óctuple es, por ejemplo, un grupo de 32 bytes consecutivos en un límite de 32 bytes.

5. La transacción ejecutada en esta CPU, o almacenada por otras CPU o por el subsistema I/O, no accede a los operandos de almacenamiento en ningún bloque de 4 K-byte que contenga los 256 bytes de almacenamiento que empiezan con la instrucción restringida TRANSACTION BEGIN (TBEGINC).

6. La transacción no accede a las instrucciones ni a los operandos de almacenamiento que utilizan direcciones lógicas diferentes que están asignadas a la misma dirección absoluta.

7. Las referencias de operandos realizadas por la transacción deben estar dentro de una sola palabra doble, excepto para LOAD ACCESS MULTIPLE, LOAD MULTIPLE, LOAD MULTIPLE HIGH, STORE ACCESS

MULTIPLE, STORE MULTIPLE, y STORE MULTIPLE HIGH, en las que las referencias de operandos deben estar dentro de una única palabra óctuple.

Si una transacción restringida viola cualquiera de las restricciones 1 a 7, enumeradas anteriormente, entonces (a) se reconoce una interrupción del programa de transacción restringida, o (b) la ejecución continúa como si la transacción no estuviera restringida, excepto por que aún pueden producirse otras violaciones de restricciones en una interrupción de programa de transacción restringida. Es impredecible qué acción se toma, y la acción tomada puede diferir en función de qué restricción se viola.

En ausencia de violaciones de restricciones, interrupciones repetidas o conflictos con otras CPU o con el subsistema I/O, una transacción restringida, finalmente, se completará, tal como se describió anteriormente.

1. La posibilidad de completar con éxito una transacción restringida mejora si la transacción cumple los siguientes criterios:

    a. Las instrucciones emitidas son menos del máximo de 32.

    b. Las referencias del operando de almacenamiento son menos del máximo de 4 palabras óptuples.

    c. Las referencias del operando de almacenamiento están en la misma línea de la memoria caché.

    d. Las referencias del operando de almacenamiento a las mismas ubicaciones ocurren en el mismo orden mediante todas las transacciones.

2. No se garantiza que una transacción restringida se complete con éxito en su primera ejecución. No obstante, si se interrumpe una transacción restringida que no viola ninguna de las restricciones enumeradas, la CPU emplea circuitos para garantizar que una nueva ejecución de la transacción sea posteriormente satisfactoria.

3. Dentro de una transacción restringida, TRANSACTION BEGIN es una instrucción restringida, por lo tanto, no se puede anidar una transacción restringida.

4. La violación de cualquiera de las restricciones 1 a 7 anteriores por una transacción restringida puede resultar en un bucle del programa.

5. Las limitaciones de una transacción restringida son similares a las de un bucle de comparación e intercambio. Debido a la interferencia potencial de otras CPU y del subsistema I/O, no existe ninguna garantía arquitectónica de que la instrucción COMPARE AND SWAP se completen con el código de condición 0. Una transacción restringida puede sufrir interferencias similares en la forma de cancelaciones por conflicto de búsqueda o almacenamiento o interrupciones en caliente.

    La CPU emplea algoritmos de imparcialidad para garantizar que, en ausencia de cualquier restricción, una transacción restringida, finalmente, se complete.

6. Para determinar el número de repeticiones de iteraciones necesarias para completar una transacción restringida, el programa puede emplear un contador en un registro general que no está sujeto a la máscara de guardar registro general. A continuación, se muestra un ejemplo.

	LHI	15,0	Contador de reintentos de cero
Bucle	TBEGINC	0(0),X ‘FE00’	Mantener GR 0 a 13
	AHI	15,1	Incrementar contador
	,,,		
	,,,	Código de ejecución de transacción restringida	
	,,,		
	TEND		Fin de transacción
* R15 contiene ahora el recuento de intentos de transacción repetidos.			

Se debe observar que, en este ejemplo, ambos registros 14 y 15 no son restaurados. Se debe observar asimismo que, en algunos modelos, el recuento en el registro general 15 puede ser bajo si la CPU detecta la condición de cancelación después de la finalización de la instrucción TBEGINC, pero antes de la finalización de la instrucción AHI.

Según lo observado por la CPU, las búsquedas y los almacenamientos realizados en el modo de ejecución de transacción no son diferentes a los realizados mientras no esté en el modo de ejecución de la transacción.

En una realización, el usuario (es decir, el que crea la transacción) selecciona si se va a restringir o no una transacción. Una realización de la lógica asociada con el procesamiento de transacciones restringidas, y, en concreto, el procesamiento asociado con una instrucción TBEGINC, se describe haciendo referencia a la figura 11. La ejecución de la instrucción TBEGINC hace que la CPU entre en el modo restringido de ejecución de transacción o permanezca en el modo de ejecución no restringido. La CPU (es decir, el procesador) que ejecuta TBEGINC ejecuta la lógica de la figura 11.

Haciendo referencia a la figura 11, en base a la ejecución de una instrucción TBEGINC, se realiza una función de serialización, ETAPA 1100. Una función u operación de serialización incluye completar todos los accesos de almacenamiento previos conceptualmente (y, para Arquitectura/z, como ejemplo, configuraciones del bit de referencia relacionado y del bit de cambio) por la CPU, según lo observado por otras CPU y por el subsistema I/O, antes de que ocurran los accesos de almacenamiento conceptualmente posteriores (y configuraciones del bit de referencia relacionado y del bit de cambio). La serialización afecta a la secuencia de todos los accesos de la CPU al almacenamiento y a las claves de almacenamiento, excepto los asociados con una entrada de la tabla de ART y la búsqueda de una entrada de la tabla de DAT.

Según lo observado por una CPU en el modo de ejecución de transacción, la serialización funciona normalmente (tal como se describió anteriormente). Según lo observado por otras CPU y por el subsistema I/O, una operación de serialización realizada mientras una CPU está en el modo de ejecución de transacción ocurre cuando la CPU sale del modo de ejecución de transacción, ya sea como resultado de una instrucción TRANSACTION END que disminuye la profundidad de anidamiento de la transacción hasta cero (final normal), o como resultado de la cancelación de la transacción.

Después de realizar la serialización, se determina si se reconoce una excepción, CONSULTA 1102. Si es así, se gestiona la excepción, ETAPA 1104. Por ejemplo, se reconoce una excepción de operación especial y la operación se suprime si el control de la ejecución de la transacción, el bit 8 del registro de control 0, es 0. Como ejemplos adicionales, se reconoce una excepción de la especificación y se suprime la operación, si el campo B_1, los bits 16 a 19 de la instrucción, es distinto de cero; se reconoce una excepción de ejecución y se suprime la operación, si TBEGINC es el objetivo de una instrucción del tipo de ejecución; y se reconoce una excepción de operación y la operación se suprime, si la función de ejecución de transacción no está instalada en la configuración. Si la CPU ya se encuentra en el modo restringido de ejecución de transacción, entonces, se reconoce una excepción del programa de excepción de transacción restringida y se suprime la operación. Además, si la profundidad de anidamiento de la transacción, cuando se incrementa en 1, excediese la profundidad máxima de anidamiento de una transacción que depende del modelo, la transacción se cancela con el código de cancelación 13. Pueden reconocerse y manejarse otras excepciones o diferentes excepciones.

No obstante, si no hay una excepción, se determina si la profundidad de anidamiento de la transacción es cero, CONSULTA 1106. Si la profundidad de anidamiento de la transacción es cero, la dirección del bloque de diagnóstico de la transacción se considera no válida, ETAPA 1108; la PSW de cancelación de la transacción se configura a partir del contenido actual de la PSW, excepto por que la dirección de instrucción de la PSW de cancelación de la transacción designa la instrucción TBEGINC, en lugar de la siguiente instrucción secuencial, ETAPA 1110; y los contenidos de las parejas de registros generales designadas por la máscara de guardar registro general se guardan en una ubicación que depende del modelo a la que el programa no puede acceder directamente, ETAPA 1112. Además, la profundidad de anidamiento se configura en 1, ETAPA 1114. Además, el valor efectivo de la operación de permitir coma flotante (F) y los controles de filtrado de interrupción del programa (PIFC) se configuran en cero, ETAPA 1316. Además, se determina el valor efectivo del control de modificación AR (A), campo de bit 12 del campo I_2 de la instrucción, ETAPA 1118. Por ejemplo, el control A efectivo es el AND lógico del control A en la instrucción TBEGINC para el nivel actual y para cualquier instrucción exterior de TBEGIN.

Volviendo a la CONSULTA 1106, si la profundidad de anidamiento de la transacción es mayor de cero, entonces la profundidad de anidamiento se incrementa en 1, ETAPA 1120. Además, el valor efectivo de la operación de permitir coma flotante (F) se configura en cero, y el valor efectivo del control de filtrado de interrupción del programa (PIFC) no cambia, ETAPA 1122. El procesamiento continúa con la etapa 1118. En una realización, una iniciación con éxito de la transacción resulta en el código de condición 0. Esto concluye una realización de la lógica asociada con la ejecución de una instrucción TBEGINC.

En una realización, la verificación de excepciones proporcionada anteriormente puede tener lugar en un orden variable. Un orden concreto para la verificación de excepciones es el siguiente:

Excepciones con la misma prioridad que la prioridad de las condiciones de interrupción del programa para el caso general.

Excepción de especificación debido a que el campo B_1 contiene un valor distinto de cero.

Cancelación debido a exceder la profundidad de anidamiento de la transacción.

Código de condición 0 debido a la finalización normal.

Adicionalmente, en una o más realizaciones se aplica lo siguiente:

1. Los registros designados para ser guardados por la máscara de guardar registro general solo se restauran si la transacción es cancelada, no cuando la transacción finaliza normalmente por medio de TRANSACTION END. Solo los registros designados por el GRSM de la instrucción TRANSACTION BEGIN más exterior son restaurados tras la cancelación.

   El campo I_2 debe designar todas las parejas de registros que proporcionan valores de entrada que son modificados mediante una transacción restringida. Por lo tanto, si se cancela la transacción, los valores del registro de entrada se restaurarán a sus contenidos originales cuando se vuelva a ejecutar la transacción restringida.

2. En la mayoría de los modelos, se puede conseguir un mejor rendimiento, tanto en TRANSACTION BEGIN como cuando se cancela una transacción, especificando el número mínimo de registros necesarios para ser guardados y restaurados en la máscara de guardar registro general.

3. Lo siguiente ilustra los resultados de la instrucción TRANSACTION BEGIN (tanto TBEGIN como TBEGINC) en base a la profundidad actual de anidamiento de la transacción (TND) y, cuando el TND es distinto de cero, si la CPU está en el modo no restringido de ejecución de transacción o restringida:

Instrucción	TND = 0
TBEGIN	Entrar en el modo no restringido de ejecución de transacción
TBEGINC	Entrar en el modo restringido de ejecución de transacción

Instrucción	TND > 0	
	Modo TNX	Modo CTX
TBEGIN	Continuar en el modo no restringido de ejecución de transacción	Excepción de transacción restringida
TBEGINC	Continuar en el modo no restringido de ejecución de transacción	Excepción de transacción restringida

Explicación:

CTX La CPU está en el modo restringido de ejecución de transacción

NTX La CPU está en el modo no restringido de ejecución de transacción

TND Profundidad de anidamiento de la transacción al inicio de la instrucción.

Tal como se describe en este documento, en un aspecto, una transacción restringida tiene garantizada la finalización, suponiendo que no contenga una condición que imposibilite su finalización. Para garantizar que se completa, el procesador (por ejemplo, la CPU) que ejecuta la transacción puede tomar ciertas medidas. Por ejemplo, si una transacción restringida tiene una condición de cancelación, la CPU, temporalmente, puede:

(a) inhibir la ejecución de fuera de servicio;

(b) impedir el acceso de otras CPU a las ubicaciones de almacenamiento conflictivas;

(c) inducir demoras aleatorias en el procesamiento de la cancelación; y/o

(d) invocar otras medidas para facilitar la finalización con éxito.

Para resumir, el procesamiento de una transacción restringida es, como sigue:

- Si ya se está en el modo TX restringido, se reconoce una excepción de transacción restringida.
- Si la TND (Profundidad de anidamiento de transacción) actual > 0, la ejecución continúa como si fuera una transacción no restringida
    - Control efectivo F configurado en cero

- PIFC efectivo no cambia
- Permite que TX no restringida exterior llame a la función de servicio que puede o no utilizar TX restringido.

- Si TND actual = 0:
  - La dirección del bloque de diagnóstico de la transacción no es válida
    - No se ha guardado ningún TDB especificado por la instrucción tras la cancelación
  - PSW de cancelación de la transacción configurada en la dirección de TBEGINC
    - No es la siguiente instrucción secuencial
  - Parejas de registros generales designadas por GRSM guardadas en una ubicación que depende del modelo no accesible por el programa
  - Testigo de la transacción formado opcionalmente (a partir del operando D_2). El testigo de la transacción es un identificador de la transacción. Puede ser igual a la dirección del operando de almacenamiento o puede tener otro valor.
- A efectiva = TBEGINC A y cualquier A exterior
- TND incrementada
- Si TND pasa de 0 a 1, la CPU entra en el modo TX restringido
  - De lo contrario, la CPU permanece en el modo TX no restringido
  - La instrucción se completa con CC0
- Excepciones:
  - Excepción de especificación (PIC (Código de interrupción de programa) 0006) si el campo B_1 es distinto de cero
  - Excepción de operación especial (PIC 0013 hex) si el control de la ejecución de la transacción (CR0.8) es cero
  - Excepción de restricción de transacción (PIC 0018 hex) si se emite en el modo TX restringido
  - Excepción de operación (PIC 0001) si la función de ejecución de transacción restringida no está instalada
  - Excepción de ejecución (PIC 0003) si la instrucción es el objetivo de una instrucción del tipo de ejecución
  - Código de cancelación 13 si se excedió la profundidad de anidamiento
- Condiciones de cancelación en transacción restringida:
  - La PSW de cancelación apunta a la instrucción TBEGINC
    - La instrucción no la sigue
    - La condición de cancelación causa que se debe reenviar la TX completa
      * Ruta sin fallos
  - La CPU toma medidas especiales para garantizar la finalización con éxito en el reenvío
  - Suponiendo que no haya un conflicto, interrupción o violación de restricción persistentes, se garantiza la finalización de la transacción finalmente.
- Violación de restricción:
  - PIC 0018 hex - indica una violación de restricción de la transacción
  - O bien, la transacción es ejecutada como si no estuviera restringida

Tal como se describió anteriormente, además del procesamiento de transacción restringida, que es opcional, en una realización, la facilidad de transacción también proporciona procesamiento de transacción no restringida. Se describen detalles adicionales con respecto al procesamiento de transacciones no restringidas, y, en concreto, el procesamiento asociado con una instrucción TBEGIN haciendo referencia a la figura 12. La ejecución de la instrucción TBEGIN hace que la CPU entre o permanezca en el modo no restringido de ejecución de transacción. La CPU (es decir, el procesador) que ejecuta TBEGIN ejecuta la lógica de la figura 12.

Haciendo referencia a la figura 12, en base a la ejecución de la instrucción TBEGIN, se ejecuta una función de serialización (descrita anteriormente), ETAPA 1200. Después de realizar la serialización, se determina si se reconoce una excepción, CONSULTA 1202. Si es así, entonces la excepción se gestiona, ETAPA 1204. Por ejemplo, se reconoce una excepción de operación especial y la operación se suprime si el control de ejecución de la transacción, bit 8 del registro de control 0, es cero. Además, se reconoce una excepción de especificación y la operación se suprime si el control de filtrado de interrupción del programa, bits 14 a 15 del campo I_2 de la instrucción, contiene el valor 3; o la dirección del primer operando no designa un límite de palabra doble. Se reconoce una excepción de operación y la operación se suprime, si la función de ejecución de la transacción no está instalada en la configuración; y se reconoce una excepción de ejecución y la operación se suprime si la instrucción TBEGIN es el objetivo de una instrucción del tipo de ejecución. Además, si la CPU se encuentra en el modo restringido de ejecución de transacción, entonces se reconoce una excepción de programa de excepción restringida a la transacción y se suprime la operación. Además, si la profundidad de anidamiento de la transacción, cuando se incrementa en 1, excediese una profundidad máxima de anidamiento de la transacción que depende del modelo, la transacción se cancela con el código de cancelación 13.

Además, cuando el campo B_1 de la instrucción es distinto de cero y la CPU no está en el modo de ejecución de transacción, es decir, la profundidad de anidamiento de la transacción es cero, entonces se determina la accesibilidad del almacenamiento al primer operando. Si no se puede acceder al primer operando para los almacenamientos, se reconoce una excepción de acceso y la operación es anulada, suprimida o terminada, dependiendo de la condición específica de excepción del acceso. Además, cualquier evento de alteración del almacenamiento PER para el primer operando es reconocido. Cuando el campo B_1 es distinto de cero y la CPU ya está en el modo de ejecución de transacción, es impredecible si se determina la accesibilidad del almacenamiento al primer operando y se detectan eventos de alteración del almacenamiento PER para el primer operando. Si el campo B_1 es cero, entonces no se accede al primer operando.

Además de la verificación de excepciones, se determina si la CPU está en el modo de ejecución de transacción (es decir, la profundidad de anidamiento de la transacción es cero), CONSULTA 1206. Si la CPU no está en el modo de ejecución de transacción, entonces los contenidos de las parejas de registros generales seleccionadas se guardan, ETAPA 1208. En concreto, los contenidos de las parejas de registros generales designadas por la máscara de guardar registros generales se guardan en una ubicación que depende del modelo a la que el programa no puede acceder directamente.

Además, se determina si el campo B_1 de la instrucción es cero, CONSULTA 1210. Si el campo B_1 no es igual a cero, la dirección del primer operando se coloca en la dirección del bloque de diagnóstico de la transacción, ETAPA 1214, y la dirección del bloque de diagnóstico de la transacción es válida. Además, la PSW de cancelación de transacción se configura a partir del contenido de la PSW actual, ETAPA 1216. La dirección de la instrucción de la PSW de cancelar la transacción designa la siguiente instrucción secuencial (es decir, la instrucción que sigue a la instrucción TBEGIN más exterior).

Además, se realiza una determinación del valor efectivo del control permitido de modificación AR (A), bit 12 del campo I_2 de la instrucción, ETAPA 1218. El control efectivo A es el AND lógico del control A en la instrucción TBEGIN para el nivel actual y para todos los niveles exteriores. Además, se determina un valor efectivo del control de la operación de coma flotante (F), bit 13 del campo I_2 de la instrucción, ETAPA 1220. El control efectivo F es el AND lógico del control F en la instrucción TBEGIN para el nivel actual y para todos los niveles exteriores. Además, se determina un valor efectivo del control de filtrado de interrupción del programa (PIFC), bits 14 a 15 del campo I_2 de la instrucción, ETAPA 1222. El valor efectivo de PIFC es el valor más alto en la instrucción TBEGIN para el nivel actual y para todos los niveles exteriores.

Además, se agrega un valor de uno a la profundidad de anidamiento de transacción, ETAPA 1224, y la instrucción se completa con el código de condición de configuración 0, ETAPA 1226. Si la profundidad de anidamiento de la transacción pasa de cero a uno, la CPU entra en el modo no restringido de ejecución de transacción; de lo contrario, la CPU permanece en el modo no restringido de ejecución de transacción.

Volviendo a la CONSULTA 1210, si B_1 es igual a cero, entonces la dirección del bloque de diagnóstico de la transacción no es válida, ETAPA 1211 y el procesamiento continúa con la ETAPA 1218. De manera similar, si la CPU está en modo de ejecución de transacción, CONSULTA 1206, el procesamiento continúa con la ETAPA 1218.

El código de condición resultante de ejecución de TBEGIN incluye, por ejemplo:

0 Iniciación de la transacción con éxito

1 --

2 --

3 --

Las excepciones del programa incluyen, por ejemplo:

- Acceso (almacenar, primer operando)
- Operación (función de ejecución de transacción no instalada)
- Operación especial
- Especificación
- Restricción de transacción (debido a instrucción restringida)

En una realización, la verificación de excepciones proporcionada anteriormente puede ocurrir en orden variable. Un orden concreto para la verificación de excepciones es el siguiente:

- Excepciones con la misma prioridad que la prioridad de las condiciones de interrupción del programa para el caso general.
- Excepción de especificación debido al valor de PIFC reservado.
- Excepción de especificación debido a que la dirección del primer operando no está en un límite de doble palabra.
- Excepción de acceso (cuando el campo B_1 es distinto de cero).
- Cancelación debido a la superación de la profundidad máxima de anidamiento de la transacción.
- Código de condición 0 debido a una finalización normal.

Notas:

1. Cuando el campo B_1 es distinto de cero, se aplica lo siguiente:

   - Se debe proporcionar un bloque de diagnóstico de transacción (TDB) accesible cuando se inicia una transacción más avanzada, incluso si la transacción nunca se cancela.
   - Dado que es impredecible si se prueba la accesibilidad del TDB para transacciones anidadas, se debe proporcionar un TDB accesible para cualquier instrucción TBEGIN anidada.
   - El rendimiento de cualquier TBEGIN en el que el campo B_1 es distinto de cero, y el rendimiento de cualquier procesamiento de cancelación que se produzca para una transacción iniciada por un TBEGIN exterior en el que el campo B_1 es distinto de cero, puede ser más lento que cuando el campo B_1 es cero.

2. Los registros designados para ser guardados por la máscara de guardar registro general solo se restauran, en una realización, si la transacción es cancelada, no cuando la transacción finaliza normalmente por medio de la instrucción TRANSACTION END. Solo los registros designados por el GRSM de la instrucción TRANSACTION BEGIN más exterior son restaurados tras la cancelación.

   - El campo I_2 debe designar todas las parejas de registros que proporcionan valores de entrada que son cambiados por la transacción. Por lo tanto, si se cancela la transacción, los valores del registro de entrada se restaurarán a su contenido original cuando se ingrese el gestor de cancelación.

3. Se espera que la instrucción TRANSACTION BEGIN (TBEGIN) vaya seguida de una instrucción de bifurcación condicional que determinará si la transacción se inició con éxito.

4. Si una transacción se cancela debido a condiciones que no resultan en una interrupción, la instrucción designada por la PSW de cancelación de la transacción recibe el control (es decir, la instrucción que sigue a la instrucción TRANSACTION BEGIN (TBEGIN) más exterior). Además del código de condición configurado por la instrucción TRANSACTION BEGIN (TBEGIN), los códigos de condición 1 a 3 también se configuran cuando se cancela una transacción.

   Por lo tanto, la secuencia de instrucciones que sigue a la instrucción TRANSACTION BEGIN (TBEGIN) más exterior debería ser capaz de contener los cuatro códigos de condición, incluso aunque la instrucción TBEGIN solo configura el código 0, en este ejemplo.

5. En la mayoría de los modelos, se puede conseguir un mejor rendimiento, tanto en TRANSACTION BEGIN como cuando se cancela una transacción, especificando el número mínimo de registros necesarios para ser guardados y restaurados en la máscara de guardar registro general.

6. Mientras está en el modo no restringido de ejecución de transacción, un programa puede llamar a una función de servicio que puede alterar los registros de acceso o los registros de coma flotante (incluido el registro de control de coma flotante). Aunque dicha rutina de servicio puede guardar los registros alterados en la entrada y restaurarlos en la salida, la transacción puede ser cancelada antes de la salida normal de la rutina. Si el programa de llamada no prevé la conservación de estos registros mientras la CPU se encuentra en el modo no restringido de ejecución de transacción, es posible que no pueda tolerar la alteración de los registros de la función de servicio.

   Para evitar la alteración inadvertida de los registros de acceso en el modo no restringido de ejecución de transacción, el programa puede configurar el control de permitir modificación AR, bit 12 del campo I_2 de la instrucción TRANSACTION BEGIN, en cero. De manera similar, para evitar la alteración inadvertida de los registros de coma flotante, el programa puede configurar el control de permitir operación de coma flotante, bit 13 del campo I_2 de la instrucción TBEGIN, en cero.

7. Las condiciones de excepción del programa reconocidas durante la ejecución de la instrucción TRANSACTION BEGIN (TBEGIN) están sujetas al control efectivo de filtrado de interrupción del programa configurado mediante cualquier instrucción exterior de TBEGIN. Las condiciones de excepción del programa reconocidas durante la ejecución de la instrucción TBEGIN más exterior no están sujetas a filtrado.

8. Para actualizar múltiples ubicaciones de almacenamiento de manera serializada, las secuencias de códigos convencionales pueden emplear una palabra de bloqueo (semáforo). Si (a) se utiliza la ejecución de transacción para implementar actualizaciones de múltiples ubicaciones de almacenamiento, (b) el programa también proporciona una ruta de "búsqueda" que se invocará si la transacción se cancela, y (c) la ruta de retroceso utiliza una palabra de bloqueo, entonces la ruta de ejecución de transacción también debe probar la disponibilidad del bloqueo y, si el bloqueo no está disponible, finalizar la transacción mediante la instrucción TRANSACTION END y bifurcarse a la ruta de retroceso. Esto garantiza un acceso coherente a los recursos serializados, independientemente de si se actualizan en modo de transacción.

   Alternativamente, el programa podría ser cancelado si el bloqueo no está disponible, no obstante, el procesamiento de cancelación puede ser significativamente más lento que simplemente finalizar la transacción mediante TEND.

9. Si el control efectivo del filtrado de interrupción del programa (PIFC) es mayor que cero, la CPU filtra la mayoría de las interrupciones del programa de excepción de datos. Si el control efectivo de la operación en coma flotante (F) es cero, el código de excepción de datos (DXC) no se configurará en el registro de control de coma flotante como resultado de una cancelación debido a una condición de excepción del programa de excepción de datos. En este escenario (se aplica el filtrado y el control F efectivo es cero), la única ubicación en la que se inspecciona el DXC está en el TDB especificado por TBEGIN. Si el controlador de cancelación del programa va a inspeccionar el DXC en dicha condición, el registro general B_1 debe ser distinto de cero, de modo que se establezca una dirección válida del bloque de diagnóstico de la transacción (TDBA).

10. Si existe una alteración de almacenamiento PER o una condición de detección de dirección cero para el TDB especificado por el TDB especificado por TBEGIN de la instrucción TBEGIN más exterior, y no se aplica la supresión de eventos PER, el evento PER se reconoce durante la ejecución de la instrucción, lo que provoca la interrupción inmediata de la transacción, independientemente de si existe alguna otra condición de cancelación.

En una realización, la instrucción TBEGIN configura implícitamente que la dirección de cancelación de transacción sea la próxima instrucción secuencial que sigue a TBEGIN. Esta dirección está destinada a ser una instrucción de bifurcación condicional que determina si se bifurca dependiendo del código de condición (CC). Una TBEGIN con éxito configura CC0, mientras que una transacción abortada configura CC1, CC2 o CC3.

En una realización, la instrucción TBEGIN proporciona un operando de almacenamiento opcional que designa la dirección de un bloque de diagnóstico de transacciones (TDB) en el que se almacena la información si se cancela la transacción.

Además, proporciona un operando inmediato que incluye lo siguiente:

una máscara de guardar registro general (GRSM), que indica qué parejas de registros generales deben guardarse al inicio de la ejecución de transacción y ser restauradas si se cancela la transacción;

un bit (A), permitir la cancelación de la transacción si la transacción modifica los registros de acceso;

un bit (F), para permitir la cancelación de la transacción si la transacción intenta ejecutar instrucciones de coma flotante; y

un control de filtrado de interrupción de programa (PIFC), que permite niveles de transacción individuales para eludir la presentación real de una interrupción de programa si se cancela una transacción.

Los controles A, F y PIFC pueden ser diferentes en varios niveles de anidamiento y restaurarse al nivel anterior cuando finalizan los niveles de transacción interior

Además, la instrucción TBEGIN (o en otra realización, TBEGINC) se utiliza para formar un testigo de recepción. Opcionalmente, el testigo puede coincidir con un testigo formado por la instrucción TEND. Para cada instrucción TBEGIN (o TBEGINC), como ejemplo, se forma un testigo a partir de la dirección del primer operando. Esta señal se puede formar independientemente de si el registro base es cero (a diferencia de la configuración de la dirección TDB que solo se produce cuando el registro base es distinto de cero). Para cada instrucción de TRANSACTION END ejecutada con un registro base distinto de cero, se forma un testigo similar a partir de su operando de almacenamiento. Si los testigos no coinciden, se puede reconocer una excepción de programa para alertar al programa de una instrucción desparejada.

La coincidencia de testigos proporciona un mecanismo destinado a mejorar la fiabilidad del software al garantizar que una declaración TEND se empareja correctamente con una TBEGIN (o TBEGINC). Cuando se ejecuta una instrucción TBEGIN en un nivel de anidamiento particular, se forma un testigo a partir de la dirección del operando de almacenamiento que identifica esta instancia de una transacción. Cuando se ejecuta una instrucción TEND correspondiente, se forma un testigo a partir de la dirección del operando de almacenamiento de la instrucción, y la CPU compara el testigo de inicio para el nivel de anidamiento con el testigo final. Si los testigos no coinciden, se reconoce una condición de excepción. Un modelo puede implementar la coincidencia de testigos solo para un cierto número de niveles de anidamiento (o para ningún nivel de anidamiento). El testigo puede no involucrar todos los bits de la dirección del operando de almacenamiento, o los bits pueden combinarse mediante verificación aleatoria u otros métodos. Se puede formar un testigo mediante la instrucción TBEGIN incluso si no se accede a su operando de almacenamiento.

En resumen, el procesamiento de una transacción no restringida es el siguiente:

- Si TND = 0:
    - Si $B_1 \neq 0$, la dirección del bloque de diagnóstico de la transacción se configura a partir de la dirección del primer operando.
    - La PSW de cancelación de transacción se configura en la siguiente dirección de instrucción secuencial.
    - Las parejas de registros generales designadas por el campo I_2 se guardan en la ubicación que depende del modelo.
        - No accesible directamente por el programa
- Controles PIFC, A y F efectivos calculados
    - A Efectivo = TBEGIN A y cualquier A exterior
    - F Efectivo = TBEGIN F y cualquier F exterior
    - PIFC Efectivo = max (TBEGIN PIFC, cualquier PIFC exterior)
- Profundidad de anidamiento de la transacción (TND) incrementada
    - Si TND pasa de 0 a 1, la CPU entra en el modo de ejecución de transacción
    - Código de condición configurado en cero
        - Cuando la instrucción siguiente a TBEGIN recibe control:
            - éxito de TBEGIN indicado por CC0
            - transacción cancelada indicada por CC distinto de cero
- Excepciones:
    - Código de cancelación 13 si se excede la profundidad de anidamiento
    - Excepción de acceso (uno de varios PIC) si el campo B_1 es cero, y no se puede acceder al operando de almacenamiento para una operación de almacenamiento

- Excepción de ejecución (PIC 0003) si la instrucción TBEGIN es el objetivo de una instrucción del tipo de ejecución
- Excepción de operación (PIC 0001) si la función de ejecución de transacción no está instalada
- PIC 0006 si cualquiera de
  - PIFC no es válido (valor de 3)
  - Dirección del segundo operando no alineada con doble palabra
- PIC 0013 hex si el control de ejecución de la transacción (CR0.8) es cero
- PIC 0018 hex si se emite en modo TX restringido

Tal como se indicó anteriormente, una transacción, ya sea restringida o no, puede finalizar mediante una instrucción de TRANSACTION END (TEND). Se describen detalles adicionales con respecto al procesamiento de la instrucción de fin de la transacción (TEND) haciendo referencia a la figura 13. La CPU (es decir, el procesador) que ejecuta la TEND ejecuta la lógica de la figura 13.

Haciendo referencia a la figura 13, inicialmente, en función de la obtención del procesador (por ejemplo, búsqueda, recepción, etc.) de la instrucción TEND, se realizan varias verificaciones de excepción y si hay una excepción, CONSULTA 1300, se trata de la excepción, ETAPA 1302. Por ejemplo, si la instrucción END OF TRANSACTION es el objetivo de una instrucción del tipo de ejecución, la operación se suprime y se reconoce una excepción de ejecución; y se reconoce una excepción de operación especial y la operación se suprime si el control de ejecución de la transacción, bit 8 de CR0, es cero. Además, se reconoce una excepción de operación y se suprime la operación, si la función de ejecución de transacción no está instalada en la configuración.

Volviendo a la CONSULTA 1300, si no se reconoce una excepción, entonces la profundidad de transacción se disminuye (por ejemplo, en uno), ETAPA 1304. Se determina si la profundidad de anidamiento de la transacción es cero después de la disminución, CONSULTA 1306. Si la profundidad de anidamiento de la transacción es cero, entonces todos los accesos al almacenamiento realizados por la transacción (y otras transacciones dentro del nido de transacciones, si las hay, de las que esta transacción forma parte) se comprometen, ETAPA 1308. Además, la CPU sale del modo de ejecución de transacción, ETAPA 1310, y la instrucción se completa, ETAPA 1312.

Volviendo a la CONSULTA 1306, si la profundidad de anidamiento de la transacción no es igual a cero, entonces la instrucción TRANSACTION END se completa.

Si la CPU está en el modo de ejecución de transacción al inicio de la operación, el código de condición se configura en 0; de lo contrario, el código de condición se configura en 2.

Se observa que el control efectivo de la operación en coma flotante (F), el control de permitir modificación de AR (A) y el control del filtrado de interrupción del programa (PIFC) son reiniciados en sus valores respectivos antes de la instrucción TRANSACTION BEGIN que inició el nivel haya terminado. Además, se realiza una función de serialización al finalizar la operación.

Los eventos de búsqueda de la instrucción PER y fin de transacción que se reconocen al completar la instrucción TRANSACTION END más exterior no resultan en la cancelación de la transacción.

En un ejemplo, la instrucción TEND incluye asimismo un campo de base B_2 y un campo de desplazamiento D_2, que se combinan (por ejemplo, se suman) para crear una dirección del segundo operando. En este ejemplo, se puede llevar a cabo la coincidencia de testigos. Por ejemplo, cuando B_2 es distinto de cero, los bits seleccionados de la dirección del segundo operando son comparados con un testigo de transacción formado por el TBEGIN correspondiente. Si hay una discrepancia, existe una excepción (por ejemplo, PIC 0006).

Además de lo anterior, una transacción puede cancelarse implícita o explícitamente mediante una instrucción TRANSACTION ABORT. Cancelar una transacción mediante TABORT u otra incluye llevar a cabo una serie de etapas. Un ejemplo de las etapas para el procesamiento de cancelación, en general, se describe haciendo referencia a la figura 14. Si hay una diferencia en el procesamiento en función de si la cancelación ha sido iniciada mediante TABORT u otra, se indica en la descripción que sigue. En un ejemplo, un procesador (por ejemplo, la CPU) está ejecutando la lógica de la figura 14.

Haciendo referencia a la figura 14, inicialmente, en base a la ejecución de la instrucción TABORT o a una cancelación implícita, los accesos al almacenamiento no de transacción realizados mientras la CPU estaba en el modo ejecución de transacción son comprometidos, ETAPA 1400. Otros almacenamientos (por ejemplo, almacenamientos de transacción) creados mientras la CPU estaba en el modo de ejecución de transacción son descartados, ETAPA 1402.

La CPU sale del modo de ejecución de transacción, ETAPA 1404, y los subsiguientes almacenamientos se producen de manera no de transacción. La PSW actual es reemplazada con el contenido de la PSW de cancelación de transacción, excepto por que el código de condición se configura tal como se describió anteriormente (aparte de la condición que sigue, en la que, si la TDBA es válida, pero el bloque es inaccesible, entonces CC = 1), ETAPA 1406. Como parte de un procesamiento posterior a la cancelación, el procesamiento se bifurca a la ubicación especificada por la PSW de cancelación de la transacción para realizar una acción. En un ejemplo en el que la transacción es una transacción restringida, la ubicación es la instrucción TBEGINC y la acción es la ejecución de nuevo de esa instrucción; y, en otro ejemplo, en el que la transacción es una transacción no restringida, la ubicación es la instrucción después de TBEGIN, y la acción es la ejecución de esa instrucción, que puede ser, por ejemplo, una bifurcación a un gestor de cancelación.

A continuación, se determina si la dirección del bloque de diagnóstico de la transacción es válida, CONSULTA 1408. Cuando la dirección del bloque de diagnóstico de la transacción es válida, la información de diagnóstico que identifica el motivo de la cancelación y los contenidos de los registros generales se almacenan en el bloque de diagnóstico de la transacción especificada por TBEGIN, ETAPA 1410. Los campos de TDB almacenados y las condiciones en las que están almacenados se describen anteriormente haciendo referencia al bloque de diagnóstico de la transacción.

Si la dirección del bloque de diagnóstico de la transacción es válida, pero el bloque se ha vuelto inaccesible, después de la ejecución de la instrucción TBEGIN más exterior, no se accede al bloque y se aplica el código de condición 1.

Para transacciones que se cancelan debido a condiciones de excepción de programa que resultan en una interrupción, se almacena el TDB de interrupción del programa.

Volviendo a la consulta 1408, si la dirección del bloque de diagnóstico de la transacción no es válida, no se almacena ningún TDB especificado por TBEGIN y se aplica el código de condición 2 o 3, dependiendo del motivo de la cancelación.

Además de lo anterior, la profundidad de anidamiento de la transacción se configura igual a cero, ETAPA 1412. Además, se restauran todas las parejas de registros generales designadas para ser guardadas por la instrucción TBEGIN más exterior, ETAPA 1414. Parejas de registros generales que no fueron designadas para ser guardadas por la instrucción TBEGIN más exterior no se restauran cuando se cancela una transacción.

Además, se lleva a cabo una función de serialización, ETAPA 1416. Una función u operación de serialización incluye completar todos los accesos de almacenamiento conceptualmente previos (y, para la Arquitectura/z, como ejemplo, configuraciones del bit de referencia relacionado y del bit de cambio) por parte de la CPU, según lo observado por otra CPU y por el subsistema I/O, antes de que ocurran los accesos de almacenamiento conceptualmente posteriores (y configuraciones del bit de referencia relacionado y del bit de cambio). La serialización efectúa la secuencia de todos los accesos de la CPU al almacenamiento y a las claves de almacenamiento, excepto los asociados con la entrada de la tabla ART y la búsqueda de entrada de la tabla DAT.

Según lo observado por una CPU en el modo de ejecución de transacción, la serialización funciona normalmente (tal como se describió anteriormente). Según lo observado por otras CPU y por el subsistema I/O, una operación de serialización realizada mientras una CPU está en el modo de ejecución de transacción ocurre cuando la CPU sale del modo de ejecución de transacción, ya sea como resultado de una instrucción TRANSACTION END que disminuye la profundidad de anidamiento de la transacción hasta cero (final normal) o como resultado de la interrupción de la transacción.

Para el procesamiento de cancelación iniciado por una instrucción distinta de TABORT, si la transacción se interrumpe debido a una condición de excepción que resulta en una interrupción, CONSULTA 1418, los códigos de interrupción o los parámetros asociados con la interrupción se almacenan en las ubicaciones de almacenamiento asignadas correspondientes al tipo de interrupción, ETAPA 1420. Además, la PSW actual, tal como se presentó anteriormente, se almacena en la PSW anterior de interrupción, ETAPA 1422. Posteriormente, o si la transacción no se canceló debido a una condición de excepción que resultó en una interrupción, la instrucción finaliza con el código de condición cero.

Además de lo anterior, en una realización para la ejecución interpretativa de la Arquitectura/z, cuando la CPU está en el modo de ejecución de transacción, y ocurre una condición de invitado que normalmente daría como resultado los códigos de interceptación 4, 12, 44, 56, 64, 68 o 72, la interceptación no ocurre. En su lugar, la CPU permanece en el modo de ejecución interpretativa, y las condiciones de cancelación se indican al invitado de la siguiente manera:

- Para una transacción no restringida, la transacción se cancela debido a una instrucción restringida (código de cancelación 11). Si se detectó un evento PER simultáneo y la CPU está habilitada para PER, se produce una interrupción del programa con el código de interrupción 0280 hex.

- Para una transacción restringida, se reconoce una excepción de restricción de transacción. Si se detectó un evento PER continuo y la CPU está habilitada para PER, se produce una interrupción del programa con el código de interrupción 0298 hex.

Cuando se cancela una transacción debido a una condición de excepción del programa, el filtrado de la interrupción del programa puede inhibir la presentación de una interrupción. Para las interrupciones del programa que pueden resultar en la interceptación, el filtrado también inhibe la interceptación.

Tal como se indicó anteriormente, de acuerdo con un aspecto, cuando un programa inicia una transacción no restringida (es decir, una transacción para la cual se proporciona una rutina de recuperación de gestor de cancelación), el programa puede designar una ubicación de almacenamiento en la cual se registrará información de diagnóstico si la transacción es cancelada. Asimismo, se puede proporcionar información de diagnóstico si se cancela una transacción debido a una interrupción del programa, o si la cancelación resulta en una interceptación de ejecución interpretativa, tal como una interceptación de interrupción del programa.

En una realización, en base a una cancelación de una transacción, se proporciona información de diagnóstico en un bloque de diagnóstico de transacción (TDB). La información incluye, por ejemplo: profundidad de anidamiento de la transacción; código de cancelación de la transacción; testigo de conflicto (para transacciones que se cancelan debido a conflictos); dirección de la instrucción de cancelar la transacción; los parámetros de interrupción del programa para las transacciones que se cancelan debido a las condiciones de excepción del programa filtrado; información de diagnóstico que depende del modelo; registros de propósito general en el momento de la cancelación; y/u otra información, tal como se describió anteriormente haciendo referencia a la figura 9.

Cero, uno o dos TDB pueden almacenarse en una cancelación, incluyendo: un TDB especificado por el programa (a.k.a., TBEGIN-TDB) que puede o no estar presente para una transacción no restringida; un TDB de interrupción de programa proporcionado al programa de control cuando se cancela una transacción debido a una condición de excepción del programa no filtrada (es decir, una condición que realmente resulta en una interrupción del programa); y/o un TDB de interceptación proporcionado a un programa anfitrión (por ejemplo, sistema operativo) cuando se cancela una transacción debido a ciertas condiciones de interceptación.

Una realización del procesamiento asociado con el almacenamiento de cero o más TDB está descrita haciendo referencia a la figura 15. Esta lógica la ejecuta un procesador, tal como el procesador que detecta una condición de cancelación.

Haciendo referencia a la figura 15, inicialmente se detecta una cancelación (es decir, un final anormal) de una transacción, ETAPA 1500. A continuación, se determina si se ha especificado una dirección válida del bloque de diagnóstico de transacción (TDBA), CONSULTA 1502. Por ejemplo, es una dirección válida del bloque de diagnóstico de transacción especificada por una instrucción TBEGIN (por ejemplo, B_1 de TBEGIN > 0). Si se proporciona una TDBA válida, la información se almacena en el TDB especificado por el programa, tal como se ha descrito anteriormente, ETAPA 1504.

A continuación, o si no se proporciona una TDBA válida, se realiza una determinación adicional de si la condición de cancelación se debe a una interrupción, CONSULTA 1506. Si es así, entonces la información se almacena en el TDB de interrupción del programa, tal como se indicó anteriormente, ETAPA 1508.

A continuación, o si la cancelación no se debe a una interrupción del programa, se determina si la cancelación dio lugar a una interceptación, CONSULTA 1510. Si la cancelación no dio lugar a una interceptación, este procesamiento se ha completado. De lo contrario, la información se almacena en un TDB de interceptación, tal como se ha descrito anteriormente, ETAPA 1512. Esto concluye el procesamiento.

Tal como se indicó, uno o más TDB o incluso ningún TDB pueden ser almacenados como resultado de una cancelación. No obstante, cada TDB que se almacena proporciona información de diagnóstico relacionada con la cancelación, que puede facilitar la depuración.

Además de proporcionar información de diagnóstico integral, tal como se describió anteriormente, se proporciona un medio eficiente de actualizar múltiples objetos no contiguos en la memoria sin serialización clásica (genérico), tal como el bloqueo, que proporciona un potencial de mejora significativa en el rendimiento de varios procesadores. Es decir, los objetos múltiples, no contiguos son actualizados sin la aplicación de un orden de acceso de almacenamiento más detallado que es proporcionado por técnicas clásicas, tales como bloqueos y semáforos. La ejecución especulativa se proporciona sin una configuración costosa de búsqueda, y se ofrecen transacciones restringidas para actualizaciones simples y compactas.

La ejecución de transacción se puede utilizar en una variedad de escenarios, que incluyen, entre otros, inclusión por referencia (inlining, en inglés) parcial, procesamiento especulativo y elisión de bloqueo. En inclusión por referencia parcial, la región parcial que se incluirá en la ruta ejecutada está envuelta en TBEGIN/TEND. TABORT puede ser incluida en la misma para revertir el estado en una salida lateral. Para la especulación, como en Java, las verificaciones de nulo en punteros sin referencia se pueden retrasar al borde del bucle mediante la utilización de una

transacción. Si el puntero es nulo, la transacción puede ser cancelada de manera segura utilizando TABORT, que se incluye dentro de TBEGIN/TEND.

En cuanto a la elisión de bloqueo, se describe un ejemplo de su utilización haciendo referencia a las figuras 16A a 16B y al fragmento de código proporcionado a continuación.

La figura 16A representa una lista doblemente unida 1600 de una pluralidad de elementos de cola 1602a a 1602d. Se debe introducir un nuevo elemento de cola 1602e en la lista doblemente unida de elementos de cola 1600. Cada elemento de cola 1602a a 1602e incluye un puntero de avance 1604a a 1604e y un puntero de retroceso 1606a a 1606e. Tal como se muestra en la figura 16B, para agregar el elemento de cola 1602e entre los elementos de cola 1602b y 1602c, (1) el puntero de retroceso 1606e se configura para apuntar al elemento de cola 1602b, (2) el puntero de avance 1604e se configura para señalar al elemento de cola 1602c, (3) el puntero de retroceso 1606c se configura para que apunte al elemento de cola 1602e, y (4) el puntero de avance 1604b se configura para que apunte al elemento de cola 1602e.

Un fragmento de código de ejemplo correspondiente a las figuras 16A a 16B se muestra a continuación:

* R1 - dirección del nuevo elemento de cola a introducir.

* R2 - dirección del punto de inserción; el nuevo elemento se inserta antes del elemento al que R2 apunta.

```
NUEVO     UTILIZANDO   QEL, R1
CURR      UTILIZANDO   QEL, R2
          LHI          R15, 10          Cargar recuento de reintentos.
LOOP      TBEGIN       TDB, X'C000 '    Iniciar transacción (guardar GR 0 a 3)
          JNZ          CANCELADA         CC distinto de cero significa cancelada.
          LG           R3, CURR.BWD     Apuntar al elemento anterior.
PREV      UTILIZANDO   QEL, R3          Hacerlo direccionable.
          STG          R1, PREV.FWD     Actualizar ptr. de avance previo
          STG          R1, CURR.BWD     Actualizar ptr. de retroceso actual
          STG          R2, NEW.FWD      Actualizar ptr. de avance nuevo
          STG          R3, NEW.BWD      Actualizar ptr. de retroceso nuevo
          TEND                          Fin de la transacción.
          …
ABORTED   JO           NO_REINTENTO     CC3: Cancelación no re-intentable.
          JCT          R15, LOOP        Reintentar la transacción varias veces.
          J            NO_REINTENTO     Sin éxito tras 10x; hacerlo de la manera difícil.
```

En un ejemplo, si la transacción se utiliza para la elisión del bloqueo, pero la ruta alternativa utiliza un bloqueo, la transacción es, por lo menos, para buscar la palabra de bloqueo para ver si está disponible. El procesador asegura que la transacción se cancela si otra CPU accede al bloqueo en el modo no de transacción.

Tal como se utiliza en el presente documento, almacenamiento, almacenamiento central, almacenamiento principal, memoria y memoria principal se utilizan indistintamente, a menos que se indique lo contrario, implícitamente mediante la utilización, o explícitamente. Además, aunque en una realización, el retardo efectivo incluye retrasar el compromiso de los almacenamientos de transacción a la memoria principal hasta la finalización de una transacción seleccionada; en otra realización, una transacción que efectivamente sufre un retardo incluye permitir actualizaciones de transacción a la memoria, pero manteniendo cancelados los valores anteriores, y restaurar la memoria a los valores anteriores.

Tal como resultará evidente para un experto en la técnica, uno o más aspectos pueden incorporarse como un sistema, método o producto de programa informático. Por consiguiente, uno o más aspectos pueden tomar la forma de una realización completamente de hardware, una realización completamente de software (incluyendo firmware, software residente, microcódigo, etc.) o una realización que combina aspectos de software y hardware que, en general, se pueden denominar en el presente documento un "circuito", "módulo" o "sistema". Además, uno o más

aspectos pueden tomar la forma de un producto de programa informático incorporado en uno o más medios legibles por ordenador que tienen un código de programa legible por ordenador incorporado en el mismo.

Se puede utilizar cualquier combinación de uno o más 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 no está limitado a, un sistema, aparato o dispositivo electrónico, magnético, óptico, electromagnético, infrarrojo o semiconductor, o cualquier combinación adecuada de los anteriores. Ejemplos más específicos (una lista no 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 – Random Access Memory, en inglés), una memoria de solo lectura (ROM – Read Only Memory, en inglés), una memoria de solo lectura programable borrable (memoria EPROM – Erasable Programmable Read-Only Memory o memoria rápida (flash), en inglés), una fibra óptica, una memoria de solo lectura de disco compacto (CD-ROM – Compact Disc-ROM, en inglés) portátil, un dispositivo de almacenamiento óptico, un dispositivo de almacenamiento magnético o cualquier dispositivo adecuado que sea combinación de los anteriores. En el contexto de este documento, un medio de almacenamiento legible por ordenador puede ser cualquier medio tangible que pueda contener o almacenar un programa para su utilización por o en conexión con un sistema, aparato o dispositivo de ejecución de instrucciones.

Haciendo referencia ahora a la figura 17, en un ejemplo, un producto de programa informático 1700 incluye, por ejemplo, uno o más medios de almacenamiento legibles por ordenador no transitorios 1702 para almacenar los medios de código de programa legible por ordenador o lógica 1704 en el mismo, para proporcionar y facilitar una o más realizaciones.

El código de programa incorporado en un medio legible por ordenador puede ser transmitido utilizando un medio apropiado, que incluye, pero sin estar limitado a, conexión inalámbrica, conexión por cable, fibra óptica, RF, etc., o cualquier combinación adecuada de los anteriores.

El código de programa informático para llevar a cabo operaciones para una o más realizaciones se puede escribir en cualquier combinación de uno o más lenguajes de programación, incluyendo un lenguaje de programación orientado a objetos, tal como Java, Smalltalk, C++ o similares, y lenguajes de programación de procedimientos convencionales, tales como el lenguaje de programación "C", ensamblador o lenguajes de programación similares. El código del programa puede ser ejecutado completamente 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 ordenador remoto o completamente en el ordenador o servidor remoto. En este último caso, el ordenador remoto puede estar conectado al ordenador del usuario a través de cualquier tipo de red, incluida una red de área local (LAN – Local Area Network, en inglés) o una red de área extensa (WAN – Wide Area Network, en inglés), o la conexión se puede realizar a un ordenador externo (por ejemplo, a través de Internet utilizando un proveedor de servicios de Internet).

Una o más realizaciones se describen en el presente documento haciendo referencia a ilustraciones de diagrama de flujo y/o a diagramas de bloques de métodos, aparatos (sistemas) y productos de programas informáticos. Se comprenderá que cada bloque de las ilustraciones del diagrama de flujo y/o los diagramas de bloques, y las combinaciones de bloques en las ilustraciones del diagrama de flujo y/o los diagramas de bloques, puede ser implementado mediante instrucciones del programa informático. Estas instrucciones del programa informático pueden ser proporcionadas a un procesador de un ordenador de propósito general, un ordenador de propósito especial u otro aparato de procesamiento de datos programable para producir una máquina, de tal modo que las instrucciones se ejecuten a través del procesador del ordenador o de otro aparato de procesamiento de datos programable, creen un medio para implementar las funciones / actos especificados en el diagrama de flujo, y/o en el bloque o bloques del diagrama de bloques.

Estas instrucciones del programa informático se pueden almacenar asimismo en un medio legible por ordenador que puede dirigir un ordenador, otro aparato de procesamiento de datos programable u otros dispositivos para funcionar de manera particular, de modo que las instrucciones almacenadas en el medio legible por ordenador produzcan un artículo de fabricación incluidas las instrucciones que implementan la función / acto especificada en el diagrama de flujo y/o en el bloque o bloques del diagrama de bloques.

Las instrucciones del programa informático se pueden cargar asimismo en un ordenador, otro aparato de procesamiento de datos programable u otros dispositivos para provocar que se realicen una serie de etapas operacionales en el ordenador, otros aparatos programables u otros dispositivos para producir un proceso implementado por ordenador de manera que las instrucciones que se ejecutan en el ordenador u otro aparato programable proporcionan procesos para implementar las funciones / actos especificados en el diagrama de flujo y/o en el bloque o bloques del diagrama de bloques.

El diagrama de flujo y los diagramas de bloques en las figuras ilustran la arquitectura, funcionalidad y operación de posibles implementaciones de sistemas, métodos y productos de programas informáticos de acuerdo con diversas realizaciones. A este respecto, cada bloque en el diagrama de flujo o diagramas de bloques puede representar un módulo, segmento o porción de código, que comprende una o más instrucciones ejecutables para implementar la

función lógica especificada o las funciones lógicas especificadas. Se debe observar asimismo que, en algunas implementaciones alternativas, las funciones indicadas en el bloque pueden producirse fuera del orden indicado en las figuras. Por ejemplo, dos bloques que se muestran en sucesión pueden, de hecho, ser ejecutados de manera sustancialmente simultánea, o bien, los bloques a veces pueden ser ejecutados en el orden inverso, dependiendo de la funcionalidad involucrada. Se observará asimismo que cada bloque de los diagramas de bloques y/o ilustración del diagrama de flujo, y las combinaciones de bloques en los diagramas de bloques y/o diagrama de flujo, pueden ser implementadas mediante sistemas basados en hardware de propósito especial que realizan las funciones o actos especificados o combinaciones de hardware de propósito especial e instrucciones informáticas.

Además de lo anterior, uno o más aspectos pueden ser proporcionados, ofrecidos, implementados, administrados, atendidos, etc. por un proveedor de servicios que ofrece administración de entornos de clientes. Por ejemplo, el proveedor de servicios puede crear, mantener, respaldar, etc. un código informático y/o una infraestructura informática que realiza uno o más aspectos para uno o más clientes. A cambio, el proveedor de servicios puede recibir el pago del cliente en virtud de un acuerdo de suscripción y/o tarifa, como ejemplos. Adicional o alternativamente, el proveedor de servicios puede recibir el pago de la venta del contenido publicitario a uno o más terceros.

En un aspecto, se puede implementar una aplicación para realizar una o más realizaciones. Como ejemplo, la implementación de una aplicación comprende proporcionar una infraestructura informática operable para realizar una o más formas de realización.

Como aspecto adicional, se puede desplegar una infraestructura informática que comprende la integración de un código legible por ordenador en un sistema informático, en la que el código, en combinación con el sistema informático, es capaz de realizar una o más realizaciones.

Como otro aspecto adicional, se puede proporcionar un proceso para integrar una infraestructura informática que comprende la integración de un código legible por ordenador en un sistema informático. El sistema informático comprende un medio legible por ordenador, en el que el medio informático comprende una o más realizaciones. El código en combinación con el sistema informático es capaz de realizar una o más realizaciones.

Aunque anteriormente se han descrito diversas realizaciones, estos son solo ejemplos. Por ejemplo, se pueden utilizar entornos informáticos de otras arquitecturas. Además, se pueden utilizar diferentes instrucciones, formatos de instrucción, campos de instrucción y/o valores de instrucción. Además, se puede proporcionar / utilizar información de diagnóstico diferente, distinta y/o adicional y/o tipos de bloques de diagnóstico de transacción. Son posibles muchas variaciones.

Además, se pueden utilizar otros tipos de entornos informáticos. Como ejemplo, se puede utilizar un sistema de procesamiento de datos adecuado para almacenar y/o ejecutar código de programa que incluye, por lo menos dos procesadores acoplados directa o indirectamente a elementos de memoria a través de un bus del sistema. Los elementos de memoria incluyen, por ejemplo, la memoria local empleada durante la ejecución real del código de programa, almacenamiento masivo y memoria caché que proporcionan almacenamiento temporal, por lo menos, de algún código de programa para reducir el número de veces que se debe recuperar el código del almacenamiento masivo durante la ejecución.

Dispositivos de entrada / salida o I/O (incluidos, entre otros, teclados, pantallas, dispositivos señaladores, DASD, cinta, CD, DVD, memorias USB y otros medios de memoria, etc.) pueden ser acoplados al sistema directamente o mediante los controladores de I/O intermedios. Adaptadores de red también pueden ser acoplados al sistema 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. Los módems, módems de cable y tarjetas Ethernet son solo algunos de los tipos de adaptadores de red disponibles.

Haciendo referencia a la figura 18, se representan componentes representativos de un sistema informático anfitrión 5000 para implementar una o más realizaciones. El ordenador anfitrión 5000 representativo comprende una o más CPU 5001 en comunicación con la memoria del ordenador (es decir, almacenamiento central) 5002, así como interfaces de I/O para dispositivos de almacenamiento de medios 5011 y redes 5010 para comunicarse con otros ordenadores o SAN y similares. La CPU 5001 es compatible con una arquitectura que tiene un conjunto de instrucciones estructuradas y funcionalidad estructurada. La CPU 5001 puede tener una traducción del registro de acceso (ART) 5012, que incluye una memoria temporal de búsqueda de ART (ALB) 5013, para seleccionar un espacio de direcciones para ser utilizado por la traducción dinámica de direcciones (DAT) 5003 para transformar direcciones de programa (direcciones virtuales) en direcciones reales de memoria. Una DAT típicamente incluye una memoria temporal de traducción (TLB) 5007 para almacenar las traducciones en una memoria caché, de modo que los accesos posteriores al bloque de la memoria del ordenador 5002 no requieren el retraso de la traducción de la dirección. Típicamente, se emplea una memoria caché 5009 entre la memoria 5002 y el procesador 5001. La memoria caché 5009 puede ser jerárquica teniendo una memoria caché grande disponible para más de una CPU y memorias caché más pequeñas, más rápidas (nivel inferior) entre la memoria caché grande y cada CPU. En algunas implementaciones, las memorias caché de nivel más bajo se dividen para proporcionar memorias cachés separadas de bajo nivel para la búsqueda de instrucciones y los accesos a los datos. En una realización, para la función de TX,

un bloque de diagnóstico de transacción (TDB) 5100 y uno o más almacenamientos intermedios 5101 pueden ser almacenados en uno o más de la memoria caché 5009 y la memoria 5002. En un ejemplo, en modo TX, los datos se almacenan inicialmente en una memoria temporal TX y, cuando finaliza el modo TX (por ejemplo, TEND más exterior), los datos en la memoria temporal son almacenados (comprometidos) en la memoria, o si se cancela, los datos en la memoria temporal son descartados.

En una realización, una unidad de búsqueda de instrucciones 5004 recupera una instrucción de la memoria 5002 mediante un cache 5009. La instrucción es descodificada en una unidad de descodificación de instrucciones 5006 y enviada (con otras instrucciones en algunas realizaciones) a la unidad o unidades de ejecución de instrucciones 5008. Típicamente se emplean varias unidades de ejecución 5008, por ejemplo, una unidad de ejecución aritmética, una unidad de ejecución de punto de flotación y una unidad de ejecución de instrucción de bifurcación. Además, en una realización de la función TX, se pueden emplear diversos controles TX 5110. La instrucción es ejecutada por la unidad de ejecución, accediendo a los operandos desde la instrucción de los registros especificados o la memoria según sea necesario. Si se va a acceder a un operando (cargado o almacenado) desde la memoria 5002, una unidad de carga / almacenamiento 5005 típicamente 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 ambos.

De acuerdo con un aspecto de la función de TX, el procesador 5001 incluye asimismo una PSW 5102 (por ejemplo, PSW de TX y/o cancelación), una profundidad de anidamiento 5104, un TDBA 5106 y uno o más registros de control 5108.

Tal como se señaló, un sistema informático incluye información en el almacenamiento local (o principal), así como también el direccionamiento, la protección y la referencia y la grabación 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. Parte del almacenamiento principal incluye ubicaciones de almacenamiento asignadas permanentemente. El almacenamiento principal proporciona al sistema un almacenamiento de datos de acceso rápido directamente direccionable. Tanto los datos como los programas deben ser cargados en el almacenamiento principal (desde los dispositivos de entrada) antes de que puedan ser procesados.

El almacenamiento principal puede incluir uno o más almacenamientos intermedios más pequeños y de acceso más rápido, a veces denominados memorias caché. Una memoria caché está asociada físicamente, en general, con una CPU o con un procesador de I/O. Los efectos, excepto en el rendimiento, de la construcción física y la utilización de medios de almacenamiento distintos no son observables, en general, por el programa.

Se pueden mantener memorias caché separadas para las instrucciones y para los operandos de datos. La información dentro de una memoria caché se mantiene en bytes contiguos en un límite integral denominado bloque de memoria caché o línea de memoria caché (o línea, dicho de manera abreviada). Un modelo puede proporcionar una instrucción EXTRACT CACHE ATTRIBUTE que devuelve el tamaño de una línea de la memoria caché en bytes. Un modelo puede proporcionar asimismo instrucciones PREFETCH DATA y PREFETCH DATA RELATIVE LONG, que efectúan la búsqueda previa de almacenamiento en la memoria caché de datos o instrucciones o la liberación de datos de la memoria caché.

El almacenamiento se ve como una larga secuencia horizontal de bits. Para la mayoría de las operaciones, los accesos al almacenamiento proceden en una secuencia de izquierda a derecha. La secuencia de bits se subdivide en unidades de ocho bits. Una unidad de ocho bits se denomina byte, que es el componente básico de todos los formatos de información. La ubicación de cada byte en el almacenamiento se identifica mediante un entero no negativo único, que es la dirección de la ubicación de ese byte o, simplemente, la dirección del byte. Las ubicaciones de bytes adyacentes tienen direcciones consecutivas, comenzando con 0 a la izquierda y siguiendo en una secuencia de izquierda a derecha. Las direcciones son enteros binarios sin signo y tienen 24, 31 o 64 bits.

La información se transmite entre el almacenamiento y una CPU o un subsistema de canales de un byte, o un grupo de bytes, a la vez. A menos que se especifique lo contrario, por ejemplo, en la Arquitectura/z, un grupo de bytes en el almacenamiento está dirigido por el byte más a la izquierda del grupo. El número de bytes en el grupo está implícito o explícitamente especificado por la operación que se realizará. Cuando se utiliza en una operación de CPU, un grupo de bytes se denomina campo. Dentro de cada grupo de bytes, por ejemplo, en la Arquitectura/z, los bits se numeran en una secuencia de izquierda a derecha. En la Arquitectura/z, los bits más a la izquierda se denominan a veces bits de "orden superior" y los bits de más a la derecha se denominan bits de "orden inferior". No obstante, los números de bits no son direcciones de almacenamiento. Solo se pueden abordar los bytes. Para operar sobre bits individuales de un byte en almacenamiento, se accede a todo el byte. Los bits en un byte están numerados de 0 a 7, de izquierda a derecha (por ejemplo, en la Arquitectura/z). Los bits en una dirección pueden estar numerados de 8 a 31 o de 40 a 63 para direcciones de 24 bits, o de 1 a 31 o de 33 a 63 para direcciones de 31 bits; están numerados de 0 a 63 para direcciones de 64 bits. En un ejemplo, los bits de 8 a 31 y de 1 a 31 se aplican a las direcciones que están en una ubicación (por ejemplo, registro) que tiene 32 bits de ancho, mientras que los bits de 40 a 63 y de 33 a 63 se aplican a las direcciones que están en una ubicación extensa de 64 bits. Dentro de cualquier otro formato de longitud fija de múltiples bytes, los bits que componen el formato se numeran

consecutivamente comenzando desde 0. Para propósitos de detección de errores, y preferiblemente para corrección, se pueden transmitir uno o más bits de verificación con cada byte o con un grupo de bytes. Dichos bits de verificación 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 de uno, dos, cuatro, ocho o dieciséis bytes. Campos mayores pueden estar implicados para algunas instrucciones. Cuando la longitud de un campo del operando de almacenamiento no está implícita, sino que se declara explícitamente, se dice que el campo tiene una longitud variable. Los operandos de longitud variable pueden variar en longitud en incrementos de un byte (o con algunas instrucciones, en múltiplos de dos bytes u otros múltiplos). Cuando la información se almacena, solo se reemplazan los contenidos de las ubicaciones de bytes que están incluidas en el campo designado, aunque el ancho de la ruta física de almacenamiento puede ser mayor que la longitud del campo que se almacena.

Ciertas unidades de información deben estar en un límite integral en el almacenamiento. Un límite se denomina integral para una unidad de información cuando su dirección de almacenamiento es un múltiplo de la longitud de la unidad en bytes. A los campos de 2, 4, 8, 16 y 32 bytes se les dan nombres especiales en un límite integral. Una media palabra es un grupo de dos bytes consecutivos en un límite de dos bytes y es el bloque básico de instrucciones. Una palabra es un grupo de cuatro bytes consecutivos en un límite de cuatro bytes. Una doble palabra es un grupo de ocho bytes consecutivos en un límite de ocho bytes. Una palabra cuádruple es un grupo de 16 bytes consecutivos en un límite de 16 bytes. Una palabra óctuple es un grupo de 32 bytes consecutivos en un límite de 32 bytes. Cuando las direcciones de almacenamiento designan medias palabras, palabras, dobles palabras, palabras cuádruples y palabras óctuples, la representación binaria de la dirección contiene uno, dos, tres, cuatro o cinco bits a la derecha, respectivamente. Las instrucciones deben estar en límites integrales de dos bytes. Los operandos de almacenamiento de la mayoría de las instrucciones no tienen requisitos de alineación de límites.

En dispositivos que implementan memorias caché separadas para instrucciones y operandos de datos, se puede experimentar un retardo significativo si el programa almacena en una línea de la memoria caché cuyas instrucciones son buscadas posteriormente, independientemente de si el almacenamiento altera las instrucciones que se buscan posteriormente.

En un ejemplo, las realizaciones pueden ser llevadas a la práctica mediante software (a veces se hace referencia al código interno, firmware, microcódigo, mili-código, pico-código y similares, cualquiera de los cuales sería coherente con una o más realizaciones). Haciendo referencia a la figura 18, el procesador 5001 del sistema anfitrión 5000 puede acceder al código del programa de software que incorpora uno o más aspectos desde los dispositivos de medios de almacenamiento a largo plazo 5011, tales como una unidad de CD-ROM, unidad de cinta o disco duro. El código del programa de software puede ser incorporado en cualquiera de una variedad de medios conocidos para su utilización con un sistema de procesamiento de datos, tal como un disquete, disco duro o CD-ROM. El código puede ser distribuido en dichos medios, o puede ser distribuido a los usuarios desde la memoria del ordenador 5002 o el almacenamiento de un sistema informático a través de una red 5010 a otros sistemas informáticos, para su utilización por los usuarios de dichos otros sistemas.

El código del programa de software incluye un sistema operativo que controla la función y la interacción de los diversos componentes del ordenador y uno o más programas de aplicación. El código de programa es localizado normalmente desde el dispositivo de medios de almacenamiento 5011 al almacenamiento 5002 del ordenador de velocidad relativamente alta donde está disponible para procesamiento por el procesador 5001. Las técnicas y métodos para incorporar código de programa de software en memoria, en medios físicos y/o software de distribución el código a través de redes es bien conocido y no se analizará más en este documento. El código de programa, cuando está creado y almacenado en un medio tangible (incluidos, entre otros, módulos de memoria electrónicos (RAM), memoria rápida, discos compactos (CD), DVD, cinta magnética y similares se denominan a menudo "producto de programa informático". El medio del producto del programa informático es típicamente legible por un circuito de procesamiento preferiblemente en un sistema informático para su ejecución por el circuito de procesamiento.

La figura 19 ilustra una estación de trabajo o un sistema de hardware de servidor representativos en el que se pueden poner en práctica una o más realizaciones. El sistema 5020 de la figura 19 comprende un sistema informático base 5021 representativo, tal como un ordenador personal, una estación de trabajo o un servidor, que incluye dispositivos periféricos opcionales. El sistema informático base 5021 incluye uno o más procesadores 5026 y un bus empleado para conectar y habilitar la comunicación entre el o los procesadores 5026 y los otros componentes del sistema 5021 de acuerdo con técnicas conocidas. El bus conecta el procesador 5026 a la memoria 5025 y el almacenamiento a largo plazo 5027 que puede incluir un disco duro (que incluye cualquiera de los medios magnéticos, CD, DVD y memoria rápida, por ejemplo) o una unidad de cinta, por ejemplo. El sistema 5021 podría incluir asimismo un adaptador de interfaz de usuario, que conecta el microprocesador 5026 a través del bus a uno o más 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, tal como una pantalla táctil, entrada digitalizada, etc. El bus conecta asimismo un dispositivo de visualización 5022, tal como una pantalla LCD o monitor, al microprocesador 5026 a través de un adaptador de pantalla.

El sistema 5021 puede comunicarse con otros ordenadores o redes de ordenadores por medio de un adaptador de red capaz de comunicarse 5028 con una red 5029. Los adaptadores de red de ejemplo son canales de comunicaciones, anillo de testigo, Ethernet o módems. Alternativamente, el sistema 5021 puede comunicarse utilizando una interfaz inalámbrica, tal como una tarjeta CDPD (datos en paquetes digitales celulares – Cellular Digital Packet Data, en inglés). El sistema 5021 puede estar asociado con dichos otros ordenadores en una red de área local (LAN) o en una red de área amplia (WAN), o el sistema 5021 puede ser un cliente en una disposición cliente / servidor con otro ordenador, etc. Todas estas configuraciones, así como el hardware y software de comunicaciones apropiados, son conocidos en la técnica.

La figura 20 ilustra una red 5040 de procesamiento de datos en la que se pueden llevar a la práctica una o más realizaciones. La red de procesamiento de datos 5040 puede incluir una pluralidad de redes individuales, tales como una red inalámbrica y una red de cable, cada una de las cuales puede incluir una pluralidad de estaciones de trabajo individuales 5041, 5042, 5043, 5044. Además, tal como resultará evidente para los expertos en la materia, se pueden incluir una o más LAN, donde una LAN puede comprender una pluralidad de estaciones de trabajo inteligentes acopladas a un procesador central.

Aun haciendo referencia a la figura 20, las redes pueden incluir asimismo ordenadores o servidores centrales, tales como un ordenador de puerta de enlace (servidor de cliente 5046) o servidor de aplicaciones (servidor remoto 5048 que puede acceder a un depósito de datos y también se puede acceder al mismo directamente desde una estación de trabajo 5045). Un ordenador de puerta de enlace 5046 sirve como punto de entrada en cada red individual. Se necesita una puerta de enlace cuando se conecta un protocolo de red a otro. La puerta de enlace 5046 se puede acoplar preferiblemente a otra red (la Internet 5047 por ejemplo) por medio de un enlace de comunicaciones. La puerta de enlace 5046 puede estar directamente acoplada asimismo a una o más estaciones de trabajo 5041, 5042, 5043, 5044 utilizando un enlace de comunicaciones. El ordenador de la puerta de enlace puede ser implementado utilizando un servidor IBM eServer de System z comercializado por International Business Machines Corporation.

Haciendo referencia simultáneamente a la figura 19 y la figura 20, el procesador 5026 del sistema 5020 puede acceder al código 5031 de programación de software que puede incorporar una o más realizaciones a partir de un medio de almacenamiento a largo plazo 5027, tal como una unidad de CD-ROM o un disco duro. El código de programación de software puede estar incorporado en cualquiera de una variedad de medios conocidos para su utilización con un sistema de procesamiento de datos, tales como un disquete, disco duro o CD-ROM. El código puede estar distribuido en dichos medios, o puede ser distribuirse a los usuarios 5050, 5051 desde la memoria o el almacenamiento de un sistema informático a través de una red a otros sistemas informáticos, para su utilización por los usuarios de dichos otros sistemas.

Alternativamente, el código de programación puede estar incorporado en la memoria 5025, y el procesador 5026 puede acceder al mismo utilizando el bus del procesador. Dicho código de programación incluye un sistema operativo que controla la función e interacción de los diversos componentes informáticos y uno o más programas de aplicación 5032. El código de programa normalmente es localizado desde el medio de almacenamiento 5027 hasta la memoria de alta velocidad 5025 donde está disponible para su procesamiento por el procesador 5026. Las técnicas y métodos para incorporar código de programación de software en memoria, en medios físicos, y/o distribuir código de software a través de redes son bien conocidos y no se analizarán más en este documento. El código de programa, cuando se crea y almacena en un medio tangible (que incluye, entre otros, un módulo de memoria electrónica (RAM), una memoria rápida, discos compactos (CD), DVD, una cinta magnética y similares, a menudo se denomina "producto de programa informático". El medio de producto de programa informático es típicamente legible por un circuito de procesamiento, preferiblemente en un sistema informático, para su ejecución 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 de más bajo nivel (L1 o nivel uno) y el almacén principal (memoria principal) es la memoria caché de más alto nivel (L3 si hay 3 niveles). La memoria caché de más bajo nivel a menudo se divide en una memoria caché de instrucciones (I-Cache) que contiene las instrucciones de la máquina que serán ejecutadas y una memoria caché de datos (D-Cache) que contiene los operandos de los datos.

Haciendo referencia a la figura 21, se representa una realización a modo de ejemplo del procesador para el procesador 5026. Típicamente, se emplean uno o más niveles de memoria caché 5053 para almacenar bloques de memoria, a fin de mejorar el rendimiento del procesador. La memoria caché 5053 es una memoria temporal de alta velocidad que contiene líneas de memoria caché de datos de memoria que probablemente se utilizarán. Las líneas de memoria caché típicas son 64, 128 o 256 bytes de datos de la memoria. A menudo se emplean memorias caché separadas para almacenar instrucciones en la memoria caché y para almacenar datos en la memoria caché. La coherencia de la memoria caché (sincronización de copias de las líneas en la memoria y las memorias caché) a menudo se proporciona mediante diversos algoritmos "espía" (snoop, en inglés) bien conocidos en la técnica. El almacenamiento en la memoria principal 5025 de un sistema de procesador a menudo se denomina memoria caché. En un sistema de procesador que tiene 4 niveles de memoria caché 5053, el almacenamiento principal 5025 a veces se denomina memoria caché de nivel 5 (L5), ya que típicamente es más rápido y solo contiene una parte del almacenamiento no volátil (DASD, cinta, etc.) que está disponible para un sistema informático. El almacenamiento

principal 5025 "almacena en memoria caché" páginas de datos localizados dentro y fuera del almacenamiento principal 5025 por el sistema operativo.

Un contador de programa (contador de instrucciones) 5061 realiza un seguimiento de la dirección de la instrucción actual a ejecutar. Un contador de programa en un procesador de Arquitectura/z es de 64 bits y puede estar truncado a 31 o 24 bits para soportar los límites de direccionamiento anteriores. Un contador de programa normalmente está integrado en una PSW (palabra de estado del programa) de un ordenador de manera que persiste durante el cambio de contexto. De este modo, un programa en ejecución que tenga un valor de contador de programa, puede ser interrumpido, por ejemplo, por el sistema operativo (cambio de contexto del entorno del programa al entorno del sistema operativo). La PSW del programa mantiene el valor del contador del programa mientras el programa no está activo, y el contador del programa (en la PSW) del sistema operativo se utiliza mientras se está ejecutando el sistema operativo. Típicamente, el contador del programa se incrementa en una cantidad igual al número de bytes de la instrucción actual. Las instrucciones RISC (cálculo de conjunto reducido de instrucciones – Reduced Instruction Set Computing, en inglés) suelen ser de longitud fija, mientras que las instrucciones CISC (cálculo de conjunto complejo de instrucciones – Complex Instruction Set Computing, en inglés) suelen ser de longitud variable. Las instrucciones de IBM de Arquitectura/z son instrucciones CISC que tienen una longitud de 2, 4 o 6 bytes. El contador de programa 5061 se modifica mediante una operación de cambio de contexto o una operación de bifurcación de una instrucción de bifurcación, por ejemplo. En una operación de cambio de contexto, el valor actual del contador del programa se guarda en la palabra de estado del programa junto con otra información de estado sobre el programa que se está ejecutando (tal como códigos de condición) y se carga un nuevo valor de contador de programa que apunta a una instrucción de un nuevo módulo de programa para ser ejecutado. Se lleva a cabo una operación de bifurcación para permitir que el programa tome decisiones o realice un bucle dentro del programa cargando el resultado de la instrucción de bifurcación en el contador de programa 5061.

Típicamente, se emplea una unidad de búsqueda de instrucción 5055 para buscar instrucciones en nombre del procesador 5026. La unidad de búsqueda busca "instrucciones secuenciales siguientes", instrucciones de destino de instrucciones tomadas de una bifurcación, o primeras instrucciones de un programa después de un cambio de contexto. Las unidades de búsqueda de Instrucción modernas a menudo emplean técnicas de búsqueda adelantada para buscar especulativamente de manera adelantada las instrucciones en función de la probabilidad de que se puedan utilizar las instrucciones de búsqueda adelantada. Por ejemplo, una unidad de búsqueda puede buscar 16 bytes de instrucción que incluyen la siguiente instrucción secuencial y bytes adicionales de instrucciones secuenciales adicionales.

Las instrucciones buscadas son ejecutadas a continuación, por el procesador 5026. En una realización, la instrucción buscada o las instrucciones buscadas son pasadas a una unidad de envío 5056 de la unidad de búsqueda. La unidad de envío descodifica la instrucción o instrucciones y reenvía la información acerca de la instrucción descodificada o las instrucciones descodificadas a las unidades 5057, 5058, 5060 apropiadas. Una unidad de ejecución 5057 típicamente recibirá información acerca de instrucciones aritméticas descodificadas de la unidad de búsqueda de instrucción 5055, y realizará operaciones aritméticas sobre operandos de acuerdo con el código de operación de la instrucción. Los operandos son proporcionados a la unidad de ejecución 5057 preferiblemente desde la memoria 5025, los registros estructurados 5059 o desde un campo inmediato de la instrucción que se está ejecutando. Los resultados de la ejecución, cuando se almacenan, se almacenan en la memoria 5025, en los registros 5059 o en otro hardware de la máquina (como registros de control, registros PSW y similares).

Las direcciones virtuales son transformadas en direcciones reales utilizando la traducción dinámica de direcciones 5062 y, opcionalmente, utilizando la traducción de registros de acceso 5063.

Un procesador 5026 tiene típicamente una o más unidades 5057, 5058, 5060 para ejecutar la función de la instrucción. Haciendo referencia a la figura 22A, una unidad de ejecución 5057 puede comunicarse 5081 con registros generales estructurados 5059, una unidad de descodificación / envío 5056, una unidad de almacenamiento de carga 5060 y otras unidades de procesador 5065 por medio de la lógica de interfaz 5071. Una unidad de ejecución 5057 puede emplear varios circuitos de registro 5067, 5068, 5069 para contener información sobre la que operará la unidad de lógica aritmética (ALU – Arithmetic Logic Unit, en inglés) 5066. La ALU realiza operaciones aritméticas tales como sumar, restar, multiplicar y dividir, así como funciones lógicas tales como y, o y o exclusiva (XOR), rotar y desplazar. Preferiblemente, la ALU soporta operaciones especializadas que dependen del diseño. Otros circuitos pueden proporcionar otras funciones estructuradas 5072 que incluyen códigos de condición y lógica de soporte de búsqueda, por ejemplo. Típicamente, el resultado de una operación de ALU está contenida en un circuito de registro de salida 5070 que puede enviar el resultado a una variedad de otras funciones de procesamiento. Existen muchas disposiciones de unidades procesadoras, la presente descripción solo pretende proporcionar una comprensión representativa de una realización.

Una instrucción ADD (sumar), por ejemplo, se ejecutaría en una unidad de ejecución 5057 que tiene una funcionalidad aritmética y lógica, mientras que una instrucción de coma flotante, por ejemplo, se ejecutaría en una ejecución de coma flotante que tiene capacidad de coma flotante especializada. Preferiblemente, una unidad de ejecución opera en operandos identificados por una instrucción realizando una función definida de código de

operación sobre los operandos. Por ejemplo, una instrucción ADD puede ser ejecutada por una unidad de ejecución 5057 sobre operandos encontrados en dos registros 5059 identificados por los campos de registro de la instrucción.

La unidad de ejecución 5057 realiza la suma aritmética sobre dos operandos y almacena el resultado en un tercer operando, donde el tercer operando puede ser un tercer registro o uno de los dos registros fuente. La unidad de ejecución utiliza preferiblemente una unidad de lógica aritmética (ALU) 5066 que es capaz de realizar una variedad de funciones lógicas tales como desplazar, rotar, y, o y XOR, así como una variedad de funciones algebraicas, que incluyen cualquiera de sumar, restar, multiplicar, dividir. Algunas ALU 5066 están diseñadas para operaciones escalares, y algunas para coma flotante. Los datos pueden ser Big Endian (donde el byte menos significativo está en la dirección del byte más alto) o Little Endian (donde el byte menos significativo está en la dirección del byte más bajo) dependiendo de la arquitectura. La Arquitectura/z de IBM es Big Endian. Los campos con signo pueden ser signo y magnitud, complemento a 1 o complemento a 2, dependiendo de la arquitectura. Un número de complemento a 2 es ventajoso porque la ALU no necesita diseñar una capacidad de resta, ya que un valor negativo o positivo en el complemento a 2 requiere solo una suma dentro de la ALU. Los números se describen comúnmente en forma abreviada, donde un campo de 12 bits define una dirección de un bloque de 4.096 bytes y se describe comúnmente como un bloque de 4 Kbytes (kilobyte), por ejemplo.

Haciendo referencia a la figura 22B, la información de la instrucción de bifurcación para ejecutar una instrucción de bifurcación se envía típicamente a una unidad de bifurcación 5058 que, a menudo, emplea un algoritmo de predicción de bifurcación tal como una tabla de historial de bifurcaciones 5082 para predecir el resultado de la bifurcación antes de que se completen otras operaciones condicionales. El objetivo de la instrucción de bifurcación actual se buscará y se ejecutará especulativamente antes de que se completen las operaciones condicionales. Cuando se completan las operaciones condicionales, las instrucciones de bifurcación ejecutadas especulativamente se completan o se descartan en función de las condiciones de la operación condicional y del resultado especulado. Una instrucción de bifurcación típica puede probar códigos de condición y bifurcarse a una dirección de destino si los códigos de condición cumplen el requisito de la bifurcación de la instrucción de bifurcación, una dirección de destino puede calcularse sobre la base de varios números, incluidos los encontrados en campos de registro o en un campo inmediato de la instrucción, por ejemplo. La unidad de bifurcación 5058 puede emplear una ALU 5074 que tiene 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, una unidad de envío de descodificación 5056 u otros circuitos 5073, por ejemplo.

La ejecución de un grupo de instrucciones puede ser interrumpida por una variedad de motivos que incluyen un cambio de contexto iniciado por un sistema operativo, una excepción de programa o un error que causa un cambio de contexto, una señal de interrupción de I/O que causa un cambio de contexto o una actividad de subprocesamiento múltiple de una pluralidad de programas (en un entorno de múltiples subprocesos), por ejemplo. Preferiblemente, una acción de cambio de contexto guarda la información de estado sobre un programa actualmente en ejecución y, a continuación, carga la información de estado acerca de otro programa que se invoca. La información de estado puede estar guardada en registros de hardware o en memoria, por ejemplo. La información de estado comprende, preferiblemente, un valor de contador de programa que apunta a una siguiente instrucción a ejecutar, códigos de condición, información de traducción de memoria y contenido de registro estructurado. Una actividad de cambio de contexto puede ser ejercida por circuitos de hardware, programas de aplicación, programas de sistema operativo o código de firmware (microcódigo, picocódigo o código interno autorizado (LIC – Licensed Internal Code, en inglés)), solos o en combinación.

Un procesador accede a los operandos de acuerdo con los métodos definidos por la instrucción. La instrucción puede proporcionar un operando inmediato que utilizan el valor de una parte de la instrucción, puede proporcionar uno o más campos de registro que apuntan explícitamente 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 de operación. La instrucción puede utilizar ubicaciones de memoria para operandos. Una ubicación de memoria de un operando puede ser proporcionada por un registro, un campo inmediato o una combinación de registros y campo inmediato, tal como se muestra como ejemplo en la función de desplazamiento largo de Arquitectura/z, en la que la instrucción define un registro 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. La ubicación en este documento implica, típicamente, una ubicación en la memoria principal (almacenamiento principal) a menos que se indique lo contrario.

Haciendo referencia a la figura 22C, un procesador accede al almacenamiento utilizando una unidad 5060 de carga / almacenamiento. La unidad 5060 de carga / almacenamiento puede realizar una operación de carga obteniendo la dirección del operando objetivo en la memoria 5053, y cargando el operando en un registro 5059 o en otra ubicación 5053 de memoria, o puede realizar una operación de almacenamiento obteniendo la dirección del operando objetivo en la memoria 5053 y almacenando datos obtenidos de un registro 5059 u otra ubicación 5053 de memoria en la ubicación del operando objetivo en la memoria 5053. La unidad 5060 de carga / almacenamiento puede ser especulativa y puede acceder a la memoria en una secuencia que está fuera de servicio con respecto a la secuencia de instrucciones; no obstante, la unidad de carga / almacenamiento 5060 debe mantener la apariencia de que los programas fueron ejecutados en orden. Una unidad de carga / almacenamiento 5060 se puede comunicar 5084 con los registros generales 5059, la unidad de descodificación / envío 5056, la interfaz de memoria / memoria caché

5053 u otros elementos 5083, y comprende varios circuitos de registro 5086, 5087, 5088 y 5089, ALU 5085 y lógica de control 5090, para calcular direcciones de almacenamiento y para proporcionar la secuencia de tubería para mantener las operaciones en orden. Algunas operaciones pueden estar fuera de servicio, pero la unidad de carga / almacenamiento proporciona la funcionalidad para hacer que las operaciones fuera de servicio aparezcan en el programa como si se hubieran realizado en orden, como es bien conocido en la técnica.

Preferentemente, las direcciones que un programa de aplicación "ve" a menudo se denominan direcciones virtuales. Las direcciones virtuales a veces se denominan "direcciones lógicas" y "direcciones efectivas". Estas direcciones virtuales son virtuales en el sentido de que son redirigidas a la ubicación de la memoria física mediante una variedad de tecnologías de traducción dinámica de direcciones (DAT) que incluyen, entre otros, el prefijo de una dirección virtual con un valor de compensación, que traducen la dirección virtual por medio de una o más tablas de traducción, comprendiendo las tablas de traducción, preferiblemente, por lo menos una tabla de segmentos y una tabla de páginas, sola o en combinación, teniendo la tabla de segmentos, preferiblemente, una entrada que apunta a la tabla de páginas. En Arquitectura/z, 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 páginas opcional. El rendimiento de la traducción de la dirección a menudo se mejora utilizando una memoria temporal de búsqueda de traducción (TLB) que comprende entradas que reproducen (map, en inglés) una dirección virtual en una ubicación de memoria física asociada. Las entradas se crean cuando la DAT traduce una dirección virtual utilizando las tablas de traducción. La utilización posterior de la dirección virtual puede utilizar la entrada del TLB rápida en lugar de acceder a la tabla de traducción secuencial lenta. El contenido del TLB puede ser gestionado mediante una variedad de algoritmos de reemplazo que incluyen LRU (menos recientemente utilizado – Least Recently Used, en inglés).

En el caso en el que el procesador es un procesador de un sistema de múltiples procesadores, cada procesador tiene la responsabilidad de mantener los recursos compartidos, como I/O, memorias caché, TLB y memoria, enclavados para garantizar la coherencia. Por lo general, las tecnologías "espía" se utilizarán para mantener la coherencia de la memoria caché. En un entorno espía, cada línea de la memoria caché se puede marcar como estando en cualquier estado compartido, estado exclusivo, estado modificado, estado no válido y similares para facilitar el intercambio.

Las unidades de I/O 5054 (figura 21) proporcionan al procesador medios de conexión a dispositivos periféricos que incluyen cinta, disco, impresoras, pantallas y redes, por ejemplo. Las unidades de I/O a menudo se presentan en el programa informático mediante controladores de software. En ordenadores principales, tal como el System z de IBM®, los adaptadores de canal y los adaptadores de sistema abierto son unidades de I/O del ordenador principal que proporcionan las comunicaciones entre el sistema operativo y los dispositivos periféricos.

Además, otros tipos de entornos informáticos pueden beneficiarse de uno o más aspectos. Como ejemplo, un entorno puede incluir un emulador (por ejemplo, software u otros mecanismos de emulación), en el que una arquitectura particular (incluyendo, por ejemplo, ejecución de instrucciones, funciones estructuradas, tales como traducción de direcciones y registros estructurados) o un subconjunto de los mismos se emula (por ejemplo, en un sistema informático nativo que tiene un procesador y memoria). En dicho entorno, una o más funciones de emulación del emulador pueden implementar una o más realizaciones, incluso aunque un ordenador que ejecuta el emulador pueda tener una arquitectura diferente de las capacidades que se emulan. Como ejemplo, en el modo de emulación, la instrucción u operación específica que se está emulando se descodifica, y se construye una función de emulación apropiada para implementar la instrucción u operación individual.

En un entorno de emulación, un ordenador anfitrión incluye, por ejemplo, una memoria para almacenar instrucciones y datos; una unidad de búsqueda de instrucciones para buscar instrucciones en la memoria y, opcionalmente, proporcionar almacenamiento en una memoria temporal local para la instrucción buscada; una unidad de descodificación de instrucciones para recibir las instrucciones buscadas y para determinar el tipo de instrucciones que se han buscado; y una unidad de ejecución de instrucción para ejecutar las instrucciones. La ejecución puede incluir cargar datos en un registro de la memoria; almacenar datos en la memoria desde un registro; o realizar algún tipo de operación aritmética o lógica, según lo determinado por la unidad de descodificación. En un ejemplo, cada unidad se implementa en software. Por ejemplo, las operaciones que realizan las unidades se implementan como una o más subrutinas dentro del software del emulador.

Más concretamente, en un ordenador principal, los programadores utilizan instrucciones de máquina estructuradas, por lo general, hoy en día, programadores en "C", a menudo por medio de una aplicación de compilación. Estas instrucciones almacenadas en el medio de almacenamiento pueden ser ejecutadas de manera nativa en un servidor de Arquitectura/z IBM®, o, alternativamente, en máquinas que ejecutan otras arquitecturas. Pueden ser emuladas en los servidores de ordenador principal de IBM® existentes y futuros y en otras máquinas de IBM® (por ejemplo, servidores Power Systems y servidores System x). Se pueden ejecutar en máquinas que ejecutan Linux en una amplia variedad de máquinas utilizando hardware fabricado por IBM®, Intel®, AMD y otros. Además de la ejecución en ese hardware bajo Arquitectura/z, se puede utilizar Linux, así como máquinas que utilizan emulación mediante Hercules, UMX o FSI (Fundamental Software, Inc), donde, en general, la ejecución está en un modo de emulación. En el modo de emulación, un procesador nativo ejecuta el software de emulación para emular la arquitectura de un procesador emulado.

El procesador nativo normalmente ejecuta un software de emulación que comprende firmware o un sistema operativo nativo para realizar la emulación del procesador emulado. El software de emulación es responsable de buscar y ejecutar las 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 las instrucciones. El software de emulación puede buscar una o más instrucciones de máquina emuladas a la vez y convertir una o más instrucciones de máquina emuladas en un grupo correspondiente de instrucciones nativas de máquina para su ejecución por el procesador nativo. Estas instrucciones convertidas pueden ser almacenadas en memoria caché de manera que se pueda lograr una conversión más rápida. No obstante, el software de emulación debe mantener las reglas de arquitectura de la arquitectura del procesador emulado para garantizar 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 incluye, entre otros, registros de control, registros de propósito general, registros de coma flotante, función de traducción de direcciones dinámicas incluyendo tablas de segmentos y tablas de páginas, por ejemplo, mecanismos de interrupción, mecanismos de conmutación por contexto, relojes con Hora del día (TOD – Time of Day, en inglés) e interfaces estructuradas para los subsistemas de I/O, de tal manera que un sistema operativo o un programa de aplicación diseñado para ser ejecutado en el procesador emulado pueda ser ejecutado en el procesador nativo que tiene 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 es implementada, por ejemplo, en una subrutina o controlador "C", o algún otro método de proporcionar un controlador para el hardware específico como estará entre las habilidades de los expertos en la materia tras comprender la descripción de la realización preferida. Diversas patentes de emulación de software y hardware que incluyen, pero no están limitadas a, la Patente de Estados Unidos Nº 5.551.013, titulada "Multiprocessor for Hardware Emulation", por Beausoleil et al.; y las Cartas de Patente de U.S.A. Nº 6.009.261, tituladas "Preprocessing of Stored Target Routines for Emulating Incompatible Instructions on a Target Processor", por Scalzi et al; y las Cartas de Patente de U.S.A Nº 5.574.873, titulada "Decoding Guest Instruction to Directly Access Emulation Routines that emulate the Guest Instructions", por Davidian et al; y las Cartas de Patente de U.S.A Nº 6.308.255, tituladas "Symmetrical Multiprocessing Bus and Chipset Used for Coprocessor Support Allowing Non-Native Code to run in a System", por Gorishek et al; y las Cartas de Patente de U.S.A Nº 6.463.582, tituladas "Dynamic Optimizing Object Code Translator for Architecture Emulation and Dynamic Optimizing Object Code Translation Method", por Lethin et al; y las Cartas de Patente de U.S.A Nº 5.790.825, titulada "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 maneras conocidas de lograr la emulación de un formato de instrucción diseñado para una máquina diferente para una máquina objetivo disponible para los expertos en la técnica.

En la figura 23, se proporciona un ejemplo de un sistema informático anfitrión emulado 5092 que emula un sistema informático anfitrión 5000' de una arquitectura de servidor. En el sistema informático anfitrión emulado 5092, el procesador anfitrión (CPU) 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 la memoria 5094 accesible para el procesador de emulación 5093. En la realización de ejemplo, la memoria 5094 está dividida en una parte de la memoria del ordenador anfitrión 5096 y una parte de las rutinas de emulación 5097. La memoria del ordenador anfitrión 5096 está disponible para programas del ordenador anfitrión 5092 emulados de acuerdo con la arquitectura del ordenador anfitrión. El procesador de emulación 5093 ejecuta instrucciones nativas de un conjunto de instrucciones estructuradas de una arquitectura distinta a la del procesador emulado 5091, siendo obtenidas las instrucciones nativas de la memoria de rutinas de emulación 5097, y puede acceder a una instrucción de anfitrión para su ejecución desde un programa en la memoria del ordenador anfitrión 5096, empleando una o más instrucciones obtenidas en una rutina de secuencia y acceso / descodificación que puede descodificar la instrucción de anfitrión o las instrucciones de anfitrión a la que se ha accedido o a las que se han accedido para determinar una rutina de ejecución de instrucción nativa para emular la función de la instrucción de anfitrión a la que se ha accedido. Otras funciones definidas para la arquitectura 5000' del sistema informático anfitrión pueden ser emuladas por rutinas de funciones estructuradas, incluyendo funciones tales como registros de propósito general, registros de control, traducción dinámica de direcciones y soporte del subsistema I/O y memoria caché del procesador, por ejemplo. Las rutinas de emulación se pueden beneficiar asimismo de las funciones disponibles en el procesador de emulación 5093 (tales como registros generales y traducción dinámica de direcciones virtuales) para mejorar el rendimiento de las rutinas de emulación. Asimismo, se puede proporcionar un hardware especial y motores de desconexión - carga para ayudar al procesador 5093 a emular la función del ordenador anfitrión 5000'.

La terminología utilizada en el presente documento tiene el propósito de describir solamente realizaciones particulares, y no pretende ser limitativa. Tal como se utiliza en el presente documento, las formas singulares “un“, “una“, “el” y “la“ pretenden incluir también las formas plurales, a menos que el contexto indique claramente lo contrario. Se comprenderá además que los términos "comprende", “comprenden”, y/o "que comprende", “que comprenden”, cuando se utilizan en esta especificación, especifican la presencia de características, enteros, etapas,

operaciones, elementos y/o componentes indicados, pero no excluyen la presencia o adición de una o más de otras características, enteros, etapas, operaciones, elementos, componentes y/o grupos de los mismos.

La descripción de una o más realizaciones se ha presentado con fines de ilustración y descripción, pero no pretende ser exhaustiva o estar limitada a la forma descrita. Muchas modificaciones y variaciones serán evidentes para los expertos en la materia. La realización se eligió y describió con el fin de explicar mejor los diversos aspectos y la aplicación práctica, y para permitir que otros expertos en la técnica entendieran diversas realizaciones con diversas modificaciones, que son adecuadas para la utilización particular contemplada.

REIVINDICACIONES

1. Un producto de programa informático para proporcionar información de diagnóstico acerca de cancelaciones de transacción, comprendiendo el producto de programa informático:

un medio de almacenamiento legible por ordenador, legible por un circuito de procesamiento y que almacena instrucciones para su ejecución por parte del circuito de procesamiento para llevar a cabo un método que comprende:

detectar, mediante un procesador, una cancelación de una transacción, comprendiendo la transacción una o más instrucciones; siendo la transacción una transacción anidada; en el que una condición de cancelación a cualquier profundidad de anidamiento provoca la interrupción de todos los niveles de la transacción; y retardando la transacción de manera efectiva el compromiso de los almacenamientos de transacción a la memoria principal hasta la finalización de una transacción más exterior:

determinar, mediante el procesador, si la información de diagnóstico debe ser almacenada en un bloque de diagnóstico de transacción (900) en base a la cancelación; y

en base a la determinación que indica que la información de diagnóstico debe ser almacenada, almacenar la información de diagnóstico en el bloque de diagnóstico de transacción, incluyendo la información de diagnóstico una dirección (912) de una instrucción correspondiente a la transacción que fue cancelada, dependiendo la dirección de la instrucción de un motivo para la cancelación de la transacción, el motivo proporcionado en un código de cancelación (908), y en el que: en base al código de cancelación que tiene un primer valor de uno o más primeros valores, la información de diagnóstico incluye una dirección de una instrucción que se estaba ejecutando cuando la cancelación fue detectada; en base al código de cancelación que tiene un segundo valor de uno o más segundos valores y a una condición de excepción del programa que no anula, la información de diagnóstico incluye una dirección de una instrucción que es posterior a la instrucción que se estaba ejecutando cuando se detectó la cancelación; y en base al código de cancelación que tiene un tercer valor de uno o más terceros valores, la información de diagnóstico incluye una dirección de una instrucción que es anterior o posterior a la instrucción que se estaba ejecutando cuando se detectó la cancelación.

2. El producto de programa informático de la reivindicación 1, en el que existe una pluralidad de tipos de bloques de diagnóstico de transacción, y la determinación comprende verificar (1502, 1504, 1506, 1508, 1510, 1512) si se debe realizar un almacenamiento en uno o más tipos de bloques de diagnóstico de transacción, en base a la cancelación.

3. El producto de programa informático de la reivindicación 2, en el que la pluralidad de tipos de bloques de diagnóstico de transacción comprende: un bloque de diagnóstico de transacción especificado por programa especificado mediante una instrucción de inicio de transacción no restringida, un bloque de diagnóstico de transacción de interrupción de programa para ser utilizado para cancelaciones debidas a una interrupción, y un bloque de diagnóstico de transacción de interceptación para ser utilizado para cancelaciones debidas a cualquier condición de excepción del programa que dé lugar a una interceptación.

4. El producto de programa informático de la reivindicación 2, en el que la verificación indica una pluralidad de bloques de diagnóstico de transacción en los que se debe realizar un almacenamiento, en base a la cancelación.

5. El producto de programa informático de cualquier reivindicación precedente, en el que el bloque de diagnóstico de transacción comprende además uno o más de: una profundidad de anidamiento de transacción (906), un código de cancelación de transacción (908), un testigo de conflicto (910), uno o más parámetros de interrupción de programa (918), contenidos de uno o más registros de propósito general (930) en el momento de la cancelación, e información de diagnóstico que depende del modelo (924).

6. El producto de programa informático de la reivindicación 5, en el que el bloque de diagnóstico de transacción comprende además uno o más de: una identificación de acceso a excepción (914), un código de excepción de datos (916), una identificación de interrupción de programa (918), una identificación de excepción de transacción (920), y una dirección de evento de ruptura (922).

7. El producto de programa informático de la reivindicación 5, en el que el método comprende además ejecutar una instrucción de cancelación de transacción para cancelar la transacción, especificando la instrucción de cancelación de transacción el código de cancelación de la transacción.

8. El producto de programa informático de la reivindicación 1, en el que la determinación comprende determinar si se proporciona una dirección válida del bloque de diagnóstico de la transacción en una instrucción de inicio de transacción, en el que, en base a que se proporciona una dirección válida del bloque de diagnóstico de la transacción, la información de diagnóstico es almacenada en un bloque de diagnóstico de transacción especificado por el programa.

9. El producto de programa informático de la reivindicación 8, en el que la instrucción de inicio de transacción inicia una transacción no restringida, siendo la transacción no restringida una transacción más exterior.

10. El producto de programa informático de la reivindicación 9, en el que determinar si se proporciona una dirección válida del bloque de diagnóstico de la transacción comprende verificar un campo de base de la instrucción de inicio de transacción de la transacción más exterior, en el que un campo de base distinto de cero indica una dirección válida del bloque de diagnóstico de la transacción.

11. El producto de programa informático de la reivindicación 1, en el que existen una pluralidad de tipos de bloques de diagnóstico de transacción, y en el que el almacenamiento comprende determinar en cuál de la pluralidad de tipos de bloques de diagnóstico se debe realizar el almacenamiento, comprendiendo la determinación:

verificar si una dirección válida del bloque de diagnóstico de la transacción es proporcionada en una instrucción de inicio de transacción y, en base a la verificación, indicar una dirección válida del bloque de diagnóstico de la transacción, indicando la determinación un bloque de diagnóstico de transacción especificado por el programa en el que se debe realizar el almacenamiento;

verificar si la cancelación se debe a una interrupción y, en base a que la cancelación se debe a una interrupción, indicando la determinación un bloque de diagnóstico de transacción de interrupción del programa en el que se debe realizar el almacenamiento; y

verificar si la cancelación se debe a una condición de interceptación, y en base a que la cancelación se debe a la condición de interceptación, indicando la determinación un bloque de diagnóstico de transacción de interceptación del programa en el que se debe realizar el almacenamiento.

12. Un sistema informático para proporcionar información de diagnóstico acerca de la cancelación de la transacción, comprendiendo el sistema informático:

una memoria; y

un procesador en comunicación con la memoria, en el que el sistema informático está configurado para llevar a cabo un método, comprendiendo dicho método:

detectar, mediante un procesador, una cancelación de una transacción, comprendiendo la transacción una o más instrucciones; siendo la transacción una transacción anidada; en el que una condición de cancelación a cualquier profundidad de anidamiento provoca la interrupción de todos los niveles de la transacción; y la transacción retarda de manera efectiva el compromiso de los almacenamientos de transacciones a la memoria principal hasta la finalización de una transacción más exterior,

determinar, mediante el procesador, si la información de diagnóstico debe ser almacenada en un bloque de diagnóstico de transacción (900) en base a la cancelación; y

en base a que la determinación indica que la información de diagnóstico debe ser almacenada, almacenar la información de diagnóstico en el bloque de diagnóstico de transacción, incluyendo la información de diagnóstico una dirección (912) de una instrucción correspondiente a la transacción que fue cancelada, dependiendo la dirección de la instrucción de un motivo para la cancelación de la transacción, siendo el motivo proporcionado en un código de cancelación (908), y en el que: en base al código de cancelación que tiene un primer valor de uno o más primeros valores, la información de diagnóstico incluye una dirección de una instrucción que se estaba ejecutando cuando la cancelación fue detectada; en base al código de cancelación que tiene un segundo valor de uno o más segundos valores y una condición de excepción del programa que no anula, la información de diagnóstico incluye una dirección de una instrucción que es posterior a la instrucción que se estaba ejecutando cuando la cancelación fue detectada; y en base al código de cancelación que tiene un tercer valor de uno o más valores de terceros, la información de diagnóstico incluye una dirección de una instrucción que es anterior o posterior a la instrucción que se estaba ejecutando cuando la cancelación fue detectada.

13. El sistema informático de la reivindicación 12, en el que el bloque de diagnóstico de transacción comprende además uno o más de: una profundidad de anidamiento de la transacción, un código de cancelación de transacción, un testigo de conflicto, uno o más parámetros de interrupción del programa, contenidos de uno o más registros de propósito general en el momento de la cancelación y la información de diagnóstico que depende del modelo.

14. Sistema informático según la reivindicación 13, en el que el bloque de transacción comprende además uno o más de entre: una identificación de acceso de excepción, un código de excepción de datos, una identificación de interrupción del programa, una identificación de excepción de transacción y una dirección del evento de ruptura.

15. El sistema informático de la reivindicación 13, en el que el método comprende además ejecutar una instrucción de cancelación de transacción para cancelar la transacción, especificando la instrucción de cancelación de transacción el código de cancelación de la transacción.

16. El sistema informático de la reivindicación 12, en el que la determinación comprende determinar si se proporciona una dirección válida del bloque de diagnóstico de la transacción en una instrucción de inicio de transacción, en el que en base a que se proporciona una dirección válida del bloque de diagnóstico de la transacción, la información de diagnóstico es almacenada en un bloque de diagnóstico de transacción especificado por el programa, iniciando la instrucción de inicio de transacción una transacción no restringida, siendo la transacción no restringida una transacción más exterior, y en el que determinar si se proporciona una dirección válida del bloque de diagnóstico de la transacción comprende verificar un campo de base de la instrucción de inicio de transacción de la transacción más exterior, en el que un campo de base distinto de cero indica una dirección válida del bloque de diagnóstico de la transacción.

17. El sistema informático de la reivindicación 12, en el que existen una pluralidad de tipos de bloques de diagnóstico de transacción, y en el que el almacenamiento comprende determinar en cuál de la pluralidad de tipos de bloques de diagnóstico se debe realizar un almacenamiento, comprendiendo la determinación:

verificar si una dirección válida del bloque de diagnóstico de la transacción es proporcionada en una instrucción de inicio de transacción y, en base a la verificación, indicar una dirección válida del bloque de diagnóstico de la transacción, indicando la determinación un bloque de diagnóstico de transacción especificado por el programa en el que se debe realizar el almacenamiento;

verificar si la cancelación se debe a una interrupción y, en base a que la cancelación se debe a una interrupción, indicando la determinación un bloque de diagnóstico de transacción de interrupción del programa en el que se debe realizar el almacenamiento; y

verificar si la cancelación se debe a una condición de interceptación, y en base a que la cancelación se debe a la condición de interceptación, indicando la determinación un bloque de diagnóstico de transacción de interceptación del programa en el que se debe realizar el almacenamiento.

18. Un método para proporcionar información de diagnóstico acerca de cancelaciones de transacciones, comprendiendo el método:

detectar, mediante un procesador, una cancelación de una transacción, comprendiendo la transacción una o más instrucciones; siendo la transacción una transacción anidada; en el que una condición de cancelación a cualquier profundidad de anidamiento provoca la interrupción de todos los niveles de la transacción; y retardando la transacción de manera efectiva el compromiso de los almacenamientos de transacción a la memoria principal hasta la finalización de una transacción más exterior:

determinar, mediante el procesador, si la información de diagnóstico debe ser almacenada en un bloque de diagnóstico de transacción (900) en base a la cancelación; y

en base a la determinación que indica que la información de diagnóstico debe ser almacenada, almacenar la información de diagnóstico en el bloque de diagnóstico de transacción, incluyendo la información de diagnóstico una dirección (912) de una instrucción correspondiente a la transacción que fue cancelada, dependiendo la dirección de la instrucción de un motivo para la cancelación de la transacción, el motivo proporcionado en un código de cancelación (908), y en el que: en base al código de cancelación que tiene un primer valor de uno o más primeros valores, la información de diagnóstico incluye una dirección de una instrucción que se estaba ejecutando cuando la cancelación fue detectada; en base al código de cancelación que tiene un segundo valor de uno o más segundos valores y a una condición de excepción del programa que no anula, la información de diagnóstico incluye una dirección de una instrucción que es posterior a la instrucción que se estaba ejecutando cuando se detectó la cancelación; y en base al código de cancelación que tiene un tercer valor de uno o más terceros valores, la información de diagnóstico incluye una dirección de una instrucción que es anterior o posterior a la instrucción que se estaba ejecutando cuando se detectó la cancelación.

19. El método de la reivindicación 18, en el que la determinación comprende determinar si se proporciona una dirección válida del bloque de diagnóstico de la transacción en una instrucción de inicio de transacción, en el que en base a que se proporciona una dirección válida del bloque de diagnóstico de la transacción, la información de diagnóstico es almacenada en un bloque de diagnóstico de transacción especificado por el programa, iniciando la instrucción de inicio de transacción una transacción no restringida, siendo la transacción no restringida una transacción más exterior, y en el que determinar si se proporciona una dirección válida del bloque de diagnóstico de la transacción comprende verificar un campo de base de la instrucción de inicio de transacción de la transacción más exterior, en el que un campo de base distinto de cero indica una dirección válida del bloque de diagnóstico de la transacción.

20. El método según la reivindicación 18, en el que existen una pluralidad de tipos de bloques de diagnóstico de transacción, y en el que el almacenamiento comprende determinar en cuál de la pluralidad de tipos de bloques de diagnóstico se debe realizar el almacenamiento, comprendiendo la determinación:

verificar si una dirección válida del bloque de diagnóstico de la transacción es proporcionada en una instrucción de inicio de transacción y, en base a la verificación, indicar una dirección válida del bloque de diagnóstico de la transacción, indicando la determinación un bloque de diagnóstico de transacción especificado por el programa en el que se debe realizar el almacenamiento;

verificar si la cancelación se debe a una interrupción y, en base a que la cancelación se debe a una interrupción, indicando la determinación un bloque de diagnóstico de transacción de interrupción del programa en el que se debe realizar el almacenamiento; y

verificar si la cancelación se debe a una condición de interceptación, y en base a que la cancelación se debe a la condición de interceptación, indicando la determinación un bloque de diagnóstico de transacción de interceptación del programa en el que se debe realizar el almacenamiento.

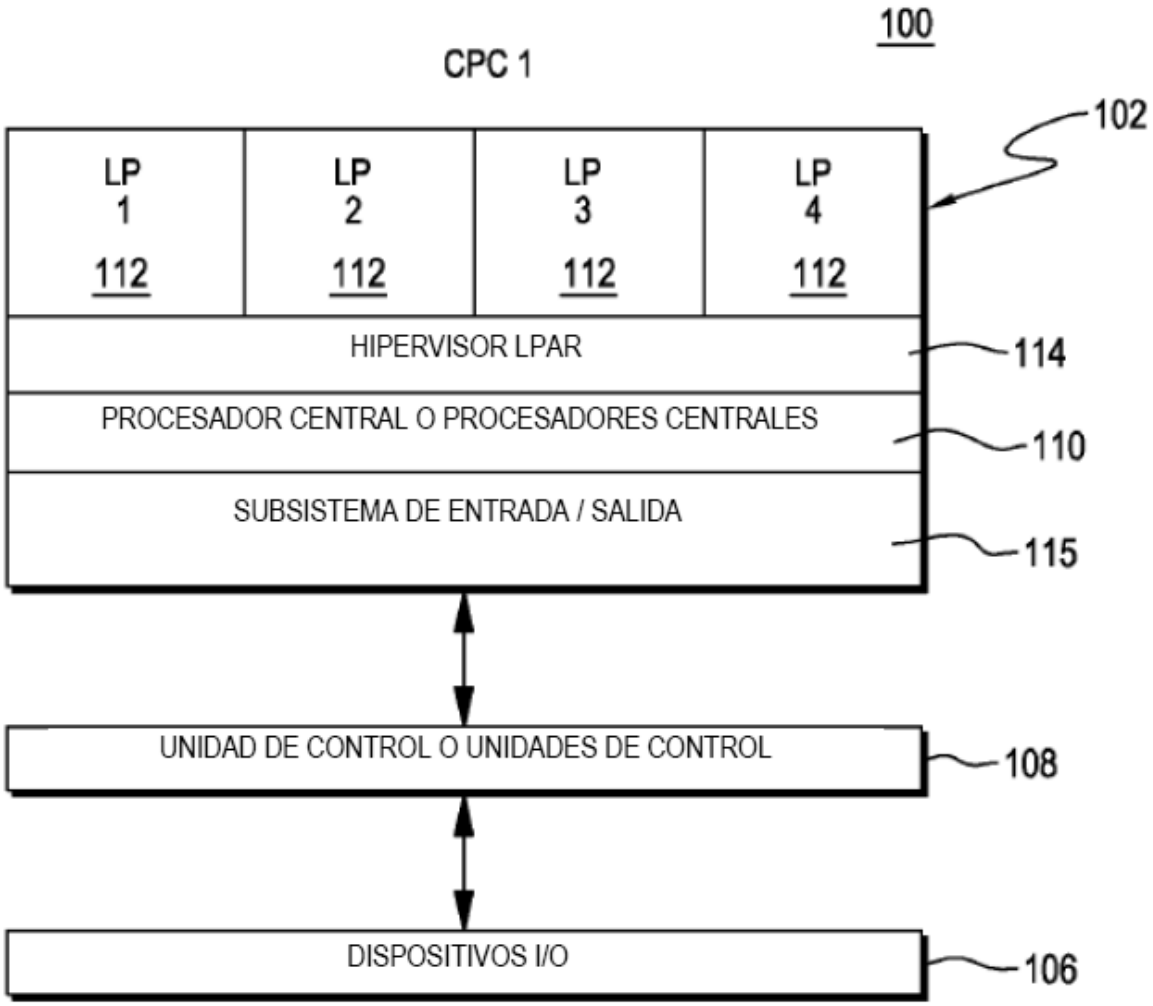
100
CPC 1
102
LP 1
112
LP 2
112
LP 3
112
LP 4
112
HIPERVISOR LPAR
114
PROCESADOR CENTRAL O PROCESADORES CENTRALES
110
SUBSISTEMA DE ENTRADA / SALIDA
115
UNIDAD DE CONTROL O UNIDADES DE CONTROL
108
DISPOSITIVOS I/O
106

FIG. 1

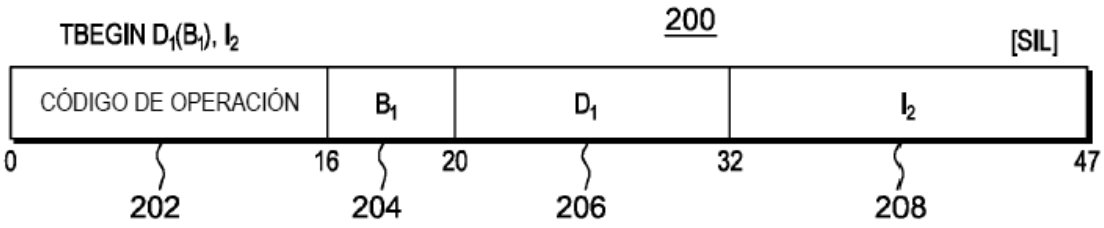

FIG. 2A

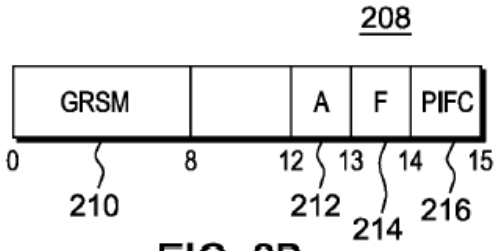

FIG. 2B

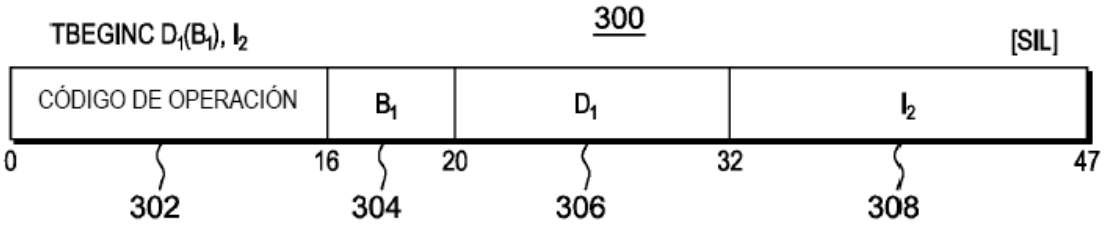

FIG. 3A

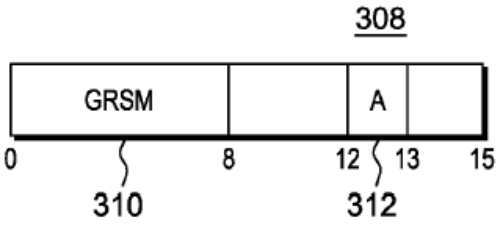

FIG. 3B

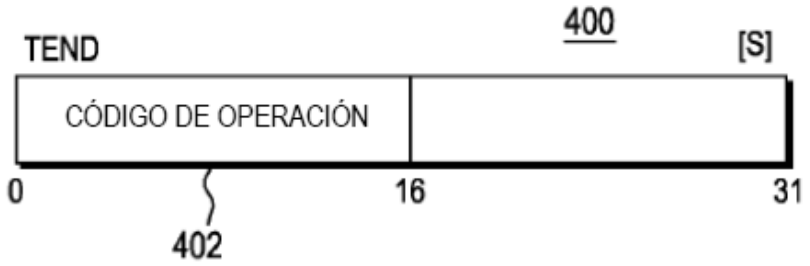
400
TEND
[S]
CÓDIGO DE OPERACIÓN
0
16
31
402

FIG. 4

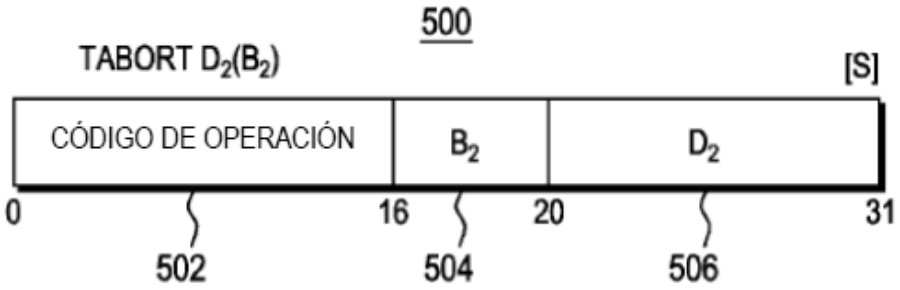
500
TABORT D2(B2)
[S]
CÓDIGO DE OPERACIÓN
B2
D2
0
16
20
31
502
504
506

FIG. 5

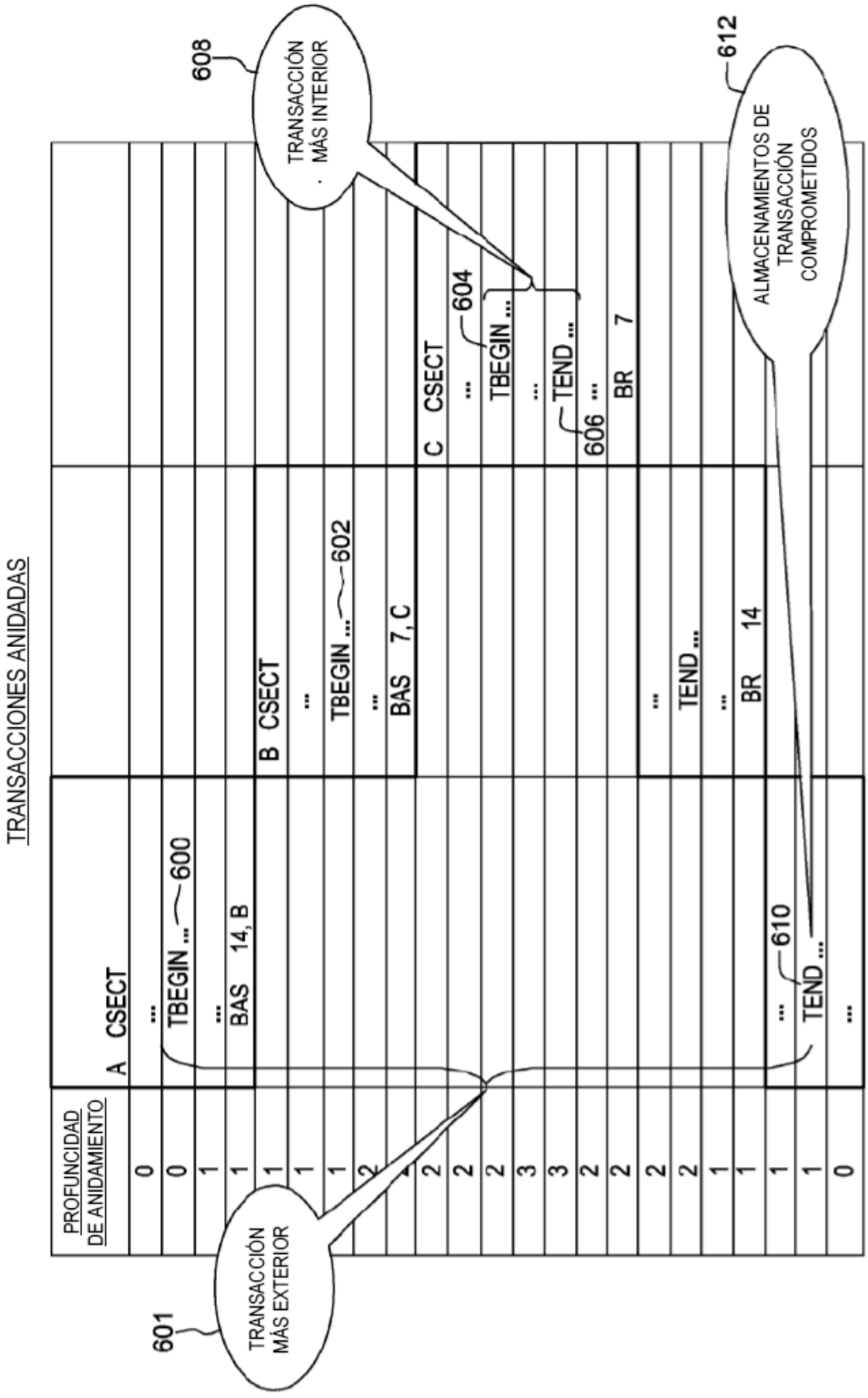
TRANSACCIONES ANIDADAS
PROFUNCIDAD DE ANIDAMIENTO
A CSECT
TBEGIN ... 600
BAS 14, B
B CSECT
TBEGIN ... 602
BAS 7, C
C CSECT
TBEGIN ... 604
TEND ... 606
BR 7
TEND ...
BR 14
TEND ... 610
601
TRANSACCIÓN MÁS EXTERIOR
608
TRANSACCIÓN MÁS INTERIOR
612
ALMACENAMIENTOS DE TRANSACCIÓN COMPROMETIDOS

FIG. 6

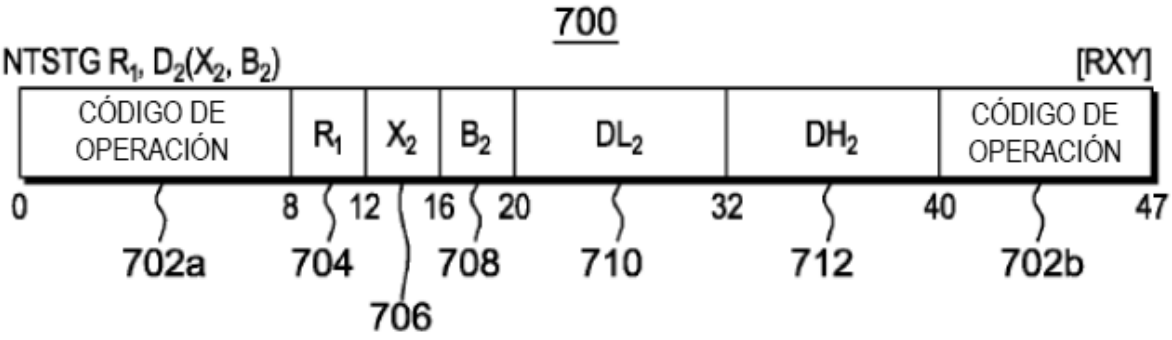
700
NTSTG R_1, $D_2(X_2, B_2)$
[RXY]
CÓDIGO DE OPERACIÓN
R_1
X_2
B_2
DL_2
DH_2
CÓDIGO DE OPERACIÓN
0
8
12
16
20
32
40
47
702a
704
706
708
710
712
702b

FIG. 7

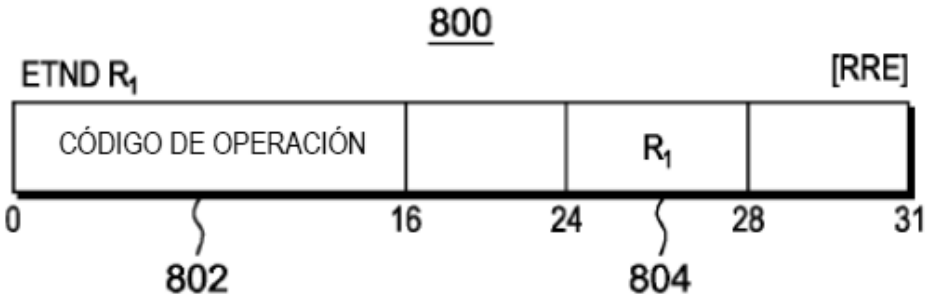
800
ETND R_1
[RRE]
CÓDIGO DE OPERACIÓN
R_1
0
16
24
28
31
802
804

FIG. 8

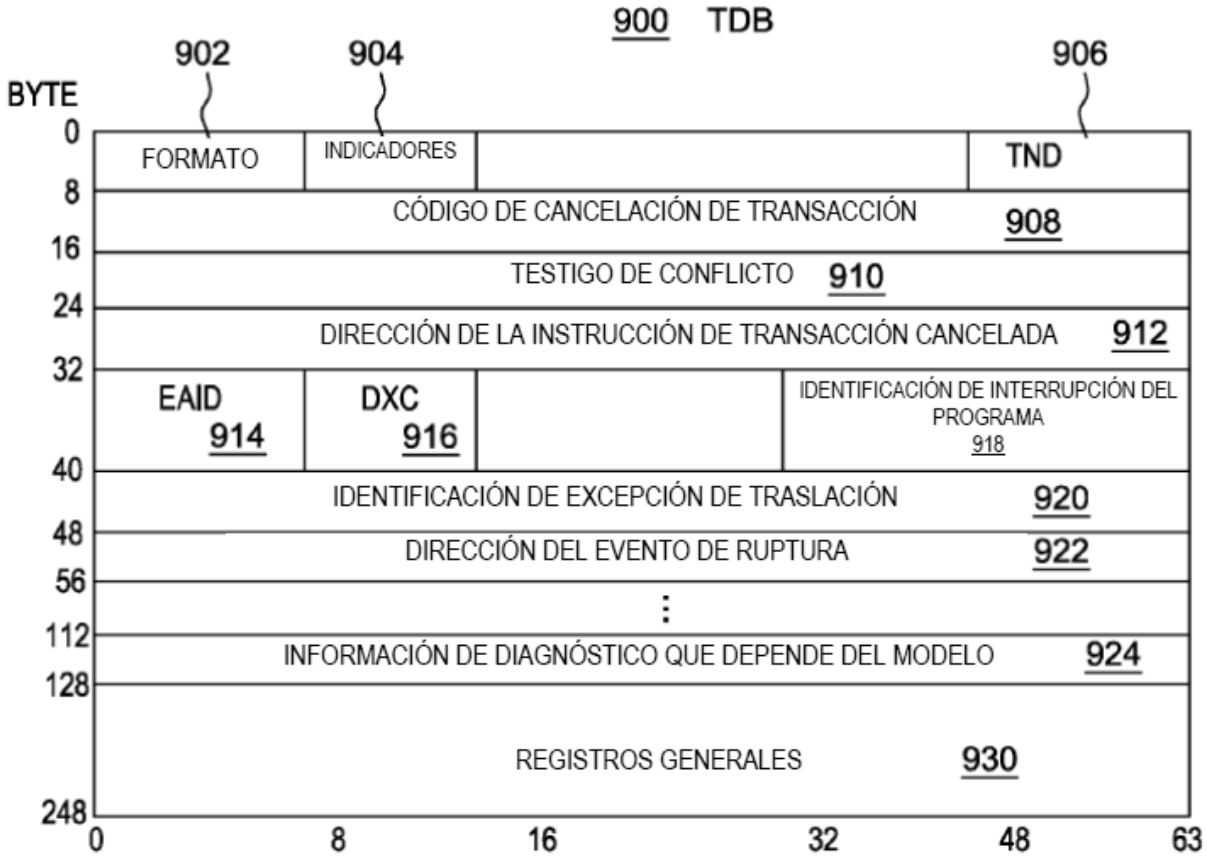
900 TDB
902
904
906
BYTE
0
8
16
24
32
40
48
56
112
128
248
FORMATO
INDICADORES
TND
CÓDIGO DE CANCELACIÓN DE TRANSACCIÓN
908
TESTIGO DE CONFLICTO
910
DIRECCIÓN DE LA INSTRUCCIÓN DE TRANSACCIÓN CANCELADA
912
EAID
914
DXC
916
IDENTIFICACIÓN DE INTERRUPCIÓN DEL PROGRAMA
918
IDENTIFICACIÓN DE EXCEPCIÓN DE TRASLACIÓN
920
DIRECCIÓN DEL EVENTO DE RUPTURA
922
INFORMACIÓN DE DIAGNÓSTICO QUE DEPENDE DEL MODELO
924
REGISTROS GENERALES
930
0
8
16
32
48
63

FIG. 9

CÓDIGO	RAZÓN PARA LA CANCELACIÓN	CC SET
2	INTERRUPCIÓN EXTERNA	2
4	INTERRUPCIÓN DEL PROGRAMA (NO FILTRADO)	2 O 3 +
5	INTERRUPCIÓN DE VERIFICACIÓN MEDIANTE MÁQUINA	2
6	INTERRUPCIÓN DE I/O	2
7	BUSCAR ANULACIÓN	2 O 3
8	ALMACENAR ANULACIÓN	2 O 3
9	BUSCAR CONFLICTO	2
10	ALMACENAR CONFLICTO	2
11	INSTRUCCIÓN RESTRINGIDA	3
12	CONDICIÓN DE EXCEPCIÓN DEL PROGRAMA (FILTRADA)	3
13	PROFUNDIDAD DE ANIDAMIENTO EXCEDIDA	3
14	ALMACENAR EN MEMORIA CACHÉ LA BÚSQUEDA RELATIVA	2 O 3
15	ALMACENAR EN MEMORIA CACHÉ LA BÚSQUEDA RELATIVA	2 O 3
16	ALMACENAR OTRO EN MEMORIA CACHÉ	2 O 3
255	CONDICIÓN MISCELÁNEA	2 O 3
>255	INSTRUCCIÓN TABORT	2 O 3
‡	NO PUEDE SER DETERMINADO; NINGÚN TDB ALMACENADO	1

EXPLICACIÓN:
+ EL CÓDIGO DE LA CONDICIÓN ESTÁ BASADO EN EL CÓDIGO DE INTERRUPCIÓN

‡ ESTA SITUACIÓN OCURRE CUANDO UNA TRANSACCIÓN ES CANCELADA, PERO EL TDB SE HA VUELTO INACCESIBLE DESPUÉS DE LA EJECUCIÓN CON ÉXITO DE LA INSTRUCCIÓN TBEGIN MÁS EXTERIOR. NINGÚN TDB ESPECÍFICO PARA EBEGIN ES ALMACENADO, Y EL CÓDIGO DE CONDICIÓN ES CONFIGURADO A 1.

FIG. 10

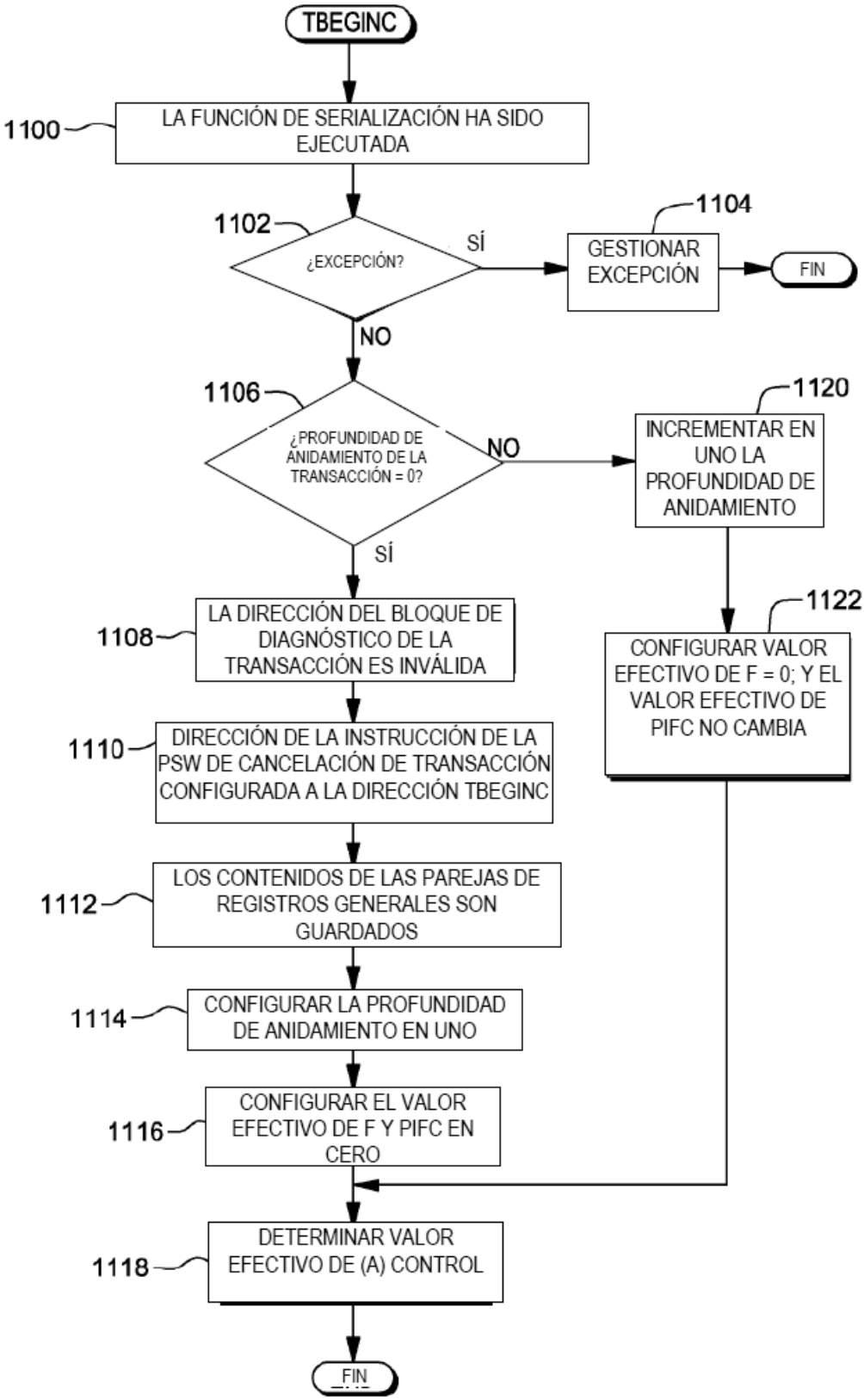
TBEGINC
1100
LA FUNCIÓN DE SERIALIZACIÓN HA SIDO EJECUTADA
1102
¿EXCEPCIÓN?
SÍ
1104
GESTIONAR EXCEPCIÓN
FIN
NO
1106
¿PROFUNDIDAD DE ANIDAMIENTO DE LA TRANSACCIÓN = 0?
NO
1120
INCREMENTAR EN UNO LA PROFUNDIDAD DE ANIDAMIENTO
SÍ
1122
CONFIGURAR VALOR EFECTIVO DE F = 0; Y EL VALOR EFECTIVO DE PIFC NO CAMBIA
1108
LA DIRECCIÓN DEL BLOQUE DE DIAGNÓSTICO DE LA TRANSACCIÓN ES INVÁLIDA
1110
DIRECCIÓN DE LA INSTRUCCIÓN DE LA PSW DE CANCELACIÓN DE TRANSACCIÓN CONFIGURADA A LA DIRECCIÓN TBEGINC
1112
LOS CONTENIDOS DE LAS PAREJAS DE REGISTROS GENERALES SON GUARDADOS
1114
CONFIGURAR LA PROFUNDIDAD DE ANIDAMIENTO EN UNO
1116
CONFIGURAR EL VALOR EFECTIVO DE F Y PIFC EN CERO
1118
DETERMINAR VALOR EFECTIVO DE (A) CONTROL
FIN

FIG. 11

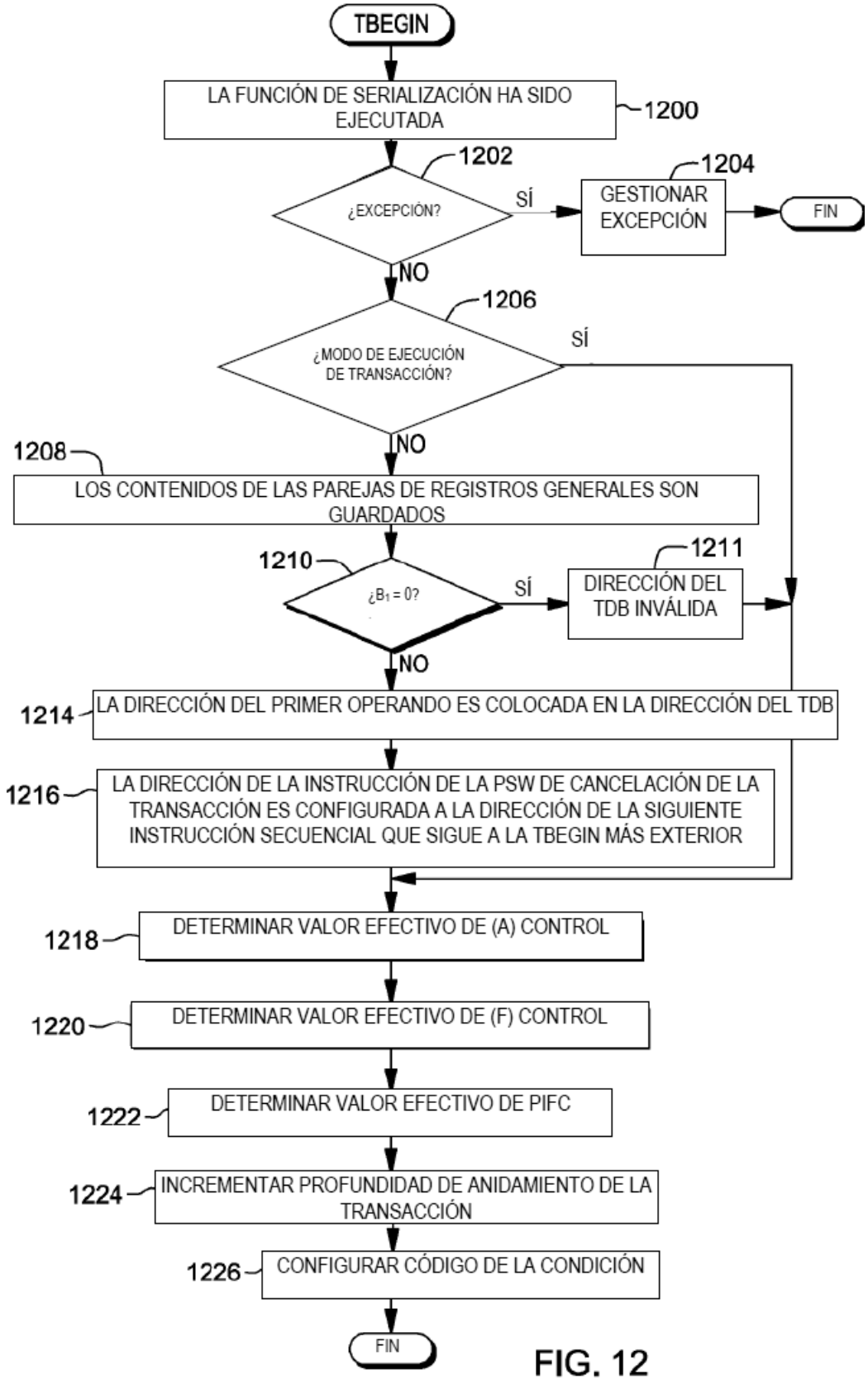
TBEGIN
LA FUNCIÓN DE SERIALIZACIÓN HA SIDO EJECUTADA
1200
1202
¿EXCEPCIÓN?
SÍ
1204
GESTIONAR EXCEPCIÓN
FIN
NO
1206
¿MODO DE EJECUCIÓN DE TRANSACCIÓN?
SÍ
NO
1208
LOS CONTENIDOS DE LAS PAREJAS DE REGISTROS GENERALES SON GUARDADOS
1210
¿B1 = 0?
SÍ
1211
DIRECCIÓN DEL TDB INVÁLIDA
NO
1214
LA DIRECCIÓN DEL PRIMER OPERANDO ES COLOCADA EN LA DIRECCIÓN DEL TDB
1216
LA DIRECCIÓN DE LA INSTRUCCIÓN DE LA PSW DE CANCELACIÓN DE LA TRANSACCIÓN ES CONFIGURADA A LA DIRECCIÓN DE LA SIGUIENTE INSTRUCCIÓN SECUENCIAL QUE SIGUE A LA TBEGIN MÁS EXTERIOR
1218
DETERMINAR VALOR EFECTIVO DE (A) CONTROL
1220
DETERMINAR VALOR EFECTIVO DE (F) CONTROL
1222
DETERMINAR VALOR EFECTIVO DE PIFC
1224
INCREMENTAR PROFUNDIDAD DE ANIDAMIENTO DE LA TRANSACCIÓN
1226
CONFIGURAR CÓDIGO DE LA CONDICIÓN
FIN

FIG. 12

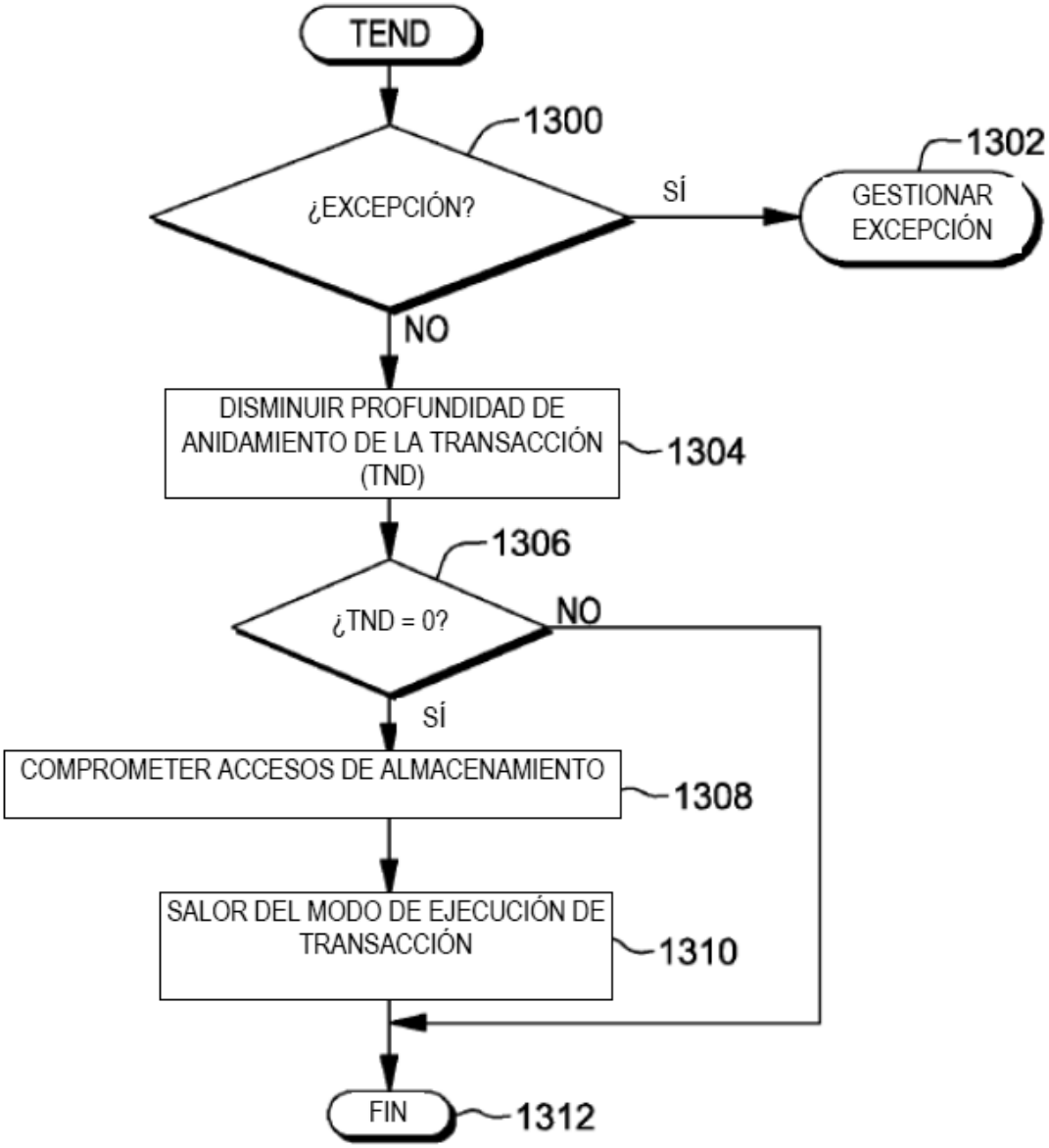
TEND
1300
¿EXCEPCIÓN?
SÍ
1302
GESTIONAR EXCEPCIÓN
NO
DISMINUIR PROFUNDIDAD DE ANIDAMIENTO DE LA TRANSACCIÓN (TND)
1304
1306
¿TND = 0?
NO
SÍ
COMPROMETER ACCESOS DE ALMACENAMIENTO
1308
SALOR DEL MODO DE EJECUCIÓN DE TRANSACCIÓN
1310
FIN
1312

FIG. 13

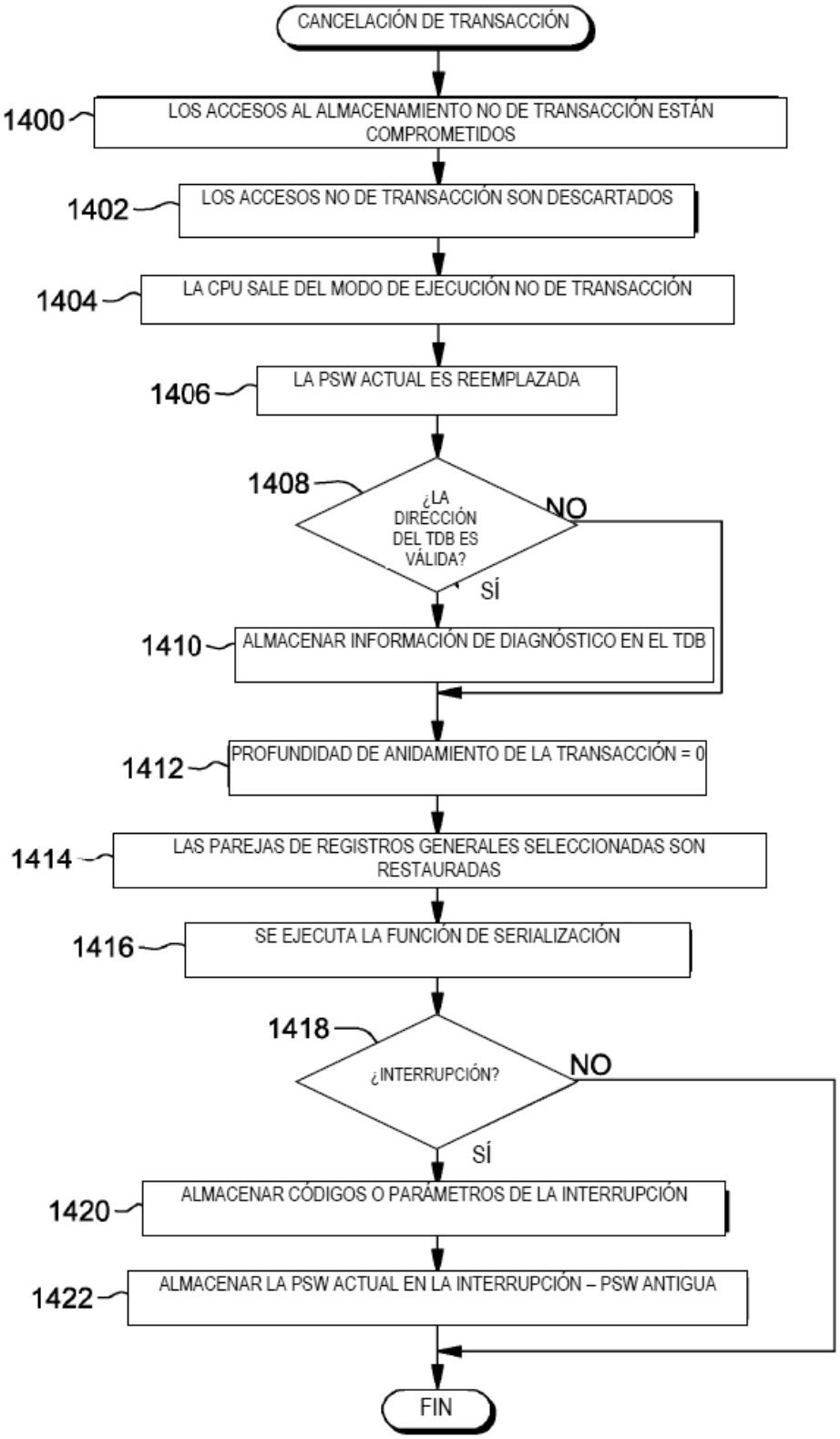
CANCELACIÓN DE TRANSACCIÓN
1400
LOS ACCESOS AL ALMACENAMIENTO NO DE TRANSACCIÓN ESTÁN COMPROMETIDOS
1402
LOS ACCESOS NO DE TRANSACCIÓN SON DESCARTADOS
1404
LA CPU SALE DEL MODO DE EJECUCIÓN NO DE TRANSACCIÓN
1406
LA PSW ACTUAL ES REEMPLAZADA
1408
¿LA DIRECCIÓN DEL TDB ES VÁLIDA?
NO
SÍ
1410
ALMACENAR INFORMACIÓN DE DIAGNÓSTICO EN EL TDB
1412
PROFUNDIDAD DE ANIDAMIENTO DE LA TRANSACCIÓN = 0
1414
LAS PAREJAS DE REGISTROS GENERALES SELECCIONADAS SON RESTAURADAS
1416
SE EJECUTA LA FUNCIÓN DE SERIALIZACIÓN
1418
¿INTERRUPCIÓN?
NO
SÍ
1420
ALMACENAR CÓDIGOS O PARÁMETROS DE LA INTERRUPCIÓN
1422
ALMACENAR LA PSW ACTUAL EN LA INTERRUPCIÓN – PSW ANTIGUA
FIN

FIG. 14

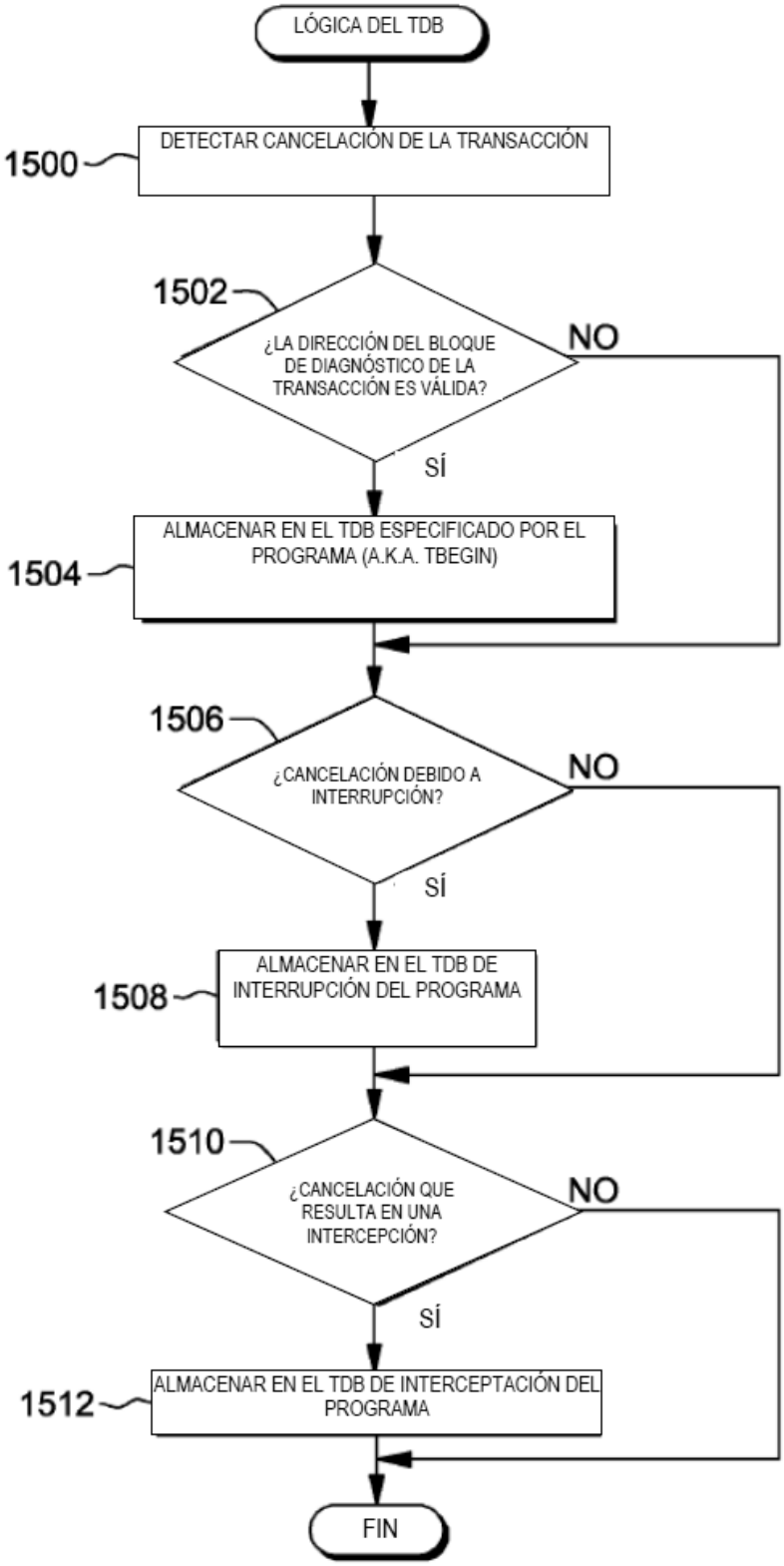
LÓGICA DEL TDB
1500
DETECTAR CANCELACIÓN DE LA TRANSACCIÓN
1502
¿LA DIRECCIÓN DEL BLOQUE DE DIAGNÓSTICO DE LA TRANSACCIÓN ES VÁLIDA?
NO
SÍ
1504
ALMACENAR EN EL TDB ESPECIFICADO POR EL PROGRAMA (A.K.A. TBEGIN)
1506
¿CANCELACIÓN DEBIDO A INTERRUPCIÓN?
NO
SÍ
1508
ALMACENAR EN EL TDB DE INTERRUPCIÓN DEL PROGRAMA
1510
¿CANCELACIÓN QUE RESULTA EN UNA INTERCEPCIÓN?
NO
SÍ
1512
ALMACENAR EN EL TDB DE INTERCEPTACIÓN DEL PROGRAMA
FIN

FIG. 15

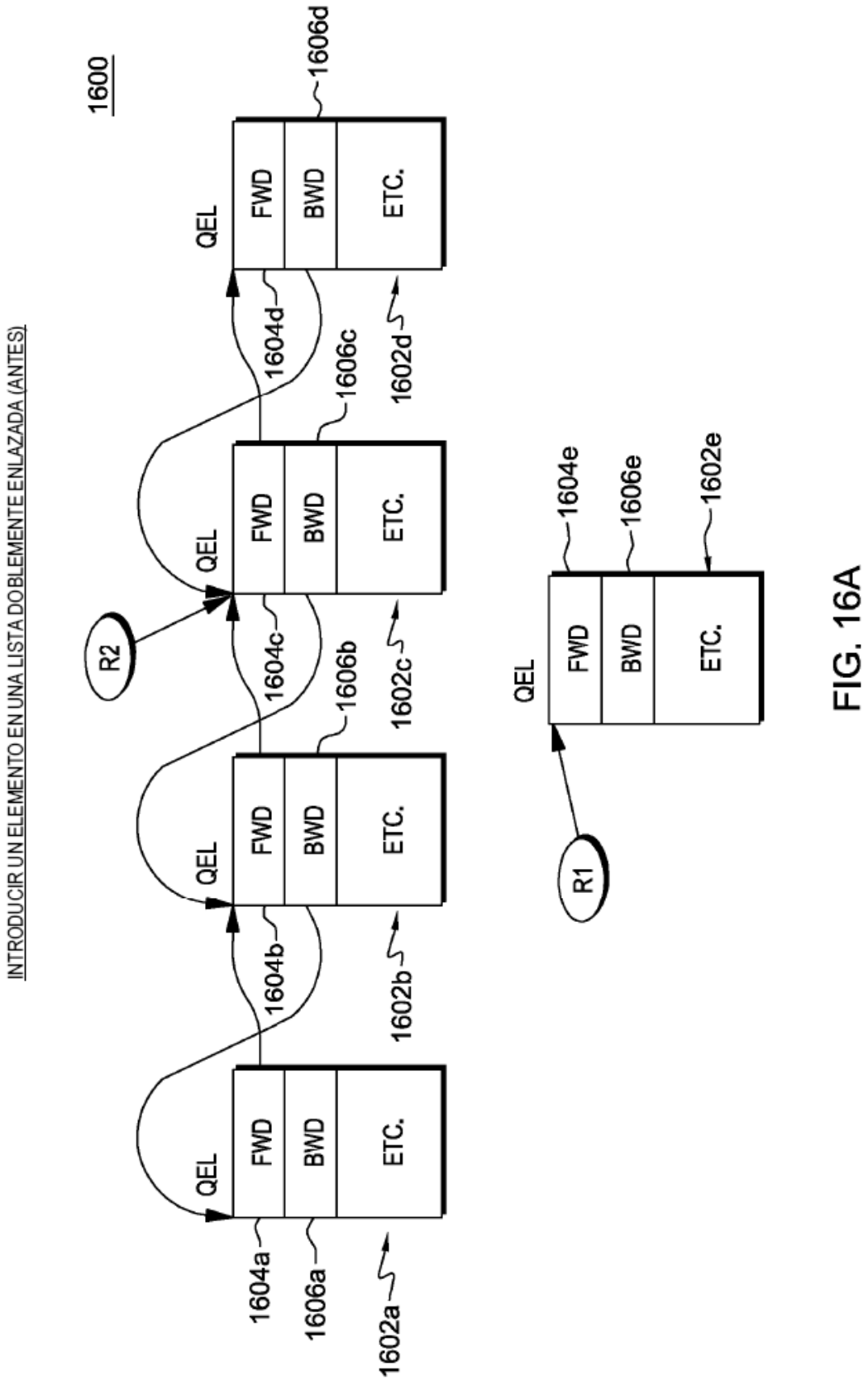
INTRODUCIR UN ELEMENTO EN UNA LISTA DOBLEMENTE ENLAZADA (ANTES)
1600
QEL
FWD
BWD
ETC.
1604a
1606a
1602a
1604b
1606b
1602b
1604c
1606c
1602c
1604d
1606d
1602d
R2
1604e
1606e
1602e
R1

FIG. 16A

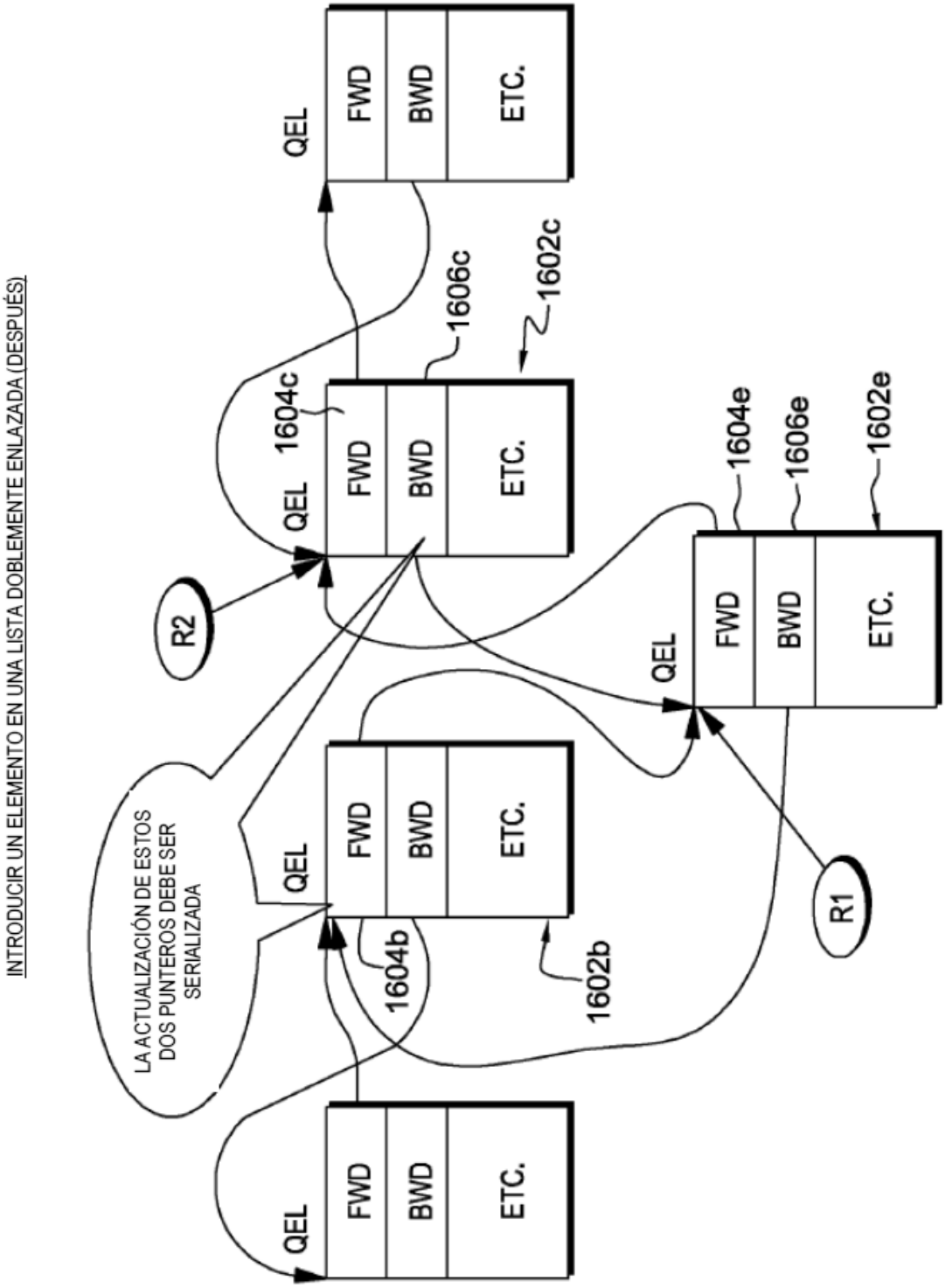
INTRODUCIR UN ELEMENTO EN UNA LISTA DOBLEMENTE ENLAZADA (DESPUÉS)
QEL
FWD
BWD
ETC.
1604c
1606c
1602c
R2
1604e
1606e
1602e
LA ACTUALIZACIÓN DE ESTOS DOS PUNTEROS DEBE SER SERIALIZADA
1604b
1602b
R1

FIG. 16B

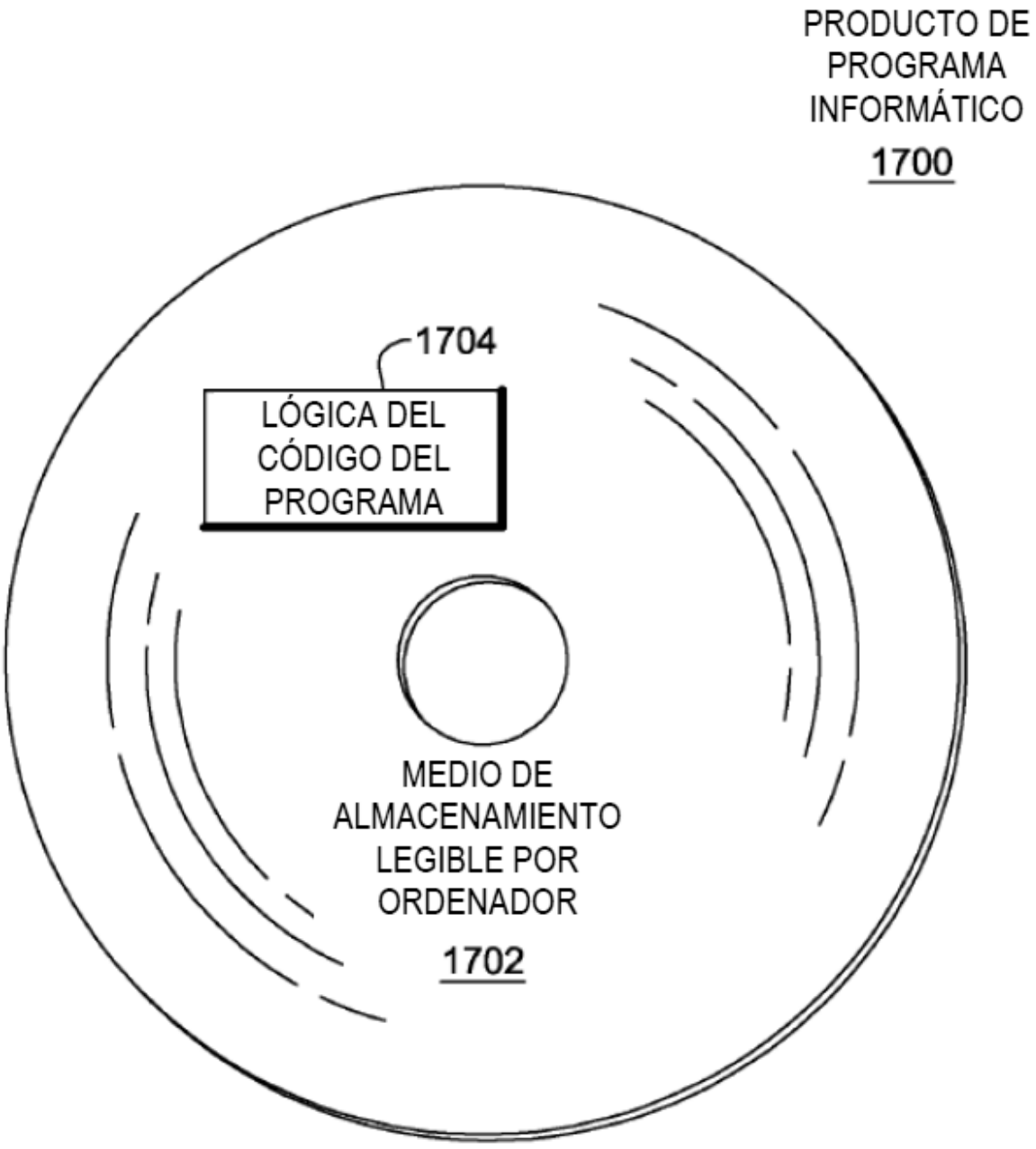
PRODUCTO DE PROGRAMA INFORMÁTICO
1700
1704
LÓGICA DEL CÓDIGO DEL PROGRAMA
MEDIO DE ALMACENAMIENTO LEGIBLE POR ORDENADOR
1702

FIG. 17

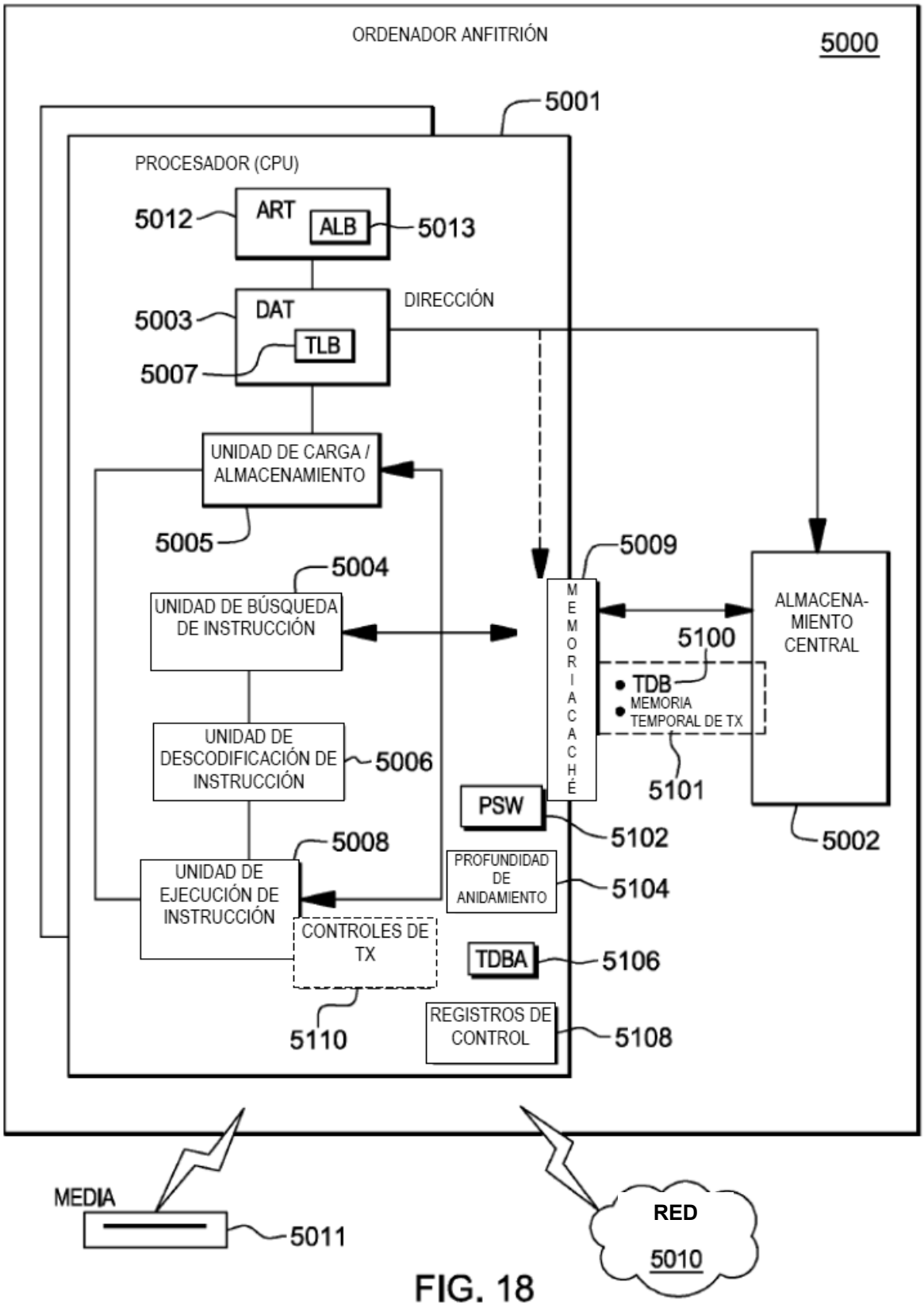
ORDENADOR ANFITRIÓN
5000
5001
PROCESADOR (CPU)
5012
ART
ALB
5013
5003
DAT
DIRECCIÓN
TLB
5007
UNIDAD DE CARGA / ALMACENAMIENTO
5005
5004
UNIDAD DE BÚSQUEDA DE INSTRUCCIÓN
UNIDAD DE DESCODIFICACIÓN DE INSTRUCCIÓN
5006
5008
UNIDAD DE EJECUCIÓN DE INSTRUCCIÓN
CONTROLES DE TX
5110
5009
MEMORIA CACHÉ
5100
TDB
MEMORIA TEMPORAL DE TX
5101
ALMACENA-MIENTO CENTRAL
5002
PSW
5102
PROFUNDIDAD DE ANIDAMIENTO
5104
TDBA
5106
REGISTROS DE CONTROL
5108
MEDIA
5011
RED
5010

FIG. 18

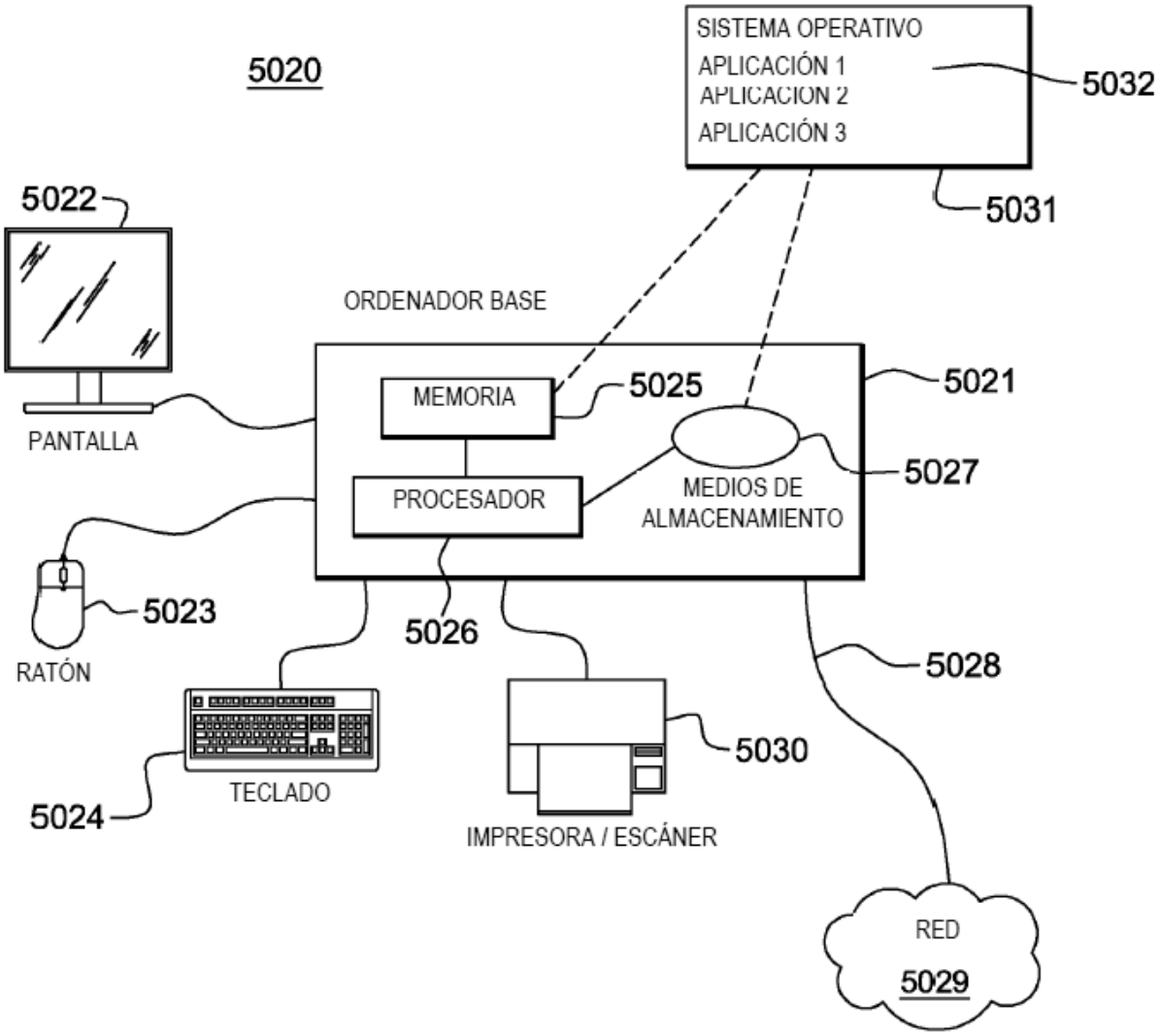
5020
SISTEMA OPERATIVO
APLICACIÓN 1
APLICACION 2
APLICACIÓN 3
5032
5031
5022
ORDENADOR BASE
MEMORIA
5025
5021
PANTALLA
PROCESADOR
MEDIOS DE ALMACENAMIENTO
5027
5023
RATÓN
5026
5028
5030
TECLADO
5024
IMPRESORA / ESCÁNER
RED
5029

FIG. 19

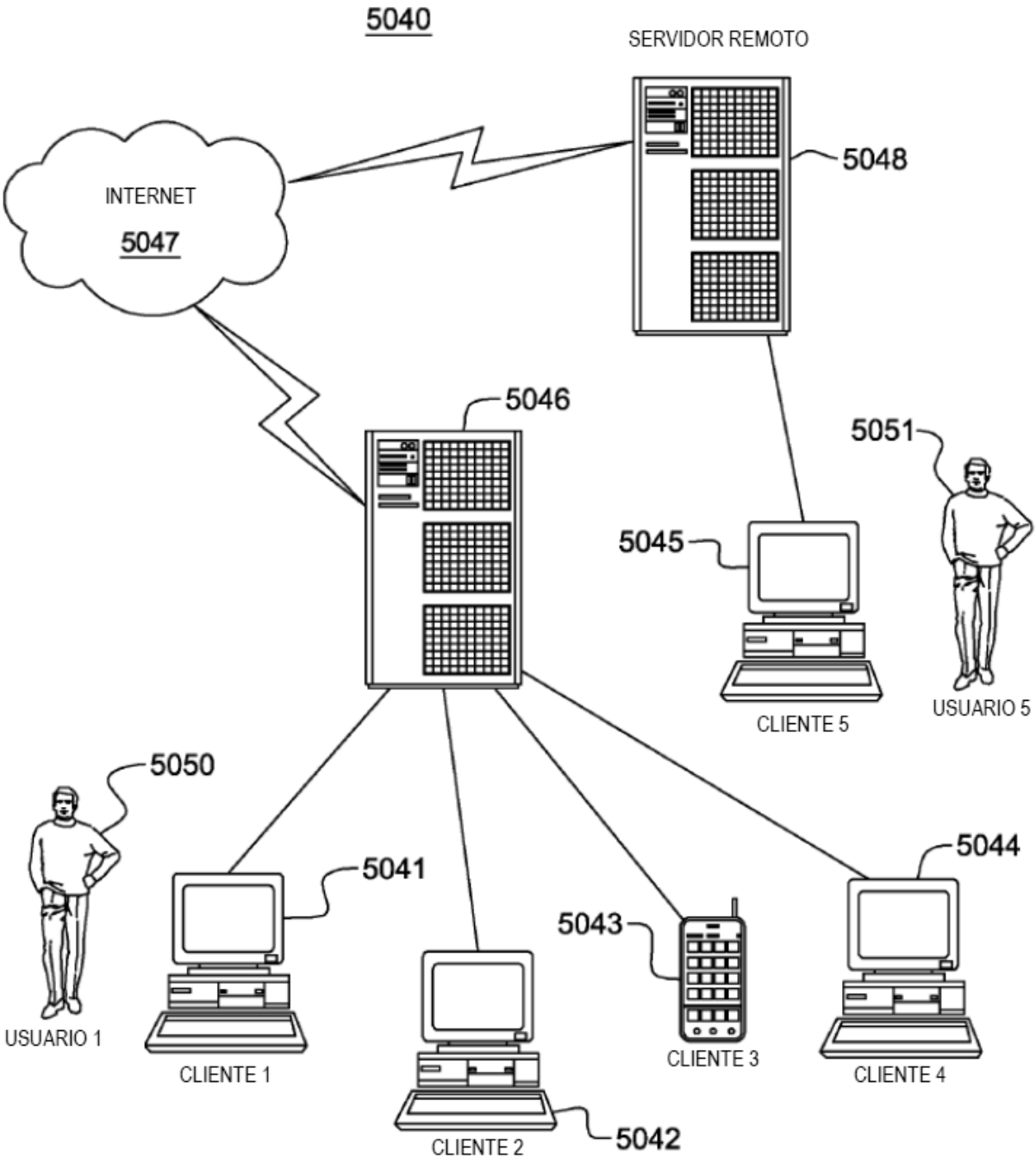
5040
SERVIDOR REMOTO
5048
INTERNET
5047
5046
5051
5045
CLIENTE 5
USUARIO 5
5050
5041
5044
5043
USUARIO 1
CLIENTE 1
CLIENTE 3
CLIENTE 4
CLIENTE 2
5042

FIG. 20

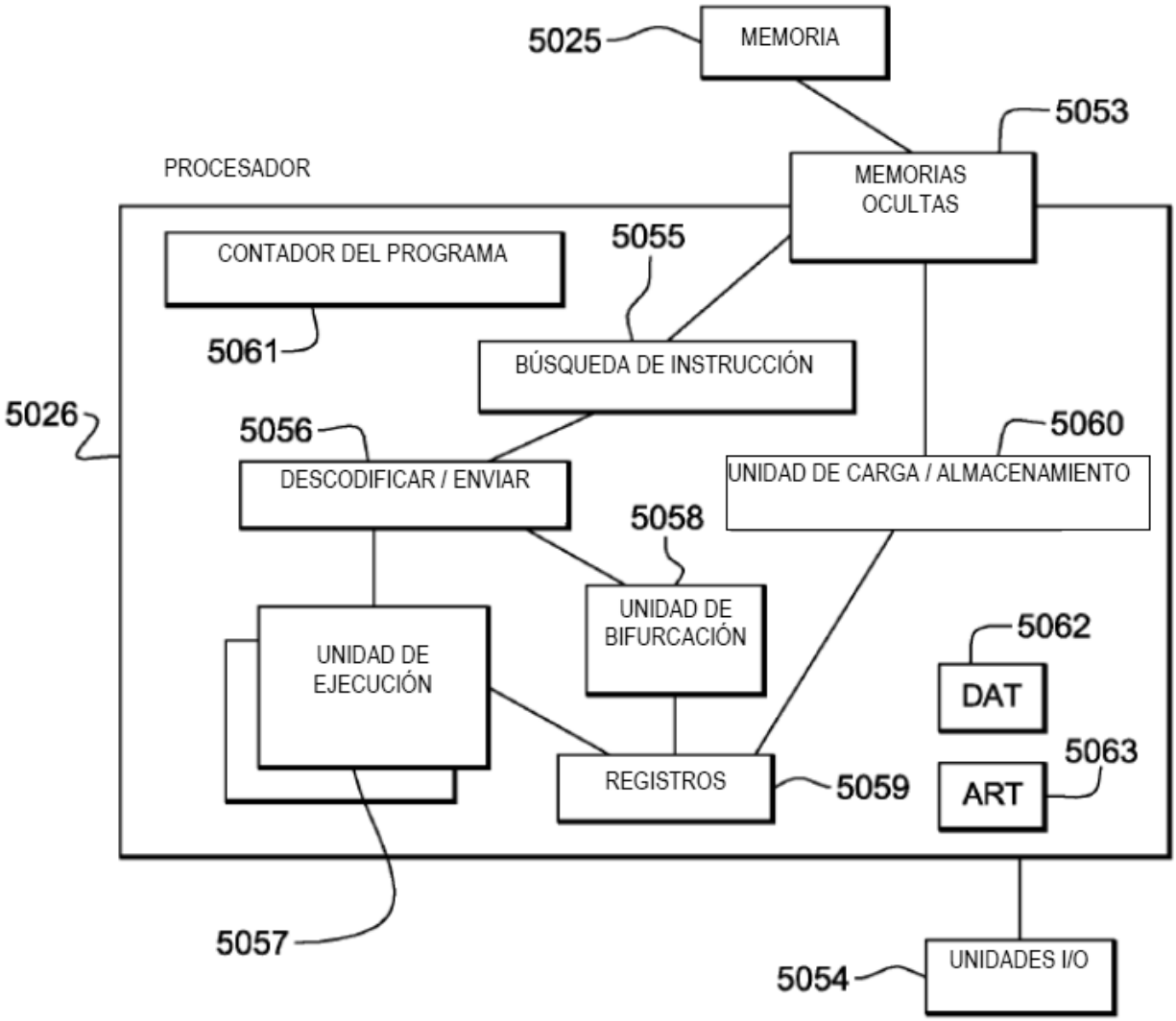
5025
MEMORIA
5053
PROCESADOR
MEMORIAS OCULTAS
CONTADOR DEL PROGRAMA
5055
5061
BÚSQUEDA DE INSTRUCCIÓN
5026
5056
5060
DESCODIFICAR / ENVIAR
UNIDAD DE CARGA / ALMACENAMIENTO
5058
UNIDAD DE BIFURCACIÓN
5062
UNIDAD DE EJECUCIÓN
DAT
5063
REGISTROS
5059
ART
5057
5054
UNIDADES I/O

FIG. 21

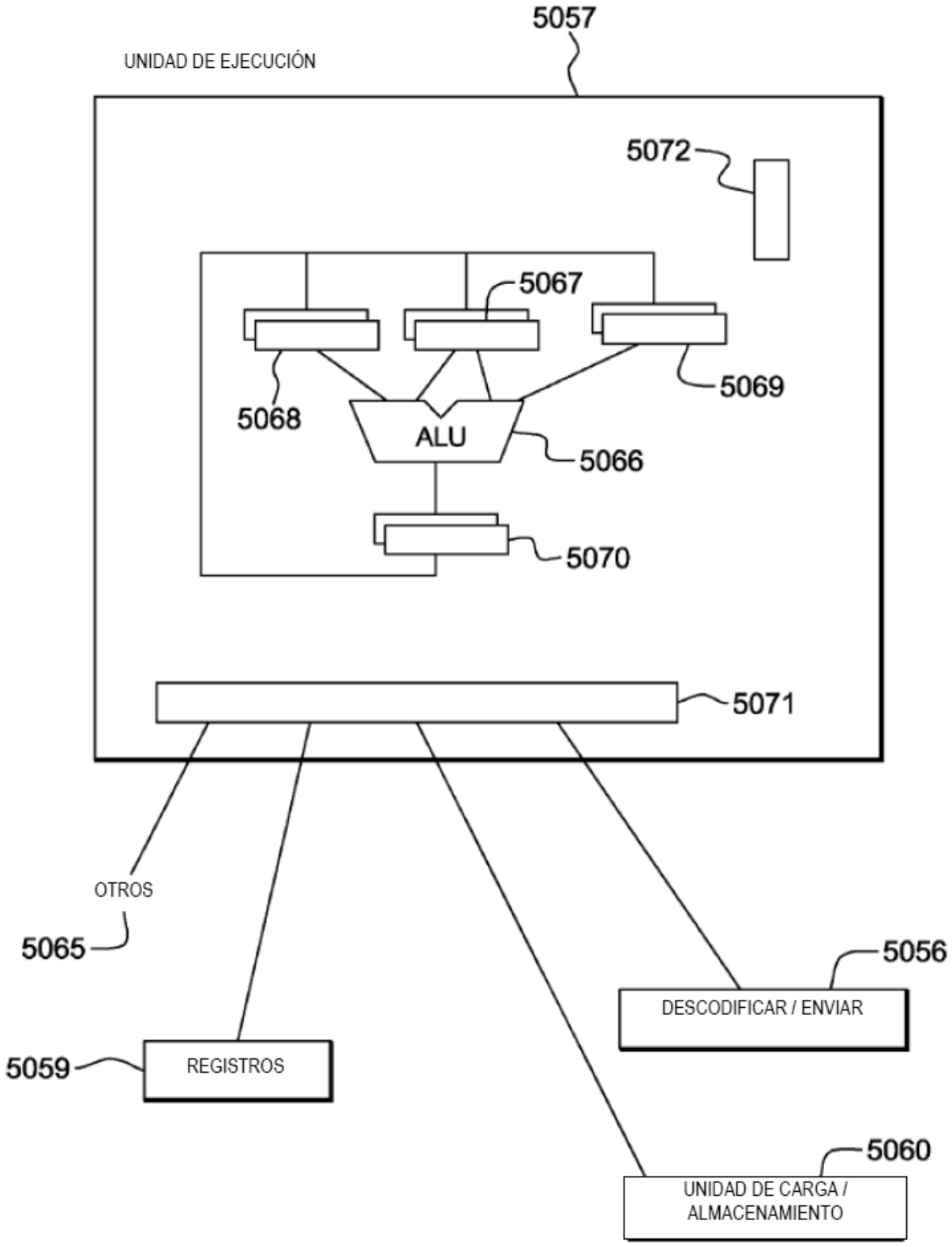
5057
UNIDAD DE EJECUCIÓN
5072
5067
5068
5069
ALU
5066
5070
5071
OTROS
5065
5056
DESCODIFICAR / ENVIAR
5059
REGISTROS
5060
UNIDAD DE CARGA / ALMACENAMIENTO

FIG. 22A

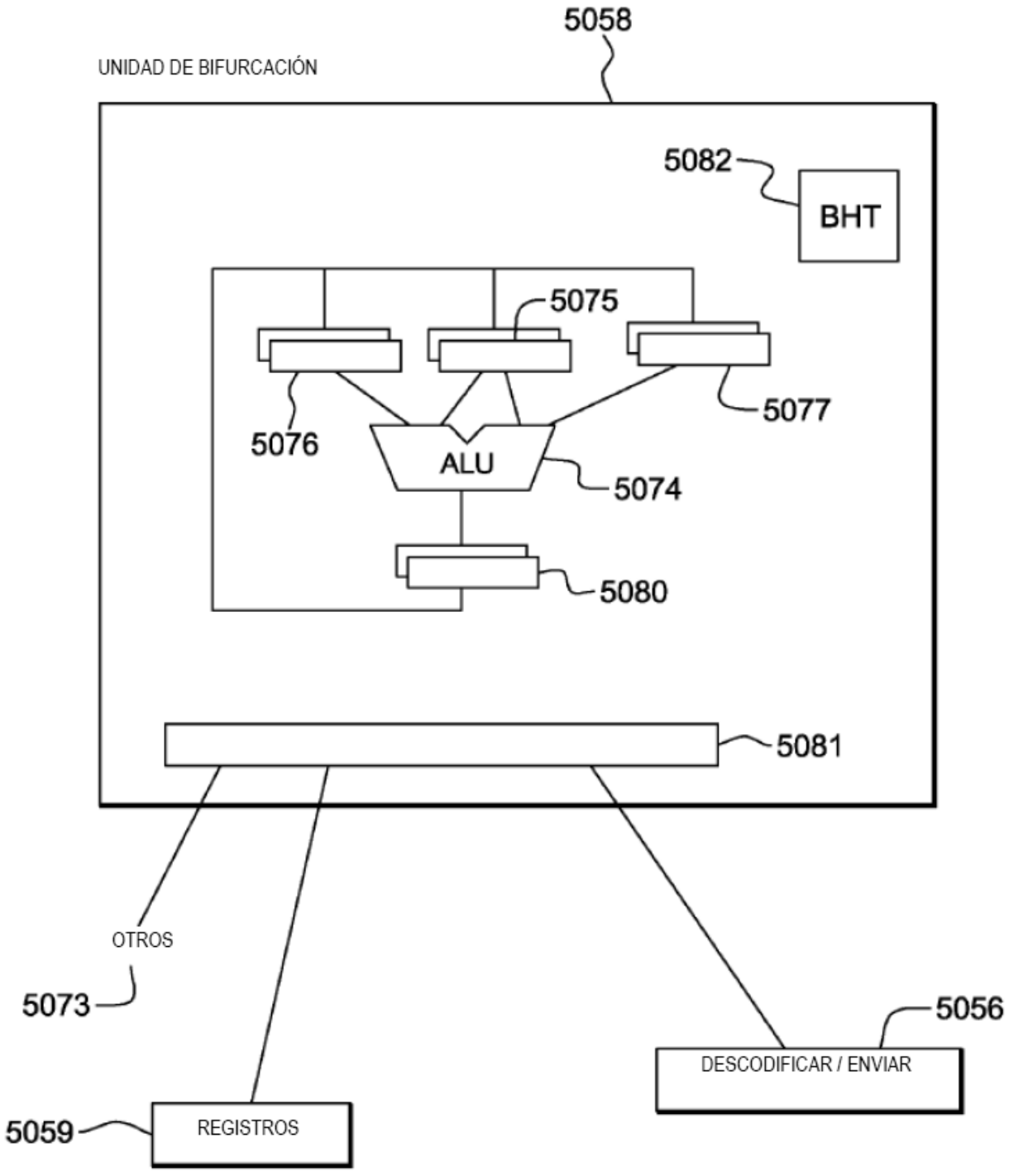
5058
UNIDAD DE BIFURCACIÓN
5082
BHT
5075
5076
5077
ALU
5074
5080
5081
OTROS
5073
5056
DESCODIFICAR / ENVIAR
5059
REGISTROS

FIG. 22B

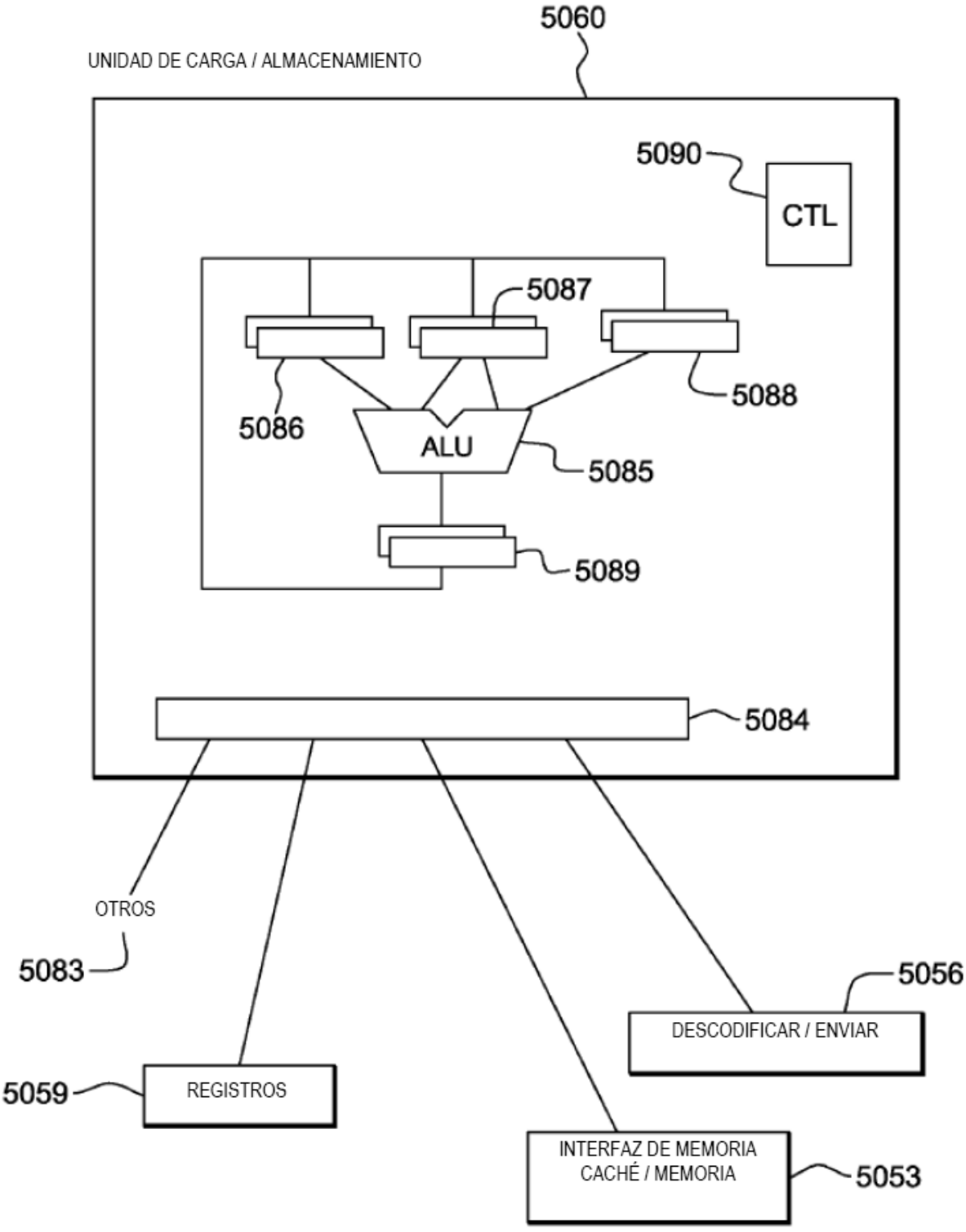
5060
UNIDAD DE CARGA / ALMACENAMIENTO
5090
CTL
5087
5086
5088
ALU
5085
5089
5084
OTROS
5083
5056
DESCODIFICAR / ENVIAR
5059
REGISTROS
INTERFAZ DE MEMORIA
CACHÉ / MEMORIA
5053

FIG. 22C

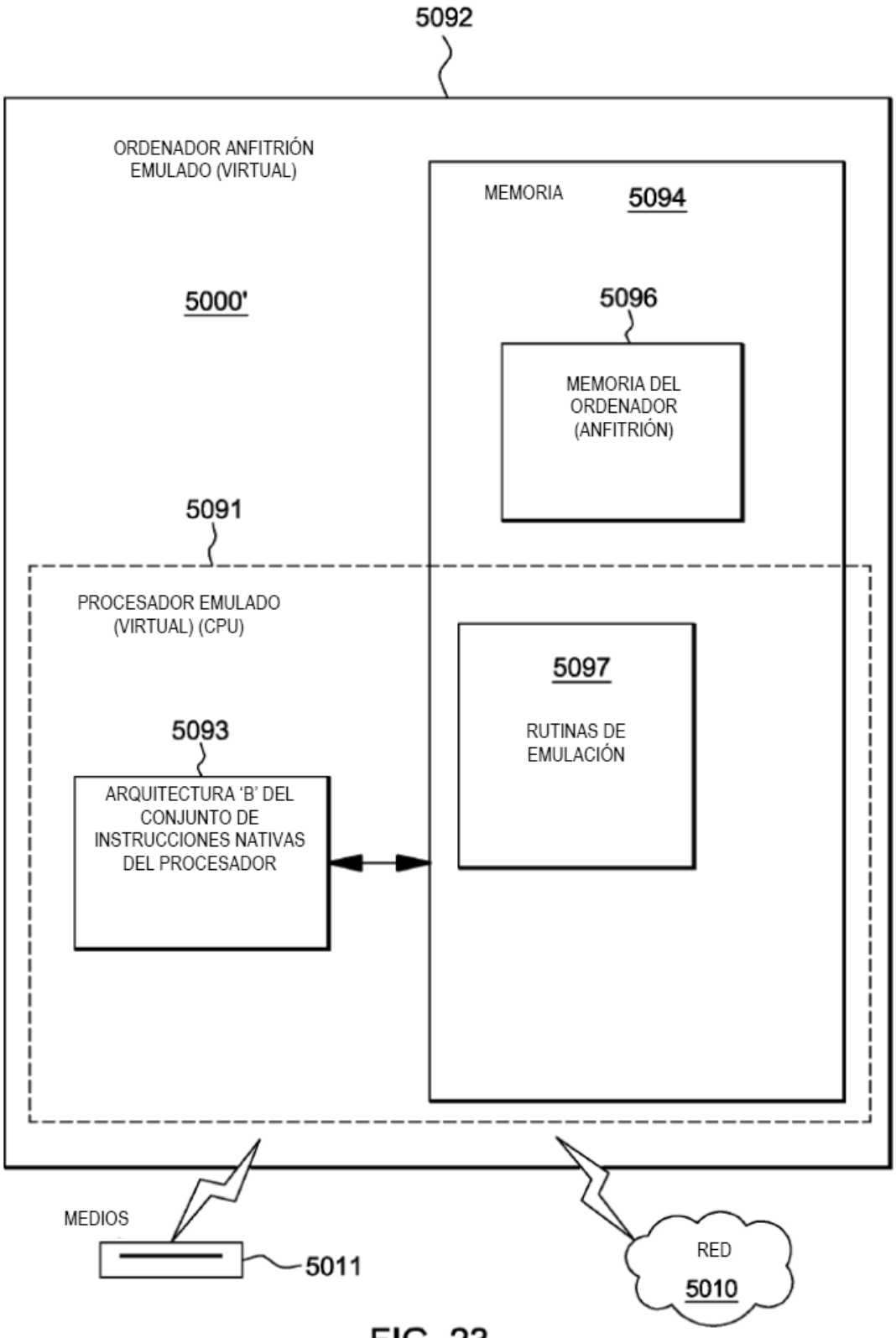
5092
ORDENADOR ANFITRIÓN EMULADO (VIRTUAL)
MEMORIA
5094
5000'
5096
MEMORIA DEL ORDENADOR (ANFITRIÓN)
5091
PROCESADOR EMULADO (VIRTUAL) (CPU)
5097
RUTINAS DE EMULACIÓN
5093
ARQUITECTURA 'B' DEL CONJUNTO DE INSTRUCCIONES NATIVAS DEL PROCESADOR
MEDIOS
5011
RED
5010

FIG. 23