(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 368 682**

(51) Int. CI.:
G06F 9/445 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Número de solicitud europea: **09700229 .9**
(96) Fecha de presentación: **07.01.2009**
(97) Número de publicación de la solicitud: **2229620**
(97) Fecha de publicación de la solicitud: **22.09.2010**

(54) Título: **MECANISMO DE EXTRACCIÓN DE ATRIBUTOS DE CACHÉ E INSTRUCCIÓN PARA EL MISMO.**

(30) Prioridad:
11.01.2008 US 972675

(45) Fecha de publicación de la mención BOPI:
21.11.2011

(45) Fecha de la publicación del folleto de la patente:
21.11.2011

(73) Titular/es:
INTERNATIONAL BUSINESS MACHINES CORPORATION
NEW ORCHARD ROAD
ARMONK, NEW YORK 10504, US

(72) Inventor/es:
GREINER, Dan y
SLEGEL, Timothy

(74) Agente: **de Elzaburu Márquez, Alberto**

DESCRIPCIÓN

Mecanismo de extracción de atributos de caché e instrucción para el mismo.

Ámbito de la invención

La presente invención se refiere a sistemas informáticos y más particularmente a la funcionalidad de las instrucciones del procesador del sistema informático.

Antecedentes de la invención

Marcas: IBM® es una marca comercial registrada de International Business Machines Corporation, Armonk, Nueva York, EE.UU. S/390, Z900 y z990 y otros nombres de productos pueden ser marcas comerciales registradas o nombres de productos de International Business Machines Corporation o de otras empresas.

A partir de las máquinas conocidas como IBM® System 360 en la década de 1960 hasta la actualidad, IBM ha desarrollado una arquitectura que se conoce como "*the mainframe*", cuyos principios de funcionamiento establecen la arquitectura de la máquina mediante la descripción de las instrucciones que pueden ejecutarse con la implementación de "*mainframe*" de las instrucciones que habían sido inventadas por los inventores de IBM y adoptadas, debido a su significativa contribución a la mejora del estado de la máquina de computación representada por "*the mainframe*", como contribuciones significativas por la inclusión en los Principios de Funcionamiento de IBM tal como se estableció con los años. La sexta edición de los Principios de Funcionamiento de la z/Architecture® de IBM® que fue publicada en abril de 2007 se ha convertido en la referencia publicada estándar como SA22-7832-05 y se incorpora en los servidores *mainframe* z9® de IBM.

Haciendo referencia a la figura 1A, se retratan los componentes representativos de un sistema informático anfitrión 50 de la técnica anterior. También se pueden emplear otras disposiciones de componentes en un sistema informático, que son bien conocidos en la técnica. El Ordenador Anfitrión representativo 50 comprende una o varias CPU 1 en comunicación con el almacén principal (memoria 2 de ordenador), así como unas interfaces de E/S con dispositivos de almacenamiento 11 y redes 10 para comunicarse con otros ordenadores o SAN y similares. La CPU 1 es compatible con una arquitectura que tiene un conjunto de instrucciones con arquitectura y funcionalidad con arquitectura. La CPU 1 puede tener traducción dinámica de direcciones (DAT) 3 para la transformación de las direcciones de programa (direcciones virtuales) en la dirección real de la memoria. Una DAT normalmente incluye un búfer de traducción anticipada de direcciones (TLB: *Translation Lookaside Buffer*) 7 para almacenar en caché las traducciones de manera que un acceso posterior al bloque de memoria 2 del ordenador no requiera el retraso de la traducción de direcciones. Normalmente, se emplea una caché 9 entre la Memoria 2 del Ordenador y el Procesador 1. La caché 9 puede ser jerárquica que tiene una gran caché disponible para más de una CPU y cachés más pequeñas y más rápidas, (nivel inferior) entre la cache caché grande y cada CPU. En algunas implementaciones las cachés de nivel inferior se dividen para proporcionar cachés separadas de bajo nivel para la extracción de instrucciones y acceso a los datos. En una realización, una instrucción se extrae de la memoria 2 mediante una unidad 4 de extracción de instrucciones a través de una caché 9. La instrucción se decodifica en una unidad (6) de decodificación de instrucciones y se despacha (con otras instrucciones en algunas realizaciones) a las unidades 8 de ejecución de instrucciones. Por lo general se emplean varias unidades de ejecución 8, por ejemplo, una unidad de ejecución aritmética, una unidad de ejecución de coma flotante y una unidad de ejecución de instrucciones de salto. La instrucción es ejecutada por la unidad de ejecución, que accede a operandos de la memoria o los registros específicos de instrucciones según sea necesario, si se va a acceder a un operando (cargado o almacenado) desde la memoria 2, una unidad 5 de almacén de carga normalmente maneja el acceso bajo el control de la instrucción que se está ejecutando. Las instrucciones pueden ser ejecutadas en circuitos de hardware o en micro-código interno (*firmware*) o por una combinación de ambos.

El documento US 6088789 describe un microprocesador, que se configura para ejecutar una instrucción de extracción que especifica una línea de caché que va a ser transferida al microprocesador, así como un modo de acceso para la línea de caché.

En la figura 1B, se proporciona un ejemplo de un sistema informático anfitrión emulado 21 de la técnica anterior que emula un sistema informático anfitrión 50 de una arquitectura de anfitrión. En el sistema informático Anfitrión emulado 21, el Procesador anfitrión (CPU) 1 es un procesador Anfitrión emulado (o procesador Anfitrión virtual) y comprende un procesador de emulación 27 que tiene una arquitectura nativa de conjunto de instrucciones diferente de la del procesador 1 del Ordenador Anfitrión 50. El sistema informático Anfitrión emulado 21 tiene una memoria 22 accesible para el procesador de emulación 27. En un ejemplo de realización, la Memoria 27 se divide en una parte de Memoria 2 de Ordenador Anfitrión y una parte de Rutinas de Emulación 23. La Memoria 2 de Ordenador Anfitrión está disponible para los programas del Ordenador Anfitrión 21 según la Arquitectura de Ordenador Anfitrión. El procesador de emulación 27 ejecuta instrucciones nativas de un conjunto de instrucciones con arquitectura de una arquitectura que no sea la del procesador emulado 1, las instrucciones nativas obtenidas de la memoria 23 de Rutinas de Emulación, y puede acceder a una instrucción de Anfitrión para la ejecución desde un programa en la Memoria 2 de Ordenador Anfitrión mediante el empleo de una o varias instrucciones obtenidas en una rutina de Secuencia y Acceso/Decodificación que puede decodificar la mayoría de las instrucciones que se han accedido para

determinar una rutina de ejecución de instrucciones nativas para emular la función de la Instrucción de anfitrión a la que se ha accedido. Otros mecanismos que se definen para la arquitectura del Sistema 50 de Ordenador Anfitrión pueden ser emulados por las Rutinas de Instalaciones con Arquitectura, incluidos los mecanismos tales como Registros de Propósito General, Registros de Control, Traducción Dinámica de Direcciones y caché de procesador y apoyo de Subsistema de E/S, por ejemplo. Las Rutinas de Emulación también pueden tomar ventaja de las funciones disponibles en el Procesador de emulación 27 (como los registros generales y la traducción dinámica de direcciones virtuales) para mejorar el rendimiento de las Rutinas de Emulación. También se puede proporcionar Hardware Especial y Motores Fuera de Carga para ayudar al procesador 27 en la emulación de la función del Ordenador Anfitrión 50.

En un *mainframe*, las instrucciones de máquina con arquitectura son utilizadas por los programadores, por lo general hoy en día programadores de "C" a menudo mediante una aplicación de compilación. Estas instrucciones almacenadas en el medio de almacenamiento se pueden ejecutar de forma nativa en un servidor IBM z/Arquitecture, o bien en máquinas que ejecutan otras arquitecturas. Pueden emularse en los servidores *mainframe* de IBM existentes y futuros y en otras máquinas de IBM (por ejemplo, servidores pSeries© y xSeries®). Se pueden ejecutar en máquinas que ejecutan Linux en una amplia variedad de máquinas que utilizan equipos fabricados por IBM®, Intel®, AMD™, Sun Microsystems y otros. Además de la ejecución en ese hardware bajo z/Arquitecture©, puede utilizarse Linux así como las máquinas que utilizan la emulación de Hércules, UMX, FSI (Fundamental Software, Inc.) o Platform Solutions, Inc. (PSI), en las que la ejecución generalmente es en un modo de emulación. En modo de emulación, el software de emulación es ejecutado por un procesador nativo para emular la arquitectura de un procesador emulado.

El procesador nativo 27 normalmente ejecuta el software de emulación 23 que comprende ya sea un firmware o un sistema operativo nativo para realizar la emulación del procesador emulado. El software de emulación 23 es el responsable de extraer y ejecutar las instrucciones de la arquitectura del procesador emulado. El software de emulación 23 mantiene un contador de programa emulado para realizar un seguimiento de las fronteras de las instrucciones. El software de emulación 23 puede extraer una o varias instrucciones de máquina emulada en un momento y convertir la una o varias instrucciones de la máquina emulada en un grupo correspondiente de instrucciones nativas de máquina para la ejecución por parte del procesador nativo 27. Estas instrucciones convertidas pueden guardarse en caché de modo que se puede conseguir una conversión más rápida. No obstante, el software de emulación debe mantener las normas de arquitectura de la arquitectura del procesador emulado con el fin de asegurar que los sistemas operativos y las aplicaciones escritas para el procesador emulado funcionen correctamente. Además, el software de emulación debe proporcionar los recursos identificados por la arquitectura del procesador de emulación 1 incluyendo, pero no limitado a, registros de control, registros de propósito general, registros de coma flotante, función de traducción dinámica de direcciones, incluyendo tablas de segmentos y tablas de páginas, por ejemplo, mecanismos de interrupción, mecanismos de cambio de contexto, relojes de la hora del día (TOD) e interfaces con arquitectura para subsistemas de E/S de tal manera que un sistema operativo o un programa de aplicación diseñado para ejecutarse en el procesador emulado, se puede ejecutar en el procesador nativo con el software de emulación.

Una instrucción específica que se emula se decodifica y se llama a una subrutina para realizar la función de la instrucción individual. Se implementa una función de software de emulación 23 que emula una función de un procesador emulado 1, por ejemplo, en un controlador o una subrutina "C", o algún otro método para proporcionar un controlador para el hardware específico como estará dentro de la habilidad de los expertos en la técnica después de entender la descripción de la realización preferida. Varias patentes de emulación de software y hardware, incluyendo pero no limitado al documento US 5551013 para un "Multiprocesador para la emulación de hardware" de Beausoleil y otros, y el documento US6009261: Preprocesamiento de rutinas almacenadas de destino para emular las instrucciones incompatibles en un procesador de destino" de Scalzi y otros; y el documento US5574873: Decodificación de instrucción invitada para acceder directamente a las rutinas de emulación que emulan a las instrucciones invitadas, de Davidian y otros; documento US6308255: Bus y chipset de multiproceso simétrico utilizado para apoyo a coprocesador que permite que el código no nativo se ejecute en un sistema, de Gorlshek y otros; y documento US6463582: Traductor de optimización dinámica de código objeto para la emulación de la arquitectura y método de traducción de optimización dinámica de código objeto de Lethin y otros; y documento US5790825: Método para emular instrucciones invitadas en un ordenador anfitrión a través de recompilación dinámica de las instrucciones anfitrión de Eric Traut; y muchos otros, ilustran la variedad de formas conocidas para lograr la emulación de un formato de instrucción con arquitectura para una máquina diferente para una máquina de destino disponibles para los expertos en la materia, así como las técnicas de software comercial utilizadas por los mencionados anteriormente.

Sumario de la invención

En una realización de la invención, un procesador de un sistema de procesamiento extrae y ejecuta una instrucción de máquina de atributos de caché definida para una arquitectura de ordenador, la instrucción de máquina de caché comprende un código de operación, un identificador de operando (un campo de registro), el identificador de operando identifica una ubicación de operando (un registro). La instrucción determina a partir de un campo de desplazamiento de la instrucción, un nivel de caché y un atributo de caché que se extrae de la caché en el nivel de

caché determinado, el atributo de caché es un resumen de cachés en cada nivel de caché del procesador o unos atributos de caché de las cachés en el nivel de caché especificado asociado con el procesador. Los atributos determinados se extraen de la caché y se guardan en la ubicación del operando.

En una realización, el atributo de caché que se va a extraer es uno de: un resumen de la topología de caché de una o varias cachés; un tamaño de línea de la caché de destino; un tamaño total de la caché de destino; o un nivel de asociatividad establecida de la caché de destino.

En otra realización, el resumen de topología de caché extraído comprende uno o varios resúmenes, cada resumen para una caché en un nivel especificado por el identificador de nivel de caché, en el que un resumen para una caché en el nivel de caché correspondiente consiste en cualquiera de: si existe una caché; si una caché es privada para el procesador que ejecuta la instrucción; si una cache puede ser compartida por otros procesadores del sistema de procesamiento; si una caché consiste en una caché separada de instrucciones y una caché separada de datos; si la caché es solo una caché de instrucciones; si la caché es solo una caché de datos; y si la caché es una caché unificada de instrucciones y datos.

La invención trata de proporcionar una nueva funcionalidad de instrucciones compatible con la arquitectura existente que alivia la dependencia de los recursos de la arquitectura como los registros generales, mejora la funcionalidad y el rendimiento de las versiones de software que emplean la nueva instrucción.

Breve descripción de los dibujos

La materia objeto que se considera como la invención se puntualiza particularmente y se reivindica claramente en las reivindicaciones a la conclusión de la memoria descriptiva. Los objetos, características y ventajas anteriores y otros de la invención serán más claros a partir de la siguiente descripción detallada tomada junto con los dibujos que se acompañan, en los que:

La Figura 1A es un diagrama que representa un ejemplo de sistema informático anfitrión de la técnica anterior;

La Figura 1B es un diagrama que representa un ejemplo de sistema informático anfitrión emulado de la técnica anterior;

La Figura 1C es un diagrama que representa un ejemplo de sistema informático de la técnica anterior;

La Figura 2 es un diagrama que representa un ejemplo de red de ordenadores de la técnica anterior;

La Figura 3 es un diagrama que representa unos elementos de un sistema informático de la técnica anterior;

Las Figuras 4A-4C representan los elementos detallados de un sistema informático de la técnica anterior;

Las Figuras 5A-5F representan un formato de instrucciones de máquina de un sistema informático;

La Figura 6 representa un ejemplo de formato de instrucción de una realización de la invención;

La Figura 7 representa un identificador de atributo de acuerdo con una realización de la invención; y

La Figura 8 representa un flujo de un ejemplo de función de una realización de la invención.

Descripción detallada

En una realización, la invención puede ser practicada por software (a veces denominado Código Interno Bajo Licencia, Firmware, Micro-código, Mili-código, Pico-código y similares, cualquiera de los mismos sería compatible con la presente invención). Haciendo referencia a la figura 1A, un procesador, también conocido como CPU (*Central Processing Unit*) 1 del sistema 50, suele acceder a un código de programa de software que incorpora la presente invención, desde un almacenamiento de medios 7 a largo plazo, tal como una unidad de CD-ROM, unidad de cinta o disco duro. El código de programa de software puede incorporarse en cualquiera de una variedad de medios conocidos para su uso con un sistema de procesamiento de datos, 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 2 del ordenador o el almacenamiento de un sistema informático en una red 10 a otros sistemas informáticos para su uso por los usuarios de esos otros sistemas.

Como alternativa, el código del programa puede incorporarse en la memoria 1, y accederse a él por parte del procesador 1 utilizando el bus del procesador. Este código de programa incluye un sistema operativo que controla la función y la interacción de los diversos componentes del ordenador y uno o varios programas de aplicación. El código del programa es convocado normalmente desde medios de almacenamiento denso 11 para la memoria de alta velocidad 2 donde se encuentra disponible para su procesamiento por parte del procesador 1. Los métodos y técnicas para incorporar el código de programa de software en la memoria, en medios físicos y/o distribuir el código de software a través de redes son bien conocidos y no se explicarán adicionalmente en esta memoria. El código de programa, cuando se crea y se almacena en un medio tangible (incluyendo, pero limitado a, módulos de memoria

electrónica (RAM), memoria flash, discos compactos (CD), DVD, cintas magnéticas y similares se refiere a menudo como un "producto de programa informático". El medio de producto de programa informático suele ser legible por un circuito de procesamiento preferiblemente en un sistema informático para su ejecución por parte del circuito de procesamiento.

La figura 1C ilustra un sistema representativo de hardware de estación de trabajo o servidor en el que se puede poner en práctica la presente invención. El sistema 100 de la figura 1C comprende un sistema informático representativo 101, tal como un ordenador personal, una estación de trabajo o un servidor, que incluye unos dispositivos periféricos opcionales. La estación de trabajo 101 incluye uno o varios procesadores 106 y un bus empleado para conectar y permitir la comunicación entre el procesador o procesadores 106 y los otros componentes del sistema 101 de acuerdo con las técnicas conocidas. El bus conecta el procesador 106 con la memoria 105 y el almacenamiento a largo plazo 107, que puede incluir una unidad de disco duro (incluyendo cualquiera de los medios magnéticos, CD, DVD y Memoria Flash, por ejemplo) o una unidad de cinta, por ejemplo. El sistema 101 también podría incluir un adaptador de interfaz de usuario, que conecta el microprocesador 106 a través del bus con uno o varios dispositivos de interfaz, como un teclado 104, un ratón 103, una impresora/escáner 110 y/u otros dispositivos de interfaz, que puede ser cualquier dispositivo de interfaz de usuario, tal como una pantalla sensible al tacto, una tableta de entrada digital, etc. El bus también conecta un dispositivo de visualización 102, tal como una pantalla LCD o monitor, con el microprocesador 106 a través de un adaptador de pantalla.

El sistema 101 puede comunicarse con otros ordenadores o redes de ordenadores por medio de un adaptador de red capaz de comunicarse 108 con una red 109. Ejemplos de adaptadores de red son los canales de comunicación, Token Ring, Ethernet o módem. Como alternativa, la estación de trabajo 101 puede comunicarse utilizando una interfaz inalámbrica, tal como una tarjeta CDPD (*cellular digital packet data*). La estación de trabajo 101 puede estar asociada con otros ordenadores en una red de área local (LAN) o una red de área amplia (WAN), o la estación de trabajo 101 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 2 ilustra una red 200 de procesamiento de datos en la que se puede poner en práctica la presente invención. La red 200 de procesamiento de datos puede incluir una pluralidad de redes individuales, tales como una red inalámbrica y una red cableada, cada una de ellas puede incluir una pluralidad de estaciones de trabajo individuales 101, 201, 202, 203, 204. Además, como apreciarán los expertos en la técnica, se pueden incluir una o más redes de área local, en la que una LAN puede comprender una pluralidad de estaciones de trabajo inteligentes acopladas a un procesador anfitrión.

Aún en referencia a la figura 2, las redes también pueden incluir ordenadores o servidores *mainframe*, tales como un ordenador de pasarela (cliente-servidor 206) o servidor de aplicaciones (servidor remoto 208 que se puede acceder a un repositorio de datos y también se puede acceder a él directamente desde una estación de trabajo 205). Un ordenador de pasarela 206 sirve como punto de entrada en cada red 207. Una pasarela es necesaria cuando se conecta un protocolo de red a otro. La pasarela 206 puede acoplarse preferiblemente a otra red, (por ejemplo internet 207) por medio de un enlace de comunicaciones. La pasarela 206 también puede acoplarse directamente a una o varias estaciones de trabajo 101, 201, 202, 203, 204, mediante un enlace de comunicaciones. El ordenador de pasarela se puede implementar utilizando un servidor IBM eServer™ zSeries® z9® Server disponible de IBM Corp.

Al código de programación de software que incorpora la presente invención se suele acceder por parte del procesador 106 del sistema 101 desde unos medios de almacenamiento a largo plazo 107, tales como una unidad de CD-ROM o disco duro. El código de programación de software puede incorporarse en cualquiera de una variedad de medios conocidos para su uso 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 210, 211 desde la memoria o el almacenamiento de un sistema informático por una red a otros sistemas informáticos para su uso por parte los usuarios de esos otros sistemas.

Como alternativa, el código del programación 111 puede incorporarse en la memoria 105, y acceder a él por parte del procesador 106 utilizando el bus del procesador. Este código de programación incluye un sistema operativo que controla la función y la interacción de los diversos componentes del ordenador y uno o varios programas de aplicación 112. El código del programa es convocado normalmente desde medios de almacenamiento denso 107 para la memoria de alta velocidad 105 donde se encuentra disponible para su procesamiento por parte del procesador 106. Los métodos y técnicas para incorporar el código de programación de software en la memoria, en medios físicos y/o distribuir el código de software a través de redes son bien conocidos y no se explicarán adicionalmente en esta memoria. El código de programa, cuando se crea y se almacena en un medio tangible (incluyendo, pero limitado a, módulos de memoria electrónica (RAM), memoria flash, discos compactos (CD), DVD, cintas magnéticas y similares se conoce a menudo como un "producto de programa informático". El medio de producto de programa informático suele ser legible por un circuito de procesamiento preferiblemente en un sistema informático para su ejecución por el circuito de procesamiento.

La caché que es más fácilmente disponible para el procesador (normalmente más rápida y más pequeña que otras cachés del procesador) es la caché más baja (L1 o nivel uno) y el almacén principal (memoria principal) es la caché de más alto nivel (L3, si hay 3 niveles). La caché de nivel más bajo a menudo se divide en una caché de

instrucciones (I-Caché) que contiene las instrucciones de la máquina que se van a ejecutar y una caché de datos (D-Caché) que contiene los operandos de datos.

Haciendo referencia a la figura 3, se representa un ejemplo de realización de procesador para el procesador 106. Por lo general se emplean uno o varios niveles de caché 303 para el almacenamiento intermedio de los bloques de memoria con el fin de mejorar el rendimiento del procesador. La caché 303 es un búfer de alta velocidad que mantiene líneas de caché de datos de memoria que son susceptibles de ser utilizados. Líneas típicas de caché son 64, 128 o 256 bytes de datos de memoria. Las cachés separadas a menudo se emplean para almacenar en caché instrucciones en vez de almacenar en caché datos. La coherencia de caché (sincronización de las copias de las líneas en la memoria y las cachés) es proporcionada a menudo por diversos algoritmos "*Snoop*" (fisgoneo) bien conocidos en la técnica. El almacenamiento principal 105 de un sistema de procesador se conoce a menudo como una memoria caché. En un sistema de procesador con 4 niveles de caché 303 el almacenamiento principal 105 se conoce a veces como la memoria caché de nivel 5 (L5), ya que normalmente es más rápida y sólo contiene una parte del almacenamiento no volátil (DASD, cinta, etc.) que está disponible a un sistema informático. El almacenamiento principal 105 almacena en "caché" páginas de datos convocados en y desde el almacenamiento principal 105 por el sistema operativo.

Un contador de programa (contador de instrucciones) 311 realiza un seguimiento de la dirección de la instrucción en curso que va a ser ejecutada. Un contador de programa en un procesador de z/Arquitecture es de 64 bites y puede truncarse a 31 o 24 bites para soportar los límites de direccionamiento anteriores. Un contador de programa normalmente se incorpora en una PSW (palabra de estado de programa) de un ordenador que persiste durante el cambio de contexto. Por lo tanto, un programa en marcha, con un valor del contador de programa, puede ser interrumpido por ejemplo, por el sistema operativo (cambio de contexto desde el entorno del programa para el entorno del sistema operativo). La PSW del programa mantiene el valor del contador del programa, mientras que el programa no está activo, y el contador de programa (en la PSW) del sistema operativo se usa mientras el sistema operativo se está ejecutando. Normalmente, el contador de programa se incrementa en una cantidad igual al número de bytes de la instrucción en curso. Las instrucciones RISC (*Reduced Instruction Set Computing*: Computación con un conjunto de instrucciones complejo) son normalmente de longitud fija, mientras que las instrucciones CISC (*Complex Instruction Set Computing*: Computación con un conjunto de instrucciones reducido) son normalmente de longitud variable. Las instrucciones de la z/Arquitecture de IBM son instrucciones CISC que tienen una longitud de 2, 4 o 6 bytes. El contador 311 de Programa se modifica ya sea por una operación de cambio de contexto o una operación tomada de Salto de una instrucción de Salto, por ejemplo. En una operación de cambio de contexto, el valor de contador de programa en curso se guarda en una Palabra de Estado de Programa (PSW), junto con otra información de estado acerca del programa en ejecución (tal como códigos de condición), y un valor de contador de programa se carga apuntando a una instrucción de un nuevo módulo de programa que se va a ejecutar. Una operación tomada de salto se lleva a cabo con el fin de permitir que el programa tome decisiones o haga ciclos en el programa mediante la carga del resultado de la Instrucción de Salto en el Contador de Programa 311.

Normalmente, se emplea una Unidad 305 de Extracción de instrucciones para extraer las instrucciones en nombre del procesador 106. La unidad de extracción extrae las "siguientes instrucciones secuenciales", las instrucciones de destino de las instrucciones de salto tomadas, o las primeras instrucciones de un programa después de un cambio de contexto. Las unidades de extracción de Instrucciones Modernas emplean a menudo técnicas de extracción para extraer especulativamente instrucciones basándose en la probabilidad de que las instrucciones extraídas puedan ser utilizadas. Por ejemplo, una unidad de extracción puede extraer 16 bytes de la instrucción que incluye la siguiente instrucción secuencial y bytes adicionales de instrucciones secuenciales adicionales.

Las instrucciones extraídas son luego ejecutadas por el procesador 106. En una realización, las instrucciones extraídas se pasan a una unidad de despacho 306 de la unidad de extracción. La unidad de despacho decodifica las instrucciones y reenvía la información acerca de las instrucciones decodificadas a las unidades apropiadas 307, 308, 310. Una unidad de ejecución 307 normalmente recibirá información acerca de las instrucciones aritméticas decodificadas de la unidad 305 de extracción de instrucciones y realizará operaciones aritméticas sobre operandos de acuerdo con el código de operación de la instrucción. Se proporcionan operandos a la unidad de ejecución 307 preferiblemente ya sea desde la memoria 105, registros de arquitectura 309 o desde un campo inmediato de la instrucción que se ejecuta. Los resultados de la ejecución, cuando se almacenan, se almacenan en la memoria 105, en los registros 309 o en otro hardware de la máquina (tales como registros de control, registros de PSW y similares).

Un procesador 106 tiene normalmente una o varias unidades de ejecución 307, 308, 310, para ejecutar la función de la instrucción. Haciendo referencia a la figura 4A, una unidad de ejecución 307 puede comunicarse con registros generales 309 con arquitectura, una unidad de decodificación/despacho 306 una unidad de almacenamiento de carga 310 y otras unidades de procesador 401 a través de lógica 407 de interfaz. Una unidad de ejecución 307 puede emplear varios circuitos de registro 403, 404, 405, para guardar la información con la que operará la unidad de aritmética lógica (ALU) 402. La ALU realiza operaciones aritméticas como sumar, restar, multiplicar y dividir, así como la función lógica como AND, OR y EXCLUSIVE-OR (XOR), ROTATE y SHIFT. Preferiblemente, la ALU soporta operaciones especializadas que son dependientes del diseño. Otros circuitos pueden proporcionar otros mecanismos con arquitectura 408 incluyendo códigos de condición y la lógica de soporte de recuperación, por

ejemplo. Normalmente, el resultado de una operación ALU se lleva a cabo en un circuito de registro de salida 406, que puede reenviar el resultado a una variedad de funciones de procesamiento. Hay muchas disposiciones de las unidades de procesador, la presente descripción sólo se pretende que proporcione una comprensión representativa de una realización.

Una instrucción ADD, por ejemplo, sería ejecutada en una unidad de ejecución 307 con la funcionalidad aritmética y lógica, mientras que una instrucción de Coma Flotante, por ejemplo, sería ejecutada en una Ejecución de Coma Flotante con capacidad especializada de Coma Flotante. Preferiblemente, una unidad de ejecución opera sobre operandos identificados por una instrucción mediante la realización de una función definida de código de operación en los operandos. Por ejemplo, una instrucción ADD puede ser ejecutada por una unidad de ejecución 307 en los operandos que se encuentran en dos registros 309 identificados por los campos de registro de la instrucción.

La unidad de ejecución 307 realiza la suma aritmética de dos operandos y almacena el resultado en un tercer operando, donde el tercer operando puede ser un tercer registro, o uno de los dos registros de origen. La unidad de ejecución utiliza preferiblemente una Unidad Aritmética Lógica (ALU) 402 que es capaz de realizar una gran variedad de funciones lógicas, tales como SHIFT, ROTATE, AND, OR y XOR, así como una variedad de funciones algebraicas, incluyendo cualquiera de sumar, restar, multiplicar, dividir. Algunos ALU 402 se diseñan para operaciones escalares y algunas de coma flotante. Los datos pueden ser Big Endian (donde el byte menos significativo está en la dirección de byte más alto) o Little Endian (donde el byte menos significativo está en la dirección de byte más bajo) dependiendo de la arquitectura. El z/Arquitecture 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 en complemento a 2 es ventajoso porque la ALU no necesita designar una capacidad de restar ya que un valor negativo o un valor positivo en complemento a 2 sólo requiere una suma dentro de la ALU. Los números se describen comúnmente en forma abreviada, donde un campo de 12 bites define una dirección de un bloque de 4096 bytes y se describe comúnmente como un bloque de 4 Kbytes (Kilo-bytes), por ejemplo.

Haciendo referencia a la figura 4B, la información de instrucciones de salto para la ejecución de una instrucción de salto se envía normalmente a una unidad de salto 308 que a menudo emplea un algoritmo de predicción de saltos, tal como una tabla 432 de historial de saltos para predecir el resultado del salto antes de que otras operaciones condicionales se hayan completado. El destino de la instrucción de salto en curso será extraído y ejecutado especulativamente antes de que las operaciones condicionales se hayan completado. Cuando las operaciones condicionales se han completado las instrucciones de salto ejecutadas especulativamente se han completado o desechado sobre la base de las condiciones de la operación condicional y el resultado especulado. Una instrucción típica de salto puede probar los códigos de condición y salto a una dirección de destino si los códigos de condición cumplen con el requisito de salto de la instrucción de salto, una dirección de destino puede calcularse en base a varios números incluyendo los que se encuentran en los campos de registro o un campo inmediato de la instrucción, por ejemplo. La unidad de salto 308 puede emplear una ALU 426 que tiene una pluralidad de circuitos de registros de entrada 427, 428, 429, y un circuito de registro de salida 430. La unidad de salto 308 puede comunicarse con los registros generales 309, la unidad de despacho de decodificación 306 u otros circuitos 425, por ejemplo.

La ejecución de un grupo de instrucciones que puede ser interrumpida por diversas razones, incluyendo 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 E/S que causa un cambio de contexto o una actividad de varios hilos de ejecución de una pluralidad de programas (en un entorno de varios hilos de ejecución), por ejemplo. Preferiblemente una acción de cambio de contexto guarda la información de estado acerca de un programa que se ejecuta actualmente y luego carga la información de estado acerca de otro programa que se invoca. La información de estado se puede guardar en los registros de hardware o en la memoria, por ejemplo. La información del estado comprende preferiblemente un valor de contador de programa que apunta a una siguiente instrucción que va a ser ejecutada, códigos de condición, información de traducción de memoria y contenido de registros con arquitectura. Una actividad de cambio de contexto puede ser ejercida por circuitos de hardware, programas de aplicación, programas del sistema operativo o el código de firmware (micro-código, pico-código o código interno bajo licencia (LIC) por sí solos o combinados.

Un procesador accede a operandos de acuerdo con los métodos de instrucción definidos. La instrucción puede proporcionar un operando inmediato que utiliza el valor de una parte de la instrucción, puede proporcionar uno o varios campos de registro de forma explícita que apunta a registros de propósito general o registros de propósito especial (registros de coma flotante por ejemplo). La instrucción puede utilizar registros implicados identificados por un campo de código de operación como operandos. La instrucción puede utilizar ubicaciones de memoria para los operandos. Una ubicación de memoria de un operando puede ser proporcionada por un registro, un campo inmediato o una combinación de registros y el campo inmediato como se ejemplifica mediante el mecanismo de largo desplazamiento de z/Architecture en la que la instrucción define un Registro de Base, un Registro de Índice y un campo inmediato (campo de desplazamiento) que se suman juntos para proporcionar la dirección del operando en la memoria, por ejemplo. La ubicación en esta memoria normalmente implica una ubicación en la memoria principal (almacenamiento principal) a menos que se indique lo contrario.

Haciendo referencia a la figura 4C, un procesador accede al almacenamiento utilizando una unidad de Carga/Almacén 310. La unidad de Carga/Almacén 310 puede realizar una operación de Carga mediante la obtención de la dirección del operando de destino en la memoria 303 y cargar el operando en un registro 309 u otra ubicación de memoria 303, o puede realizar una operación de Almacenamiento mediante la obtención de la dirección del operando de destino en la memoria 303 y almacenar los datos obtenidos de un registro 309 o u otra ubicación de memoria en la ubicación del operando de destino en la memoria 303. La unidad de Carga/Almacén 310 puede ser especulativa y puede acceder a la memoria en una secuencia que está fuera de servicio con relación a la secuencia de instrucciones, sin embargo la unidad de Carga/Almacén 310 debe mantener la apariencia de programas de los que se ejecutaron instrucciones en orden. Una unidad de Carga/Almacén 310 puede comunicarse con registros generales 309, unidad de decodificación/despacho 306, (interfaz de Caché/Memoria 303 u otros elementos 455 y comprende varios circuitos de registro, ALU 458 y lógica de control 463 para el cálculo de direcciones de almacenamiento y para proporcionar la secuencia de tuberías (*pipeline*) para mantener las operaciones en orden. Algunas operaciones pueden estar fuera de servicio, pero la unidad de Carga/Almacén proporciona una funcionalidad para hacer que las operaciones fuera de servicio aparezcan al programa como si se hubieran sido realizado en orden, como es bien conocido en la técnica.

Preferiblemente las direcciones que un programa de aplicación "ve" a menudo se denominan como direcciones virtuales. Las direcciones virtuales se denominan a veces como "direcciones lógicas" y "direcciones efectivas". Estas direcciones virtuales son virtuales porque son redirigidas a una ubicación de memoria física por parte de una gran variedad de tecnologías de Traducción Dinámica de Direcciones (DAT) 312, incluyendo, pero no limitado a, la anteposición simplemente de una dirección virtual con un valor de desfase, la traducción de la dirección virtual a través de una o varias tablas de traducción, las tablas de traducción comprenden preferiblemente por lo menos una tabla de segmentos y una tabla de páginas solas o combinadas, preferiblemente, la tabla de segmentos tiene una entrada que apunta a la tabla de páginas. En z/Architecture, se proporciona una jerarquía de traducción que incluye una primera tabla de región, una segunda tabla de región, una tercera tabla de región, una tabla de segmentos y una tabla opcional de páginas. El rendimiento de la traducción de direcciones a menudo se mejora mediante la utilización de un búfer de traducción anticipada de instrucciones (TLB), que comprende entradas de asignación de una dirección virtual a una ubicación de memoria física asociada. Las entradas se crean cuando DAT 312 traduce una dirección virtual utilizando las tablas de traducción. El uso posterior de la dirección virtual puede utilizar entonces la entrada de la TLB rápida en lugar de los lentos accesos secuenciales a la Tabla de Traducción. El contenido de TLB puede ser gestionado por una variedad de algoritmos de reemplazo, incluyendo LRU (el utilizado menos recientemente).

En el caso de que el Procesador sea un procesador de un sistema con varios procesadores, cada procesador tiene la responsabilidad de mantener los recursos compartidos, tales como E/S, cachés, TLB y memoria entrelazados para que haya coherencia. Normalmente se utilizarán tecnologías "*snoop*" en el mantenimiento de la coherencia de la caché. En un entorno de *Snoop*, cada línea de caché puede ser marcada como que es cualquiera de un estado compartido, un estado exclusivo, un estado cambiado, un estado inválido y similares con el fin de facilitar la compartición.

Las unidades de E/S 304 proporcionan al procesador unos medios para conectarse a dispositivos periféricos, incluyendo cintas, discos, impresoras, pantallas y redes, por ejemplo. Las unidades de E/S se presentan a menudo al programa informático mediante Controladores de software. En *Mainframes* tales como el z/Series de IBM, Adaptadores de Canal y Adaptadores de Sistemas Abiertos son unidades de E/S del *Mainframe* que proporcionan las comunicaciones entre el sistema operativo y los dispositivos periféricos.

La siguiente descripción de los Principios de Funcionamiento de z/Architecture describe una visión arquitectónica de un sistema informático:

Almacenamiento:

Un sistema informático incluye la información en la memoria principal, así como direccionamiento, protección y grabación de referencia y cambio. Algunos aspectos del direccionamiento incluyen el formato de las direcciones, el concepto de espacios de direcciones, los distintos tipos de direcciones y la forma en que se traduce un tipo de dirección en otro tipo de dirección. Algo del almacenamiento principal incluye ubicaciones de almacenamiento asignadas permanentemente. El almacenamiento principal proporciona al sistema un almacenamiento de datos de rápido acceso directamente direccionable. Tanto los datos como los programas deben ser cargados en la memoria principal (desde los dispositivos de entrada) antes de que puedan ser procesados.

El almacenamiento principal puede incluir uno o más almacenamientos temporales más pequeños de acceso más rápido, a veces llamados cachés. Una caché se asocia normalmente físicamente con una CPU o un procesador de E/S. Los efectos, excepto en el rendimiento, de la construcción física y el uso de distintos medios de almacenamiento por lo general no son observables por el programa.

Las cachés separadas pueden ser mantenidas para los operandos de instrucciones y de datos. La información dentro de una caché se mantiene en bytes contiguos en una frontera integral denominado bloque de caché o línea de caché (o línea, para abreviar). Un modelo puede proporcionar una instrucción EXTRACT CACHE ATTRIBUTE

que devuelve el tamaño de una línea de caché en bytes. Un modelo también puede proporcionar instrucciones de PREFETCH DATA y PREFETCH DATA RELATIVE LONG que afecta a la extracción de almacenamiento en caché de datos o de instrucciones o la liberación de datos de la caché.

El almacenamiento es considerado como una larga cadena horizontal de bites. Para la mayoría de las operaciones, el acceso al almacenamiento se realiza en una secuencia de izquierda a derecha. La cadena de bites se divide a su vez en unidades de ocho bites. Una unidad de ocho bites se denomina byte, que es el bloque constructivo básico de todos los formatos de información. Cada ubicación de byte en el almacenamiento se identifica por un único entero no negativo, que es la dirección de esa ubicación de byte o, simplemente, la dirección del byte. Ubicaciones adyacentes de bytes tienen direcciones consecutivas, empezando por 0 a la izquierda y continuando en una secuencia de izquierda a derecha. Las direcciones son números enteros binarios sin signo y son de 24, 31 o 64 bites. La información se transmite entre el almacenamiento y una CPU o un subsistema de canal de un byte o un grupo de bytes, a la vez. A menos que se especifique lo contrario, un grupo de bytes en el almacenamiento es direccionado por el byte de más a la izquierda del grupo. El número de bytes en el grupo es implicado o explícitamente especificado por la operación a realizar. Cuando se utiliza en una operación de la CPU, un grupo de bytes se llama un campo. Dentro de cada grupo de bytes, los bites se numeran en una secuencia de izquierda a derecha. Los bites de más a la izquierda se denominan a veces como bites de "orden superior" y los bites de más a la derecha como bites de "orden inferior". Sin embargo los números de bites no son las direcciones de almacenamiento. Sólo pueden direccionarse bytes. Para trabajar en los bites individuales de un byte en el almacenamiento, es necesario acceder a todo el byte. Los bites en un byte se numeran del 0 al 7, de izquierda a derecha. Los bites en una dirección pueden ser numerados 8-31 o 40-63 para las direcciones de 24 bites o 1-31 o 33-63 para las direcciones de 31 bites; se numeran 0-63 para las direcciones de 64 bites. En cualquier otro formato de longitud fija de varios bytes, los bites que forman el formato son numerados consecutivamente empezando en 0. Para los efectos de detección de errores, y preferiblemente para la corrección, uno o más bites de comprobación se puede transmitir con cada byte o con un grupo de bytes. Tales bites de comprobación son generados automáticamente por la máquina y no pueden ser controlados directamente por el programa. La capacidad de almacenamiento se expresa en número de bytes. Cuando la longitud de un campo de almacenamiento operando es implicado por el código de operación de una instrucción, el campo se dice que tiene una longitud fija, que puede ser uno, dos, cuatro, ocho o dieciséis bytes. Para algunas instrucciones se pueden implicar campos más grandes. Cuando la longitud de un campo de operando de almacenamiento no está implicada pero se establece explícitamente, el campo se dice que tiene una longitud variable. Los operandos de longitud variable pueden variar en longitud en incrementos de un byte. Cuando la información se coloca en el almacenamiento, solo se sustituye el contenido de esas ubicaciones de bytes que se incluyen en el campo designado, incluso cuando el ancho de la ruta física de almacenamiento puede ser mayor que la longitud del campo que se almacena.

Determinadas unidades de información deben estar en una frontera integral en el almacenamiento. Una frontera se llama integral para una unidad de información cuando su dirección de almacenamiento es un múltiplo de la longitud de la unidad en bytes. Se dan nombres especiales a los campos de 2, 4, 8 y 16 bytes en una frontera integral. Una media-palabra es un grupo de dos bytes consecutivos en una frontera de dos bytes y es el bloque constructivo básico de instrucciones. Una palabra es un grupo de cuatro bytes consecutivos en una frontera de cuatro bytes. Una doble-palabra es un grupo de ocho bytes consecutivos en una frontera de ocho bytes. Una cuádruple-palabra es un grupo de 16 bytes consecutivos en una frontera de 16 bytes. Cuando las direcciones de almacenamiento designan medias palabras, palabras, dobles-palabras y cuádruples-palabras la representación binaria de la dirección contiene uno, dos, tres o cuatro bites de cero de más a la derecha. Las instrucciones deben estar en fronteras integrales de dos bytes. Los operandos de almacenamiento de la mayoría de las instrucciones no tienen requisitos de alineación de frontera.

En los modelos que implementan cachés separadas para instrucciones y operandos de datos, se puede experimentar un retraso significativo si el programa almacena en una línea de caché a partir de la que se extraen instrucciones posteriormente, independientemente de si el almacén altera las instrucciones que se extraen posteriormente.

Instrucciones:

Normalmente, el funcionamiento de la CPU está controlado por las instrucciones en el almacenamiento que se ejecutan de forma secuencial, una cada vez, de izquierda a derecha en una secuencia ascendente de direcciones de almacenamiento. Un cambio en el funcionamiento secuencial puede ser causado por saltos, LOAD PSW, interrupciones, órdenes SIGNAL PROCESSOR, o intervención manual.

Preferentemente una instrucción comprende dos partes principales:

- Un código de operación (código op), que especifica la operación a realizar
- Opcionalmente, la designación de los operandos que participan.

En las figuras 5A-5F se muestran formatos de instrucción de la z/Architecture . Una instrucción puede proporcionar simplemente un código de operación 501, o un código de operación y una gran variedad de campos que incluyen

operandos inmediatos o especificadores de registro para la colocación de los operandos en registros o en la memoria. El código de operación puede indicar al hardware que se van a utilizar recursos implicados (operandos, etc.) tales como uno o varios registros específicos de propósito general (GPR). Los operandos pueden agruparse en tres clases: los operandos situados en los registros, los operandos inmediatos y los operandos en el almacenamiento. Los operandos pueden diseñarse explícita o implícitamente. Los operandos de registro pueden encontrarse en registros en general, de coma flotante, de acceso o de control, con el tipo de registro identificado por el código de operación. El registro que contiene el operando se especifica mediante la identificación del registro en un campo de cuatro bites, llamado el campo R, en la instrucción. Para algunas instrucciones, un operando se encuentra en un registro designado implícitamente, dicho registro es implicado por el código de operación. Los operandos inmediatos se encuentran dentro de la instrucción, y el campo de 8-bites, 16-bites o 32-bites que contiene el operando inmediato se llama el campo 1. Los operandos en el almacenamiento pueden tener una longitud implicada; ser especificado por una máscara de bites; ser especificados por una especificación de longitud de cuatro bites u ocho bites, llamado el campo L, o tener una longitud especificada por el contenido de un registro general. Las direcciones de los operandos en el almacenamiento se especifican por medio de un formato que utiliza el contenido de un registro general como parte de la dirección. Esto hace que sea posible:

Especificar una dirección completa mediante el uso de una notación abreviada

Realizar la manipulación de direcciones utilizando instrucciones que emplean registros generales para los operandos

Modificar las direcciones por medios de programa, sin alteración de la secuencia de instrucciones

Funcionar independientemente de la ubicación de las áreas de datos al utilizar directamente las direcciones recibidas de otros programas

La dirección utilizada para referirse al almacenamiento se encuentra en un registro designado por el campo R en la instrucción o se calcula a partir de una dirección base, el índice y el desplazamiento, especificados por los campos B, X y D, respectivamente, en la instrucción. Cuando la CPU está en el modo de acceso a registro, un campo B o R podrá designar a un registro de acceso, además de ser utilizado para especificar una dirección. Para describir la ejecución de las instrucciones, los operandos son preferiblemente designados como operandos primero y segundo y, en algunos casos, operandos tercero y cuarto. En general, los dos operandos participan en una ejecución de la instrucción, y el resultado sustituye al primer operando.

Una instrucción es una, dos o tres medias palabras de largo y debe estar ubicada en el almacenamiento en una frontera de media palabra. Haciendo referencia a las figuras 5A - 5F que representan formatos de instrucciones, cada instrucción tiene uno de los 25 formatos básicos: E 501, I 502, RI 503 504, RIE 505 551 552 553 554, RIL 506 507, RIS 555, RR 510, RRE 511, RRF 512 513 514, RRS, RS 516 517, RSI 520, RSL 521, RSY 522 523, RX 524, RXE 525, RXF 526, RXY 527, S 530, SI 531, SIL 556, SIY 532, SS 533 534 535 536 537, SSE 541 y SSF 542, con tres variantes de la RRF, dos de RI, RIL, RS, y RSY, cinco de RIE y SS.

Los nombres de formato indican, en términos generales, las clases de los operandos que participan en la operación y algunos detalles acerca de los campos

• RIS denota una operación de registro e inmediato y una operación de almacenamiento.

• RRS denota una operación de registro y registro y una operación de almacenamiento.

• SIL denota una operación de almacenamiento e inmediato, con un campo inmediato de 16 bites.

En los formatos I, RR, RS, RSI, RX, SI, y SS, el primer byte de una instrucción contiene el código op. En los formatos E, RRE, RRF, S, SIL, y SSE, los dos primeros bytes de la instrucción contienen el código op, excepto que para algunas instrucciones en el formato S, el código op es sólo el primer byte. En los formatos RI y RIL, el código op está en el primer byte y posiciones de bites 12-15 de una instrucción. En el RIE, RIS, RRS, RSL, RSY, RXE, RXF, RXY y SIY, el código op está en el primer byte y el sexto byte de una instrucción. Los dos primeros bites del primer o único byte del código op especifican la longitud y el formato de la instrucción, de la siguiente manera:

En los formatos RR, RRE, RRF, RRR, RX, RXE, RXF, RXY, RS, RSY, RI, RIE y RIL, el contenido del registro designado por el campo R1 se llama el primer operando. El registro que contiene el primer operando se denomina a veces como la "primera ubicación de operando", y, a veces como "registro R1". En los formatos RR, RRE, RRF y RRR, el campo R2 designa el registro que contiene el segundo operando, y el campo R2 puede designar el mismo registro como R1. En los formatos RRF, RXF, RS, RSY, RSI y RIE, el uso del campo R3 depende de la instrucción. En los formatos RS y RSY, el campo R3 en cambio puede ser un campo M3 que especifica una máscara. El campo R1 designa un registro general o de acceso en las instrucciones generales, un registro general en las instrucciones de control y un registro de coma flotante o un registro general en las instrucciones de coma flotante. Para los registros generales y de control, el operando de registro está en posiciones de bites 32-63 del registro de 64 bites u ocupa todo el registro, dependiendo de la instrucción.

En el formato I, el contenido del campo de datos inmediatos de ocho bites, el campo I de la instrucción, se utilizan directamente como operando. En el formato SI, el contenido del campo de datos inmediatos de ocho bites, el campo 12 de la instrucción, se utilizan directamente como segundo operando. Los campos B1 y D1 especifican el primer operando, que tiene una longitud de un byte. En el formato SIY, el funcionamiento es el mismo, salvo que se utilizan campos DH1 y DL1 en lugar de un campo D1. En el formato R1 para las instrucciones ADD HALFWORD IMMEDIATE, COMPARE HALFWORD IMMEDIATE, LOAD HALF WORD IMMEDIATE y MULTIPLY HALFWORD IMMEDIATE el contenido del campo 12 de 16 bites de la instrucción se utilizan directamente como un entero binario con signo, y el campo R1 especifica el primer operando, que es de 32 o 64 bites de longitud, dependiendo de la instrucción. Para la instrucción TEST UNDER MASK (TMHH TMHL, TMLH, TMLL), el contenido del campo 12 se utiliza como una máscara, y el campo R1 especifica el primer operando, que es de 64 bites de longitud.

Para las instrucciones INSERT IMMEDIATE, AND INMMEDIATE, OR IMMEDIATE y LOAD LOGICAL IMMEDIATE, el contenido del campo 12 se utiliza como entero binario sin signo o un valor lógico, y el campo R1 especifica el primer operando, que es de 64 bites de longitud. Para las instrucciones relativas a saltos en los formatos RI y RSI, el contenido del campo 12 de 16 bites se utiliza como un entero binario con signo que designa una serie de medias palabras. Esta cifra, sumada a la dirección de la instrucción de salto, especifica la dirección del salto. Para instrucciones relativas a saltos en el formato RIL, el campo 12 es de 32 bites y se utiliza de la misma manera.

Para las instrucciones relativas a saltos en los formatos RI y RSI, el contenido del campo 12 de 16 bites se utiliza como un entero binario con signo que designa una serie de medias palabras. Esta cifra, sumada a la dirección de la instrucción de salto, especifica la dirección del salto. Para instrucciones relativas a saltos en el formato RIL, el campo 12 es de 32 bites y se utiliza de la misma manera. Para el formato RIE las instrucciones COMPARE IMMEDIATE AND BRANCH RELATIVE y COMPARE LOGICAL IMMEDIATE AND BRANCH RELATIVE, el contenido del campo 12 de 8 bites se utiliza directamente como segundo operando. Para el formato RIE las instrucciones COMPARE IMMEDIATE AND BRANCH, COMPARE IMMEDIATE AND TRAP, COMPARE LOGICAL IMMEDIATE AND BRANCH y COMPARE LOGICAL IMMEDIATE AND TRAP, el contenido del campo 12 de 16 bites se utiliza directamente como segundo operando. Para el formato RIE las instrucciones COMPARE AND BRANCH RELATIVE, COMPARE IMMEDIATE AND BRANCH RELATIVE, COMPARE LOGICAL AND BRANCH RELATIVE y COMPARE LOGICAL IMMEDIATE AND BRANCH RELATIVE, el contenido del campo 14 de 16 bites se utiliza como un entero binario con signo que designa un número de medias palabras que se añaden a la dirección de la instrucción para formar la dirección de salto.

Para las instrucciones de formato de RIL ADD IMMEDIATE, ADD LOGICAL IMMEDIATE, ADD LOGICAL WITH SIGNED IMMEDIATE, COMPARE IMMEDIATE, COMPARE LOGICAL IMMEDIATE, LOAD IMMEDIATE, y MULTIPLY SINGLE IMMEDIATE, el contenido del campo 12 de 32 bites se utiliza directamente como segundo operando.

Para las instrucciones del formato RIS, el contenido del campo 12 de 8 bites se utiliza directamente como segundo operando. Para el formato SIL, el contenido del campo 12 de 16 bites se utiliza directamente como segundo operando. Los campos B1 y D1 especifican el primer operando, como se describe a continuación.

En los formatos RSL, SI, SIL, SSE, y la mayoría de SS, el contenido del registro general designado por el campo B1 se añade al contenido del campo D1 para formar la dirección del primer operando. En los formatos RS, RSY, S, SIY, SS y SSE, el contenido del registro general designado por el campo B2 se añade al contenido del campo D2 o DH2 y los campos DL2 para formar la dirección del segundo operando. En los formatos RX, RXE, RXF y RXY contenido de los registros generales designados por los campos X2 y B2 se añade al contenido del campo D2 o DH2 y los campos DL2 para formar la dirección del segundo operando. En los formatos RIS y RRS, y en un formato SS, el contenido del registro general designado por el campo B4 se añade al contenido del campo D4 para formar la dirección del cuarto operando.

En el formato SS con un solo campo de longitud de ocho bites, para las instrucciones AND (NC), EXCLUSIVE OR (XC), MOVE (MVC), MOVE NUMERICS, MOVE ZONES y OR (OC), L especifica el número de bytes adicionales de operando a la derecha del byte designado por la dirección del primer operando. Por lo tanto, la longitud en bytes del primer operando es 1-256, que corresponde a un código de longitud en L de 0-255. Los resultados de almacenamiento sustituyen al primer operando y nunca se almacenan fuera del campo especificado por la dirección y longitud. En este formato, el segundo operando tiene la misma longitud que el primer operando. Hay variaciones de la definición anterior que se aplican a, EDIT, EDIT AND MARK, PACK ASCII, PACK UNICODE, TRANSLATE, TRANSLATE AND TEXT, UNPACK ASCII y UNPACK UNICODE.

En el formato SS con dos campos de longitud, y en el formato de RSL, L1 especifica el número de bytes adicionales de operando a la derecha del byte designado por la dirección del primer operando. Por lo tanto, la longitud en bytes del primer operando es 1-16, que corresponde a un código de longitud en L1 de 0-15. Similarmente, L2 especifica el número de bytes adicionales de operando a la derecha de la ubicación designada por la dirección del segundo operando. Los resultados sustituyen el primer operando y nunca son almacenados fuera del campo especificado por la dirección y longitud. Si el primer operando es más largo que el segundo, el segundo operando se prolonga a la izquierda con ceros hasta la longitud del primer operando. Esta prolongación no modifica el segundo operando en el almacenamiento. En el formato SS con dos campos R, tal como se utiliza por parte de las instrucciones MOVE TO

PRIMARY, MOVE TO SECONDARY y MOVE WITH KEY, el contenido del registro general especificado por el campo RI es un valor entero de 32 bites sin signo denominado la longitud verdadera. Los operandos son de una longitud denominada la longitud efectiva. La longitud efectiva es igual a la longitud real o 256, la que sea menor. Las instrucciones establecen el código de condición para facilitar la programación de un bucle para mover el número total de bytes especificados por la longitud real. El formato SS con dos campos R también se utiliza para especificar un intervalo de registros y dos operandos de almacenamiento para la instrucción LOAD MULTIPLE DISJOINT y para especificar uno o dos registros y uno o dos operandos de almacenamiento para la instrucción PERFORM LOCKED OPERATION.

Un cero en cualquiera de los campos B1, B2, X2 o B4 indica la ausencia del componente de dirección correspondiente. Para el componente ausente, se utiliza un cero para formar la suma intermedia, independientemente del contenido del registro general 0. Un desplazamiento de cero no tiene ningún significado especial.

Los bites 31 y 32 de la PSW en curso son los bites de modo de direccionamiento. El bit 31 es el bit de modo de direccionamiento extendido, y el bit 32 es el bit de modo de direccionamiento básico. Estos bites controlan el tamaño de la dirección efectiva producida por la generación de direcciones. Cuando los bites 31 y 32 de la PSW en curso son dos ceros, la CPU está en modo de direccionamiento de 24 bites, y se generan direcciones efectivas de operando e instrucción de 24 bites. Cuando el bit 31 de la PSW en curso es igual a cero y el bit 32 es uno, la CPU está en modo de direccionamiento de 31 bites, y se generan direcciones efectivas de operando e instrucción de 31 bites. Cuando los bites 31 y 32 de la PSW en curso son dos unos, la CPU está en modo de direccionamiento de 64 bites, y se generan direcciones efectivas de operando e instrucción de 64 bites. La ejecución de las instrucciones por la CPU implica la generación de las direcciones de instrucciones y operandos.

Cuando una instrucción se extrae desde el lugar designado por la PSW en curso, la dirección de la instrucción se incrementa el número de bytes de la instrucción y la instrucción se ejecuta. Los mismos pasos se repiten utilizando el nuevo valor de la dirección de la instrucción para extraer la instrucción siguiente en la secuencia. En el modo de direccionamiento de 24 bites, direcciones de instrucción se expanden, con la media palabra en la dirección de la instrucción 2^{24} - 2 seguida por la media palabra en la dirección 0 de la instrucción. De este modo, en el modo de direccionamiento de 24 bites, cualquier realización de posición de bit de PSW 104, como resultado de la actualización de la dirección de la instrucción, se pierde, en el modo de direccionamiento de 31 bites o 64 bites, las direcciones de las instrucciones se expanden de manera similar, con la media palabra en la dirección de instrucción 2^{31} - 2 o 2^{64} - 2, respectivamente, seguida de la media palabra en la dirección 0 de la instrucción. Una realización de posición 97 o 64 de bit de PSB, respectivamente, se pierde.

Una dirección de operando que se refiere al almacenamiento se deriva de un valor intermedio, que, o bien está contenido en un registro designado por un campo 11 en la instrucción o bien se calcula a partir de la suma de tres números binarios: dirección base, índice y desplazamiento. La dirección base (B) es un número de 64 bites contenido en un registro general especificado por el programa en un campo de cuatro bites, llamado el campo B, en la instrucción. Las direcciones de base pueden utilizarse como medios de direccionamiento independiente de cada programa y área de datos. En los cálculos de tipo matriz, se puede designar la ubicación de una matriz, y, en el procesamiento del tipo de registro, se puede identificar el registro. La dirección de base permite el direccionamiento de todo el almacenamiento. La dirección de base también se puede utilizar para la indexación.

El índice (X) es un número de 64 bites contenido en un registro general designado por el programa en un campo de cuatro bites, llamado el campo X, en la instrucción. Sólo se incluye en la dirección especificada por las instrucciones de los formatos RX, RXE y RXY. Las instrucciones de los formatos RX, RXE, RXF, y RXY permiten doble indexación; es decir, el índice puede ser utilizado para proporcionar la dirección de un elemento dentro de una matriz.

El desplazamiento (D) es un número de 12 bites o 20 bites contenidos en un campo, llamado campo D, en la instrucción. Un desplazamiento de 12 bites es sin signo y permite un direccionamiento relativo de hasta 4.095 bytes más allá de la ubicación designada por la dirección de base. Un desplazamiento de 20 bites es con signo y permite el direccionamiento relativo de hasta 524.287 bytes más allá de la ubicación de la dirección de base o de hasta 524.288 bytes antes. En los cálculos de tipo de matriz, el desplazamiento se puede utilizar para especificar uno de los muchos artículos asociados a un elemento. En el procesamiento de registros, el desplazamiento se puede utilizar para identificar los artículos dentro de un registro. Un desplazamiento de 12 bites se encuentra en las posiciones de bites 20-31 de las instrucciones de determinados formatos. En las instrucciones de algunos formatos, un segundo desplazamiento de 32 bites también está en la instrucción, en las posiciones de bites 36-47.

Un desplazamiento de 20 bites está en las instrucciones de sólo los formatos RSY, RXY o SIY. En estas instrucciones, el campo D consiste en un campo DL (bajo) en las posiciones de bites 20-31 y de un campo DH (alto) en las posiciones de bites 32-39. Cuando se instala el mecanismo de largo desplazamiento, el valor numérico del desplazamiento se forma añadiendo el contenido del campo DH a la izquierda del contenido del campo DL. Cuando no se instala el mecanismo de largo desplazamiento, el valor numérico del desplazamiento se forma añadiendo ocho bites de cero a la izquierda del contenido del campo DL y el contenido del campo DH se ignora.

En la formación de la suma intermedia, la dirección base y el índice se consideran como enteros binarios de 64 bites. Un desplazamiento de 12 bites es tratado como un entero binario de 12 bites sin signo y 52 bites de cero se añaden a la izquierda. Un desplazamiento de 20 bites es tratado como un entero binario de 20 bites con signo, y 44 bites iguales al bit de signo se añaden a la izquierda. Los tres se añaden como números binarios de 64 bites, ignorando el desbordamiento. La suma es siempre de 64 bites de longitud y se utiliza como un valor intermedio para formar la dirección generada. Los bites de los valores intermedios se numeran 0-63. Un cero en cualquiera de los campos B1, B2, X2 o B4 indica la ausencia del componente de dirección correspondiente. Para el componente ausente, se utilizan un cero para formar a la suma intermedia, independientemente del contenido del registro general 0. Un desplazamiento de cero no tiene ningún significado especial.

Cuando una descripción de instrucción especifica que el contenido de un registro general designado por un campo R se utiliza para direccionar un operando en el almacenamiento, el contenido del registro se utiliza como el valor intermedio de 64 bites.

Una instrucción puede designar al mismo registro general para el cálculo de la dirección y como la ubicación de un operando. El cálculo de dirección se completa antes de que los registros, si los hay, sean cambiados por la operación. A menos que se indique de otro modo en una definición de instrucción individual, la dirección de operando generada designa el byte de más a la izquierda de un operando en el almacenamiento.

La dirección del operando generada siempre es de 64 bites de largo, y los bites se numeran 0-63. La forma en que se obtiene la dirección generada a partir del valor intermedio depende del modo de direccionamiento en curso. En el modo de direccionamiento de 24 bites, los bites 0-39 del valor intermedio se ignoran, los bites 0-39 de la dirección generada se ven obligados a ser cero y los bites 40-63 del valor intermedio se convierten en los bites 40-63 de la dirección generada. En el modo de direccionamiento de 31 bites, los bites 0-32 del valor intermedio se ignoran, los bites 0-32 de la dirección generada se ven obligados a ser cero y los bites 33-63 del valor intermedio se convierten en los bites 33-63 de la dirección generada. En el modo de direccionamiento de 64 bites, los bites 0-63 del valor intermedio se convierten en los bites 0-63 de la dirección generada. Los valores negativos se pueden utilizar en los registros de dirección de base y de índice. Los bites 0-32 de estos valores se ignoran en el modo de direccionamiento de 31 bites, y los bites 0-39 se ignoran en el modo de direccionamiento de 24 bites.

Para instrucciones de salto, la dirección de la siguiente instrucción que se ejecutará cuando se tome el salo se llama la dirección de salto. Dependiendo de la instrucción de salto, el formato de la instrucción puede ser RR, RRE, RX, RXY, RS, RSY, RSI, RI, RIE o RIL. En los formatos RS, RSY, RX y RXY, la dirección de salto se especifica mediante una dirección base, un desplazamiento, y, en los formatos RX y RXY, un índice. En estos formatos, la generación del valor intermedio sigue las mismas reglas que para la generación del valor intermedio de dirección de operando. En los formatos RR y RRE, el contenido del registro general designado por el campo R2 se utiliza como valor intermedio del que se forma la dirección de salto. El registro general 0 no se puede designar como contenedor de una dirección de salto. Un valor de cero en el campo de R2 hace que la instrucción sea ejecutada sin saltos.

Las instrucciones de salto relativo están en formato RSL, RI, RIE y RIL. En los formatos RSI, RI y RIE para las instrucciones de salto relativo, el contenido del campo 12 se trata como un entero binario con signo de 16 bites que designa una serie de medias palabras. En el formato RIL, el contenido del campo 12 se trata como un entero binario de 32 bites con signo que designa una serie de medias palabras. La dirección de salto es el número de medias palabras designado por el campo 12 añadido a la dirección de la instrucción de salto relativo.

El valor intermedio de 64 bites para una instrucción de salto relativo en el formato RSI, RI, RIE o RIL es la suma de dos sumandos, con desbordamiento desde la posición de bit 0 ignorado. En el formato RSI, RI o RIE, el primer sumando es el contenido del campo 12 con un bit cero añadido a la derecha y 47 bites iguales al bit de signo del contenido anexado a la izquierda, excepto que para COMPARE AND BRANCH RELATIVE, COMPARE IMMEDIATE AND BRANCH RELATIVE, COMPARE LOGICAL AND BRANCH RELATIVE y COMPARE LOGICAL IMMEDIATE AND BRANCH RELATIVE, el primer sumando es el contenido del campo 14, con los bites anexados como se ha descrito anteriormente para el campo 12. En el formato RIL, el primer sumando es el contenido del campo 12 con un bit de cero anexado a la derecha y los 31 bites iguales al bit de signo del contenido anexado a la izquierda. En todos los formatos, el segundo sumando es el de la dirección de 64 bites de la instrucción de salto. La dirección de la instrucción de salto es la dirección de la instrucción en el PSW antes de que la dirección se actualice para direccionar la siguiente instrucción secuencial, o es la dirección del destino de la instrucción EXECUTE si se utiliza EXECUTE. Si se utiliza EXECUTE en el modo de direccionamiento de 24 bites o 31 bites, la dirección de la instrucción de salto es la dirección de destino con 40 o 33 ceros, respectivamente, anexados a la izquierda.

La dirección de salto es siempre de 64 bites de largo, con los bites numerados 0-63. La dirección de salto sustituye a los bites 64-127 de la PSW en curso. La forma en que se obtiene la dirección de salto a partir del valor intermedio depende del modo de direccionamiento. Para aquellas instrucciones de salto que cambian el modo de direccionamiento, se utiliza el nuevo modo de direccionamiento. En el modo de direccionamiento de 24 bites, los bites 0-39 del valor intermedio se ignoran, los bites 0-39 de la dirección de salto se hacen ceros y los bites 40-63 del valor intermedio se convierten en los bites 40-63 de la dirección de salto. En el modo de direccionamiento de 31 bites, los bites 0-32 del valor intermedio se ignoran, los bites 0-32 de la dirección de salto se hacen ceros y los bites

33-63 del valor intermedio se convierten en los bites 33-63 de la dirección de salto. En el modo de direccionamiento de 64 bites, los bites 0-63 del valor intermedio se convierten en los bites 0-63 de la dirección de salto.

Para varias instrucciones de salto, los saltos dependen de la satisfacción de una condición especificada. Cuando la condición no se cumple, no se toma el salto, la ejecución normal de instrucciones secuenciales continúa, y la dirección de salto no se utiliza. Cuando, se toma un salto, los bites 0-63 de la dirección de salto sustituyen a los bites 64-127 de la PSW en curso. La dirección de salto no se utiliza para el acceso al almacenamiento como parte de la operación de salto. Una excepción de las especificaciones debido a una extraña dirección de salto y las excepciones de acceso debidas a la extracción de la instrucción en ubicación del salto no se reconocen como parte de la operación de salto, sino que se reconocen como excepciones asociadas con la ejecución de la instrucción en la ubicación del salto.

Una instrucción de salto, tal como BRANCH AND SAVE, puede designar el mismo registro general, para el cálculo de la dirección de salto y como la ubicación de un operando. El cálculo de dirección de salto se completa antes de que el resto de la operación se lleve a cabo.

La palabra de estado del programa (PSW), que se describe en el capítulo 4 "Control" contiene la información necesaria para la ejecución del programa apropiado. La PSW se utiliza para controlar la secuencia de instrucciones y para mantener e indicar el estado de la CPU en relación con el programa que se está ejecutando actualmente. La PSW activa o de control se llama la PSW en curso. Las instrucciones de salto desempeñan las funciones de toma de decisiones, control de bucles y vinculación de subrutinas. Una instrucción de salto afecta a la secuencia de instrucciones al introducir una nueva dirección de instrucción en la PSW en curso. Las instrucciones de salto relativo con un campo 12 de 16 bites permite los saltos a una ubicación con un desplazamiento de hasta más de 64K - 2 bytes o menos 64K bytes con relación a la ubicación de la instrucción de salto, sin el uso de un registro de base. Las instrucciones de salto relativo con un campo 12 de 32 bites permite el salto a una ubicación con un desplazamiento de hasta más de 4G - 2 bytes o menos 4G bytes con relación a la ubicación de la instrucción de salto, sin el uso de un registro de base.

Unos mecanismos para la toma de decisiones son proporcionados por las instrucciones BRANCH ON CONDITION, BRANCH RELATIVE ON CONDITION y BRANCH RELATIVE ON CONDITION LOG. Estas instrucciones inspeccionan una condición, el código que refleja el resultado de la mayoría de las operaciones aritméticas, lógicas y operaciones de E/S. El código de condición, que consta de dos bites, permite cuatro posibles ajustes de código de condición: 0, 1, 2 y 3.

El significado específico de cualquier ajuste depende de la operación que establece el código de condición. Por ejemplo, el código de condición refleja dichas condiciones, como cero, distinto de cero, primer operando alto, igual, desbordamiento y subcanal ocupado. Una vez establecido, el código de condición se mantiene sin cambios hasta que sea modificado por una instrucción que hace que se establezca un código de condición diferente.

El control por bucle se puede realizar mediante el uso BRANCH ON CONDITION, BRANCH RELATIVE ON CONDITION y BRANCH RELATIVE ON CONDITION LONG para poner a prueba el resultado de las operaciones aritméticas y de recuento. Para algunas combinaciones particularmente frecuentes de aritmética y de pruebas se proporcionan BRANCH ON COUNT, BRANCH ON INDEX HIGH y BRANCH ON INDEX LOW OR EQUAL, y también se proporcionan equivalentes a salto relativo de estas instrucciones. Estos saltos, al estar especializados, proporcionan un mayor rendimiento para estas tareas.

La vinculación de subrutinas cuando no se necesita un cambio en el modo de direccionamiento se proporciona mediante las instrucciones BRANCH AND LINK y BRANCH AND SAVE. (Esta explicación de BRANCH AND SAVE se aplica también a BRANCH RELATIVE AND SAVE y BRANCH RELATIVE AND SAVE LONG.) Estas dos instrucciones permiten no sólo la introducción de una nueva dirección de instrucción, sino también la preservación, de una dirección de retorno y la información asociada. La dirección de retorno es la dirección de la instrucción siguiente a la instrucción de salto en el almacenamiento, excepto que es la dirección de la instrucción siguiente a una instrucción EXECUTE que tiene la instrucción de salto como su destino.

BRANCH AND LINK y BRANCH AND SAVE tienen un campo R1. Forman una dirección de salto por medio de campos que dependen de la instrucción. Las operaciones de las instrucciones que se resumen a continuación: • En el modo de direccionamiento de 24 bites, las dos instrucciones colocan la dirección de retorno en las posiciones de bites 40-63 del registro general R1 y dejan los bites 0-31 de ese registro sin cambios. BRANCH AND LINK coloca el código de longitud de instrucción para la instrucción y también el código de condición y la máscara de programa de la PSW en curso en las posiciones de bites 32-39 de registro R1 BRANCH AND SAVE coloca ceros en esas posiciones de bites.

• En el modo de direccionamiento de 31 bites, las dos instrucciones colocan la dirección de retorno en las posiciones de bites 33-63 y uno en la posición de bit 32 del registro general R1, y dejan los bites 0-31 del registro sin cambios.

• En el modo de direccionamiento de 64 bites, las dos instrucciones colocan la dirección de retorno en las posiciones de bites 0-63 del registro general R1.

• En cualquier modo de direccionamiento, las dos instrucciones generan la dirección de salto bajo el control del modo de direccionamiento en curso. Las instrucciones colocan los bites 0-63 de la dirección de salto en posiciones de bites 64-127 de la PSW. En el formato RR, las dos instrucciones no realizan ramificación si el campo R2 de la instrucción es cero.

Se puede observar que, en el modo de direccionamiento de 24 bites o 31 bites, BRANCH AND SAVE coloca el bit de modo de direccionamiento básico, bit 32 de la PSW, en la posición general de bit 32 de registro general RL BRANCH AND LINK lo hace en el modo de direccionamiento de 31 bites. Las instrucciones BRANCH AND SAVE AND SET MODE y BRANCH AND SET MODE son para utilizarlas cuando es necesario un cambio en el modo de direccionamiento durante la vinculación. Estas instrucciones tienen campos R1 y R2. Las operaciones de las instrucciones se resumen a continuación:

• BRANCH AND SAVE AND SET MODE establece el contenido del registro general R1 igual que BRANCH AND SAVE. Además, la instrucción coloca el bit de modo de direccionamiento ampliado, bit 31 de la PSW, en la posición de bit 63 del registro.

• BRANCH AND SET MODE, si R1 es distinto de cero, se comporta como sigue. En el modo de 24 o 31 bites, coloca el bit 32 de la PSW en la posición de bit 32 del registro general R1, y deja los bites 0-31 y 33-63 del registro sin cambios. Tenga en cuenta que el bit 63 del registro debe ser cero si el registro contiene una dirección de instrucción. En el modo de 64 bites, la instrucción coloca el bit 31 de la PSW (un uno) en la posición de bit 63 del registro general RL, y deja los bites 0-62 del registro sin cambios.

• Cuando R2 es distinto de cero, las dos instrucciones establecen el modo de direccionamiento y realizan los saltos de la siguiente manera. El bit 63 del registro general 112 se coloca en posición de bit 31 de la PSW. Si el bit 63 es cero, el bit 32 se coloca en la posición de bit 32 de la PSW. Si el bit 63 es uno, el bit 32 de PSW se establece en uno. A continuación, la dirección de salto se genera a partir del contenido del registro, excepto con el bit 63 del registro tratado como un cero, bajo el control del nuevo modo de direccionamiento. Las instrucciones colocan los bites 0-63 de la dirección de salto en posiciones de bites 64-127 de la PSW. El bit 63 del registro general R2 se mantiene sin cambios y, por tanto, puede ser un momento tras la entrada a un programa llamado. Si R2 es igual a R1, los resultados en el registro general designado son según lo especificado para el registro R1.

Interrupciones (cambio de contexto):

El mecanismo de interrupción permite a la CPU cambiar su estado como resultado de las condiciones externas a la configuración, dentro de la configuración, o dentro de la propia CPU. Para permitir una respuesta rápida a las condiciones de alta prioridad y reconocimiento inmediato del tipo de condición, las condiciones de interrupción se agrupan en seis clases: externas, entrada/salida, comprobación de la máquina, programa, reinicio y llamada al supervisor.

Una interrupción consiste en almacenar la PSW en curso como una PSW antigua, el almacenamiento de información que identifica la causa de la interrupción y la extracción de una nueva PSW. El procesamiento se reanuda como se especifica en la nueva PSW. La PSW antigua almacenada en una interrupción normalmente contiene la dirección de la instrucción que se habría ejecutado a continuación si la interrupción no se hubiera producido, lo que permite reanudar el programa interrumpido. Para las interrupciones de programa y llamar a supervisor, la información almacenada también contiene un código que identifica la longitud de la última instrucción ejecutada, lo que permite que el programa responda a la causa de la interrupción. En el caso de algunas condiciones de programa para las que la respuesta normal es re-ejecución de la instrucción que causa la interrupción, la dirección de la instrucción identifica directamente la última instrucción ejecutada.

A excepción de reiniciar, una interrupción puede ocurrir sólo cuando la CPU está en el estado de funcionamiento. La interrupción de reiniciar puede ocurrir con la CPU, ya sea en el estado detenido o de funcionamiento.

Cualquier excepción de acceso se reconoce como parte de la ejecución de la instrucción con la que está asociada la excepción. Una excepción de acceso no se reconoce cuando la CPU intenta extraer de una ubicación no disponible o detecta alguna otra condición de excepción de acceso, pero una instrucción de salto o una interrupción cambia la secuencia de instrucciones de tal manera que la instrucción no se ejecuta. Incluso la instrucción puede provocar que una excepción de acceso sea reconocida debido a la extracción de la instrucción. Además, las excepciones de acceso asociadas a la ejecución de instrucciones puede ocurrir debido a un acceso a un operando en el almacenamiento. Una excepción de acceso debida a la extracción de una instrucción se indica cuando la primera media palabra de instrucción no se puede extraer sin encontrar la excepción. Cuando la primera media palabra de la instrucción no tiene excepciones de acceso, las excepciones de acceso pueden ser indicadas para medias palabras adicionales de acuerdo a la longitud de la instrucción especificada por los dos primeros bites de la instrucción; sin embargo, cuando la operación se puede realizar sin acceder a la segunda o tercera medias palabras de la instrucción, es impredecible si la excepción de acceso se indica para la parte no utilizada. Ya que la indicación de las excepciones de acceso para la extracción de instrucción es común a todas las instrucciones, no está cubierto en las definiciones de instrucción individual.

Excepto cuando se indique lo contrario en la descripción de la instrucción individual, se aplicarán las siguientes reglas para las excepciones asociadas con un acceso a una ubicación de operando. Para un operando de tipo extracción, las excepciones de acceso se indican necesariamente sólo para esa parte del operando que se requiere para completar la operación, es impredecible si las excepciones de acceso se indican para aquellas partes de un operando tipo extracción que no son necesarias para completar la operación.

Para un operando de tipo almacén, las excepciones de acceso son reconocidas por todo el operando incluso si la operación podría completarse sin el uso de la parte inaccesible del operando. En situaciones en las que el valor de un operando de tipo almacén se define como impredecible, es impredecible si se indica una excepción de acceso. Cada vez que un acceso a una ubicación de operando puede provocar una excepción de acceso a reconocer, la palabra "acceso" se incluye en la lista de excepciones de programa en la descripción de la instrucción. Esta entrada también indica qué operando puede causar la excepción a reconocer y si la excepción se reconoce en una extracción o acceso a almacén a esa ubicación de operando. Las excepciones de acceso son reconocidas sólo por la parte del operando tal como se define para cada instrucción particular.

Una excepción de operación se reconoce cuando la CPU intenta ejecutar una instrucción con un código de operación no válido. El código de operación puede no estar asignado, o la instrucción con el código de operación puede no ser instalada en la CPU. La operación se suprime. El código de longitud de instrucción es 1, 2 o 3. La excepción de operación se indica mediante un código de interrupción de programa de 0001 hexadecimal (o 0081 hexadecimal si se indica un evento PER concurrente).

Algunos modelos pueden ofrecer instrucciones que no se describen en esta publicación, tales como las que se proporcionan para ayuda o como parte de características especiales o personalizadas. En consecuencia, los códigos de operación no descritos en esta publicación no necesariamente causan que una excepción de operación sea reconocida. Además, estas instrucciones pueden provocar que se establezcan modos de operación o pueden alterar de otra manera a la máquina de manera que afecte a la ejecución de las instrucciones siguientes. Para evitar que se produzca este tipo de operación, una instrucción con un código de operación no descrito en esta publicación debe ejecutarse solo cuando se desea la función específica asociada con el código de operación.

Una excepción de especificación se reconoce cuando cualquiera de las siguientes situaciones es cierta:

1. Se introduce un uno en una posición de bit sin asignar de la PSW (es decir, cualquiera de las posiciones de bit 0, 2-4, 24-30 o 33-63). Esto se maneja como una temprana excepción de especificación de PSW.

2. Se introduce un uno en la posición de bit 12 de la PSW. Esto se maneja como una temprana excepción de especificación de PSW.

3. La PSW es inválida en cualquiera de las siguientes maneras: a. Bit 31 de la PSW es uno y el bit 32 es igual a cero, b. Los bites 31 y 32 de la PSW son iguales a cero, lo que indica el modo de direccionamiento de 24 bites, y los bites 64-103 de la PSW no son todos cero, c. El bit 31 de la PSW es cero y el bit 32 es uno, lo que indica el modo de direccionamiento de 31 bites, y los bites 64-96 de la PSW no son todos cero. Esto se maneja como una temprana excepción de especificación de PSW.

4. La PSW contiene una dirección de instrucción impar.

5. Una dirección de operando no designa una frontera integral en una instrucción que requiere tal designación de frontera integral.

6. Un registro general con número impar es designado por un campo R de una instrucción que requiere una designación de registro con número par.

7. A registro de coma flotante distinto de 0, 1, 4, 5, 8, 9, 12 o 13 es designado para un operando extendido.

8. El multiplicador o el divisor en la aritmética decimal superior a 15 dígitos y signo.

9. La longitud del campo de primer operando es menor o igual a la longitud del campo de segundo operando en la multiplicación o división decimal.

10. Se intenta la ejecución de CIPHER MESSAGE, CIPHER MESSAGE WITH CHAINING, COMPUTE INTERMEDIATE MESSAGE DIGEST, COMPUTE LAST MESSAGE DIGEST o COMPUTE MESSAGE AUTHENTICATION CODE, y el código de función en los bites 57-63 del registro general 0 contiene un código de función no asignado o desinstalado.

11. Se intenta la ejecución de CIPHER MESSAGE o CIPHER MESSAGE WITH CHAINING, y el campo R1 o R2 designa un registro de número impar o registro general 0.

12. Se intenta la ejecución de CIPHER MESSAGE, CIPHER MESSAGE WITH CHAINING, COMPUTE INTERMEDIATE MESSAGE DIGEST o COMPUTE MESSAGE AUTHENTICATION CODE, y la longitud del segundo

operando no es un múltiplo del tamaño de bloque de datos de la función designada. Esta condición de excepción de especificación no se aplica a las funciones de consulta.

13. Se intenta la ejecución de COMPARE AND FORM CODEWORD, y los registros generales 1, 2 y 3 no contienen inicialmente valores pares.

32. Se intenta la ejecución de COMPARE AND SWAP AND STORE y se existe alguna de las siguientes condiciones:

• El código de función especifica un valor no asignado.

• La característica de almacén especifica un valor no asignado.

• El código de función es 0, y el primer operando no es designado en una frontera de palabra.

• El código de función es 1, y el primer operando no es designado en una frontera de doble palabra.

• El segundo operando no se ha designado en una frontera integral correspondiente al tamaño del valor de almacén.

33. Se intenta la ejecución de COMPARE LOGICAL LONG UNICODE o COMPARE LONG UNICODE, y el contenido del registro general R1 + 1 o R3 + 1 no especifica un número par de bytes.

34. Se intenta la ejecución de COMPARE LOGICAL STRING, MOVE STRING o SEARCH STRING, y los bites 32-55 del registro general 0 no son todos cero.

35. Se intenta la ejecución de COMPRESSION CALL, y los bites 48-51 del registro general 0 tienen cualquiera de los valores binarios 0000 y 0110-1111.

36. Se intenta la ejecución de COMPUTE INTERMEDIATE MESSAGE DIGEST, COMPUTE LAST MESSAGE DIGEST o COMPUTE MESSAGE AUTHENTICATION CODE, y cualquiera de las siguientes situaciones es cierta:

• El campo R2 designa un registro de número impar o registro general 0.

• El bit 56 del registro general 0 no es cero.

37. Se intenta la ejecución de CONVERT HFP TO BFP, CONVERT TO FIXED (BFP o HFP), o LOAD FP INTEGER (BFP), y el campo M3 no designa un modificador válido.

38. Se intenta la ejecución de DIVIDE TO INTEGER y el campo M4 no designa un modificador válido.

39. Se intenta la ejecución de EXECUTE y la dirección de destino es impar.

40. Se intenta la ejecución de EXTRACT STACKED STATE, y el código en las posiciones de bites 56-63 del registro general R2 es superior a 4 cuando el mecanismo ASN-an-LX-reuse no se instala o es superior a 5 cuando el mecanismo se instala.

41. Se intenta la ejecución de FIND LEFTMOST ONE, y el campo R1 designa un registro de número impar.

42. Se intenta la ejecución de INVALIDATE DAT TABLE ENTRY, y los bites 44-51 del registro general R2 no son todo ceros.

43. Se intenta la ejecución de LOAD FPC, y uno o varios bites del segundo operando correspondientes a los bites sin apoyo en el registro de FPC son uno.

44. Se intenta la ejecución de LOAD PAGE-TABLE-ENTRY ADDRESS y el campo M4 de la instrucción contiene un valor distinto de 0000-0100 binario.

45. Se intenta la ejecución de LOAD PSW y del bit12 de la doble palabra en la dirección del segundo operando es cero. Es en función del modelo si se reconoce o no esta excepción.

46. Se intenta la ejecución de MONITOR CALL, y las posiciones de bites 8-11 de la instrucción no contienen ceros.

47. Se intenta la ejecución de MOVE PAGE, y las posiciones de bites 48-51 del registro general 0 no contienen ceros o los bites 52 y 53 del registro son uno.

48. Se intenta la ejecución de PACK ASCII, y el campo L2 es mayor que 31.

49. Se intenta la ejecución PACK UNICODE, y el campo L2 es mayor que 63 o es par.

50. Se intenta la ejecución PERFORM FLOATING POINT OPERATION, el bit 32 del registro general 0 es cero, y uno o varios campos en los bites 3-63 no son válidos o designan a una función desinstalada.

51. Se intenta la ejecución de PERFORM LOCKED OPERATION, y cualquiera de lo siguiente es cierto: • El bit T, bit 55 del registro general 0 es cero, y el código de función en los bites 56-63 del registro no es válido. • Los bites 32-54 del registro general 0 no son todos cero. • En el modo de registro de acceso para los códigos de función que hacen uso de una lista de parámetros que contienen un ALET, el campo R3 es cero.

52. Se intenta la ejecución de PERFORM TIMING FACILITY FUNCTION, y cualquiera de lo siguiente es cierto: • El bit 56 del registro general 0 no es cero. • Los bites 57-63 del registro general 0 especifican un código de función asignada o desinstalada.

53. Se intenta la ejecución de PROGRAM TRANSFER o PROGRAM TRANSFER WITH INSTANCE y todo lo siguiente es cierto: • El bit de modo de direccionamiento ampliado en la PSW es cero. • El bit de modo de direccionamiento básico, bit 32, en el registro general designado por el campo R2 de la instrucción es cero. • Los bites 33-39 de la dirección de instrucción en el mismo registro no son todos ceros.

54. Se intenta la ejecución de RESUME PROGRAM, y cualquiera de las siguientes situaciones es verdadera:

• Los bites 31, 31 y 64-127 del campo PSW en el segundo operando no son válidos para la colocación en la PSW en curso. Se reconoce la excepción si algo de lo siguiente es verdadero: - Los bites 31 y 32 son cero y los bites 64-103 no son todos cero. - Los bites 31 y 32 son cero y uno, respectivamente, y los bites 64 a 96 no son todos cero. - Los bites 31 y 32 son uno y cero, respectivamente. - El bit 127 es uno.

• Los bites 0-12 de la lista de parámetros no son todos cero.

55. Se intenta la ejecución de SEARCH STRING UNICODE, y los bites 32-47 del registro general 0 no son todos cero.

56. Se intenta la ejecución de SET ADDRESS SPACE CONTROL o SET ADDRESS SPACE CONTROL, FAST, y los bites 52 y 53 de la dirección del segundo operando no son a la vez cero.

57. Se intenta la ejecución de SET ADDRESSING MODE (SAM24), y los bites 0-39 de la dirección no actualizada de instrucción en la PSW, los bites 64-103 de la PSW, no son todos cero.

58. Se intenta la ejecución de SET ADDRESSING MODE (SAM31), y los bites 0-32 de la dirección no actualizada de instrucción en la PSW, los bites 64-96 de la PSW, no son todos cero.

59. Se intenta la ejecución de SET CLOCK PROGRAMMABLE FIELD, y los bites 32-47 del registro general 0 no son todos cero.

60. Se intenta la ejecución de SET FPC, y uno o varios bites del primer operando correspondientes a los bites sin apoyo en el registro de FPC son uno.

61. Se intenta la ejecución de STORE SYSTEM INFORMATION, el código de función en el registro general 0 es válido, y alguna de las siguientes situaciones es verdadera: • Los bites 36-55 del registro general 0 y los bites 32 - 47 de un registro general 1 no son todos ceros. • La dirección del segundo operando no está alineada en una frontera de 4K-bytes.

62. Se intenta la ejecución de TRANSLATE TWO TO ONE o TRANSLATE TWO TO TWO, y la longitud en el registro general R1 + 1 no especifica un número par de bytes.

63. Se intenta la ejecución de UMPACK ASCII, y el campo L1 es mayor que 31.

64. Se intenta la ejecución de UNPACK UNICODE, y el campo L1 es mayor que 63 o es par.

65. Se intenta la ejecución de UPDATE TREE, y el contenido inicial de los registros generales 4 y 5 no son un múltiplo de 8 en el modo de direccionamiento de 24 bites o 31 bites o no son un múltiplo de 16 en el modo de direccionamiento de 64 bites. La ejecución de la instrucción identificada por el PSW antiguo se suprime. Sin embargo, para tempranas excepciones de especificación PSW (causas 1-3) la operación que introduce la nueva PSW se ha completado, pero se produce una interrupción inmediatamente después. Preferiblemente, el código de longitud de instrucción (ILC) es 1, 2 ó 3, lo que indica la longitud de la instrucción que causa la excepción. Cuando la dirección de la instrucción es impar (causa 4 en la página 6-33), es impredecible si el ILC es 1 2, o 3. Cuando la excepción es reconocida debido a una temprana excepción de especificación de PSW (causas 1-3) y la excepción ha sido introducida por LOAD PSW, LOAD PSW EXTENDED, PROGRAM RETURN, o una interrupción, el ILC es 0. Cuando la excepción se introduce por SET ADDRESSING MODE (SAM24, SAM31), el ILC es 1 o es 2 si SET ADDRESSING MODE era el destino de EXECUTE. Cuando la excepción es introducida por SET SYSTEM MASK o por STORE THEN OR SYSTEM MASK, el ILC es 2.

Las interrupciones del programa se utilizan para informar de las excepciones y eventos que se producen durante la ejecución del programa. Una interrupción del programa hace que la PSW antigua sea almacenada en ubicaciones reales 336-351 y una nueva PSW sea extraída de las ubicaciones reales 464-479. La causa de la interrupción es identificada por el código de interrupción. El código de interrupción se coloca en las ubicaciones reales 142-143, el código de longitud de instrucciones se coloca en posiciones de bites 5 y 6 del byte en la ubicación real 141 con el resto de los bites en cero, y se almacenan ceros en la ubicación real 140. Para algunas de las causas, la información adicional que identifica la causa de la interrupción se almacena en las ubicaciones reales 144-183. Si el mecanismo PER-3 se instala, entonces, como parte de la acción de la interrupción del programa, el contenido del registro dirección de eventos de ruptura se coloca en las ubicaciones de almacenamiento reales 272-279. Excepto para PER y la excepción de cripto-operación, la condición que causa la interrupción se indica con un valor codificado colocado en las posiciones de siete bites de más a la derecha del código de interrupción. Sólo se puede indicar una condición cada vez. Los bites 0-7 del código de interrupción se establecen a ceros. Los eventos PER se indican mediante la creación deL bit 8 del código de interrupción a uno. Cuando esta es la única condición, los bites 0-7 y 9-15 también se establecen a cero. Cuando un evento PER se indica al mismo tiempo que otra condición de interrupción del programa, el bit 8 es uno, y los bites 0-7 y 9-15 se establecen para la otra condición. La excepción de cripto-operación se indica mediante un código de interrupción de 0001 hexadecimal o 0199 hexadecimal si también se indica un evento PER.

Cuando hay un bit de máscara correspondiente, una interrupción del programa sólo puede ocurrir cuando el bit de la máscara es uno. La máscara del programa en la PSW controla cuatro de las excepciones, las máscaras IEEE en el registro FPC controlan las excepciones IEEE, el bit 33 en el registro de control 0 controla si SET SYSTEM MASK provoca una excepción de operación especial, los bites 48-63 en el registro de control 8 controlan las interrupciones debidas a los eventos de monitor, y una jerarquía de máscaras controla las interrupciones debidas a eventos PER. Cuando cualquier bit de máscara de control es cero, se ignora la condición; la condición no se queda pendiente.

Cuando la PSW nueva para una interrupción del programa tiene un error de formato de PSW o produce una excepción que se va a reconocer en el proceso de extracción de la instrucción, se puede producir una cadena de interrupciones de programa.

Algunas de las condiciones que se indican como excepciones de programa pueden ser reconocidas por el subsistema de canal, en cuyo caso la excepción se indica en la palabra de estado de subcanal o la palabra de estado ampliado.

Cuando una excepción de los datos provoca una interrupción del programa, un código de excepción de datos (DXC) se almacena en la ubicación 147, y se almacenan ceros en las ubicaciones 144-146. El DXC distingue entre los diferentes tipos de condiciones de excepción de datos. Cuando el bit de control de registro AFP (registro adicional de coma flotante), el bit 45 del registro de control 0, es uno, el DXC también se coloca en el campo de DXC del registro de control de coma flotante (FPC). El campo DXC en el registro FPC se mantiene sin cambios cuando se informa de cualquier otra excepción de programa. El DXC es un código de 8 bites que indica la causa específica de una excepción de los datos.

DXC 2 y 3 se excluyen mutuamente y son de mayor prioridad que cualquier otro DXC. De este modo, por ejemplo, el DXC 2 (instrucción BFP) tiene prioridad sobre cualquier excepción IEEE; y DXC 3 (instrucción DFP) tiene prioridad sobre cualquier excepción IEEE o excepción IEEE simulada. Como otro ejemplo, si existen las condiciones tanto para DXC 3 (instrucción DFP) como para DXC 1 (registro AFP), se informa DXC 3. Cuando se aplican a la vez una excepción especificación y una excepción de datos de registro AFP, será impredecible cual se informa.

Una excepción de direccionamiento se reconoce cuando la CPU intenta hacer referencia a una ubicación de almacenamiento principal que no está disponible en la configuración. Una ubicación de almacenamiento principal no está disponible en la configuración cuando la ubicación no está instalada, cuando la unidad de almacenamiento no está en la configuración o cuando la alimentación está apagada en la unidad de almacenamiento. Una dirección que designa una ubicación de almacenamiento que no está disponible en la configuración se conoce como inválida. La operación se suprime cuando la dirección de la instrucción es inválida. Similarmente, la operación se suprime cuando la dirección de la instrucción de destino de EXECUTE no es válida. Además, la unidad de operación se suprime cuando se encuentra una excepción de direccionamiento al acceder a una tabla o entrada de tabla. Las tablas y las entradas de tabla a las que se aplica la regla son la tabla de control de unidad despachable, la entrada de segunda tabla de ASN primaria, y las entradas de la lista de acceso, primera tabla de región, segunda tabla de región, tercera tabla de región, tabla de segmentos, tabla de páginas, tabla de vinculación, primera tabla de vinculación, segunda tabla de vinculación, tabla de entradas, primera tabla de ASN, segunda tabla de ASN, tabla de autoridad, pila de vinculación y tabla de trazas. Las excepciones de direccionamiento tienen como resultado una supresión cuando se encuentran para referencias a la primera tabla de región, segunda tabla de región, tercera tabla de región, tabla de segmentos y la tabla de páginas, tanto en referencias implícitas para la traducción dinámica de direcciones como para las referencias asociadas con la ejecución, de LOAD PAGE-TABLE-ENTRY ADDRESS, LOAD REAL ADDRESS, STORE REAL ADDRESS y TEST PROTECTION. Similarmente, las excepciones de direccionamiento para los accesos a la tabla de control de unidad despachable, entrada de segunda tabla de ASN primaria, lista de acceso, segunda tabla de ASN, tabla de autoridad tienen como resultado la supresión cuando se

encuentran en la traducción de registro de acceso hecha de forma implícita o como parte de LOAD PAGE-TABLE-ENTRY ADDRESS, LOAD REAL ADDRESS, STORE REAL ADDRESS, TEST ACCESS o TEST PROTECTION. A excepción de algunas instrucciones específicas cuya ejecución se suprime, la operación se termina por una dirección de operando que se puede traducir pero designa una ubicación no disponible. Para la terminación, los cambios pueden ocurrir sólo en campos de resultado. En este contexto, el término "campo de resultado" incluye el código de condición, registros y cualquier ubicación de almacenamiento que se proporciona y que se designa para ser cambiada por la instrucción.

EXTRACT CACHE ATTRIBUTE INSTRUCTION:

Haciendo referencia a la figura 8, cuando la instrucción EXTRACT CACHE-ATTRIBUTE FIG. 6 se extrae 801 y se ejecuta, la información 805 relativa al atributo especificado 804 del subsistema de almacenamiento 303 se coloca en la ubicación 806 del primer operando. La instrucción comprende un código de operación de 8 bites "EB" y una extensión de código de operación de 8 bites '4C', así como unos campos de registro R1, R3 y B2 y un campo de desplazamiento firmado DH2|DL2. El primer operando es un registro especificado por el campo de registro R1 de la instrucción y es de 64 bites.

La dirección del segundo operando no se utiliza para direccionar los datos; más bien, los 24 bites de más a la derecha de la dirección de 64 bites FIG. 7 son tratados como un código 804 que especifica qué atributo se devuelve al registro general especificado por R1. La dirección del segundo operando se calcula 802 sumando algebraicamente el valor ampliado de signo del campo de desplazamiento con signo (DH2||DL2) de la instrucción con el valor del registro especificado por el campo B2 de la instrucción cuando el campo B2 no es '0'. (Cuando el campo B2 es '0' el campo de desplazamiento con signo (DH2||DL2) se utiliza como la dirección del segundo operando).

Los códigos se definen como sigue:

Indicación de Atributo (AI): posiciones de los bites 56-59 de la dirección del segundo operando figura 7 contiene un entero sin signo de 4 bites que indica el atributo de caché que se extrae, de la siguiente manera:

0 Extracto de resumen de topología

1 Extracto de tamaño de línea de la caché, en bytes

1 Extracto de tamaño total de la caché, en bytes

3 Extracto de nivel de asociatividad establecida de la cache

15 Reservado

Indicación de nivel (LI): las posiciones de los bites 60-62 de la dirección del segundo operando FIG. 7 contienen un entero sin signo de 3 bites que indica el nivel de la caché para el que el atributo de caché se va a extraer, el 0 indica caché de primer nivel, 1 indica caché de segundo nivel y así sucesivamente. Si un nivel de caché no se implementa en un modelo, su indicación de nivel correspondiente se reserva.

Indicación de tipo (TI): El bit 63 de la dirección del segundo operando FIG. 7 indica el tipo de caché para la que el atributo de caché se va a extraer, el 0 indica caché de datos y 1 indica la caché de instrucciones. Cuando el nivel de caché tiene una caché unificada de datos e instrucciones, se devuelve el mismo resultado independientemente de la indicación del tipo.

Si la indicación de atributo es cero, las indicaciones de nivel y de tipo se ignoran.

Los bites 0-39 de la dirección del segundo operando se ignoran. Los bites 40-55 de la dirección del segundo operando se reservan y deben contener ceros. Si una posición de bit reservada en la dirección del segundo operando contiene uno, o si se especifica una indicación de atributo reservado o indicación de nivel, los bites 0-63 del registro general R1 se establecen en unos.

El contenido del registro general R3 se ignora, sin embargo el campo R3 debe especificar el registro 0; de lo contrario, el programa podría no funcionar de manera compatible en el futuro.

Si la indicación de atributo es igual a cero, se devuelve un resumen de cada nivel de caché, en general el registro R1. Cada campo de resumen es de ocho bites, en el que los bites 0-7 del registro contienen el resumen para la memoria caché de primer nivel, los bites 8-15 contienen el resumen para la memoria caché de segundo nivel, y así sucesivamente. El contenido de un campo de resumen de ocho bites es como sigue:

Resumen de campos

Bites	Significado
0-3	Reservados, almacenados como ceros
4-5	Alcance de caché, de la siguiente manera:

(00) la caché no existe en este nivel

(01) la caché es privada para la CPU

(10) la caché puede ser compartida por varias CPU

(11) Reservado

6-7 Cuando las posiciones de bites 4-5 contienen un valor distinto de cero, las posiciones de bites 6-7 contienen el tipo de caché, de la siguiente manera:

(00) Instrucción independiente y existen cachés de datos a este nivel

(01) Sólo existe una caché de instrucciones en este nivel

(10) Sólo existe una caché de datos en este nivel 11. Existe una instrucción unificada y caché de datos en este nivel

Cuando las posiciones de bites 4-5 contienen ceros, las posiciones de bites 6-7 contienen ceros.

Excepciones de Programa:

• Operación (si el mecanismo de extensión de instrucciones generales no está instalado)

Todos los demás códigos están reservados. Si se especifica un código reservado, los bites 0-63 del registro general R1 se establecen en unos. El contenido del registro general R3 se ignora, sin embargo el campo R3 debe especificar el registro 0; de lo contrario, el programa podría no funcionar de manera compatible en el futuro.

El conocimiento del tamaño de línea de caché es útil para determinar la colocación de las instrucciones de PREFETCH DATA y PREFETCH DATA RELATIVE LONG. Dependiendo del modelo, la caché de primer nivel puede ser implementada como una caché unificada que contenga los datos e instrucciones (en contraposición a una memoria caché dividida que tiene una caché de instrucciones y caché de datos por separado). En este caso, los códigos de función 0 y 1 devuelven el mismo tamaño que es el tamaño de la línea de la caché unificada, si un modelo no proporciona una caché de datos y el código de la función es 0, o si el modelo no proporciona una caché de instrucciones y el código de función es 1, entonces un valor de cero se coloca en la ubicación del primer operando.

Lo anterior es útil en la comprensión de la terminología y la estructura de una realización del sistema informático. La presente invención no se limita a z/Architecture o a la descripción proporcionada de la misma. La presente invención puede aplicarse ventajosamente a otras arquitecturas informáticas de otros fabricantes de ordenadores con la enseñanza de esta memoria.

REIVINDICACIONES

1. Un método de funcionamiento de un ordenador que comprende:

la extracción (801), por un procesador (106) de un sistema de procesamiento, de una instrucción de máquina de caché definida para una arquitectura informática, la instrucción de máquina de caché comprende un código de operación, un identificador del operando, el identificador del operando identifica una ubicación de operando;

la ejecución de la instrucción de máquina de caché comprende:

sobre la base de un identificador de nivel de caché, la determinación de una caché de destino;

sobre la base de un identificador de atributo de caché, la determinación de un atributo de caché (804) que se va a extraer;

la extracción (805) del atributo de caché determinado de la caché de destino; y

guardar (806) el atributo de caché extraído en la ubicación de operando identificada, en la que el atributo de caché que se va a extraer comprende cualquiera de:

un resumen de la topología de caché de una o varias cachés;

un tamaño de línea de la caché de destino;

un tamaño total de la caché de destino; o

un nivel de asociatividad establecida de la caché de destino.

2. El método según la reivindicación 1, en el que el identificador de operando consiste en un campo de primer registro que identifica un primer registro, en el que la determinación del identificador de atributo de caché comprende:

añadir algebraicamente el campo de desplazamiento de la instrucción a un valor asociado con un segundo campo de la instrucción para determinar el identificador de atributo de caché.

3. El método según la reivindicación 1, en el que el resumen de topología de caché extraído comprende uno o varios resúmenes, cada resumen para una caché en un nivel especificado por el identificador de nivel de caché, en el que un resumen para una caché en el nivel de caché correspondiente consiste en cualquiera de

si existe una caché;

si una caché es privada para el procesador que ejecuta la instrucción;

si una cache puede ser compartida por otros procesadores del sistema de procesamiento

si una caché consiste en una caché separada de instrucciones y una caché separada de datos;

si la caché es solo una caché de instrucciones;

si la caché es solo una caché de datos: y

si la caché es una caché unificada de instrucciones y datos.

4. El método según la reivindicación 1, en el que la instrucción de máquina de caché definida para la arquitectura informática se extrae y ejecuta mediante una unidad de procesamiento central de una arquitectura informática alternativa,

en el que el método comprende además la interpretación de la instrucción de máquina de caché para identificar a una rutina de software predeterminado para simular el funcionamiento de la instrucción de máquina de caché; y

en el que la ejecución de la instrucción de máquina de caché comprende la ejecución de la rutina de software predeterminado para ejecutar los pasos del método para la ejecución de la instrucción de máquina de caché.

5. Un producto de programa informático, el producto de programa informático comprende un medio de almacenamiento tangible legible por un circuito de procesamiento e instrucciones de almacenamiento para su ejecución por el circuito de procesamiento para la realización del método de cualquier reivindicación anterior.

6. Un sistema informático que comprende:

una memoria:

un procesador en comunicación con la memoria, el procesador comprende un elemento de extracción de instrucciones para la extracción de instrucciones de la memoria y uno o varios elementos de ejecución para la ejecución de las instrucciones extraídas;

en el que el sistema informático se configura para realizar el método según cualquiera de las reivindicaciones 1 a 4.

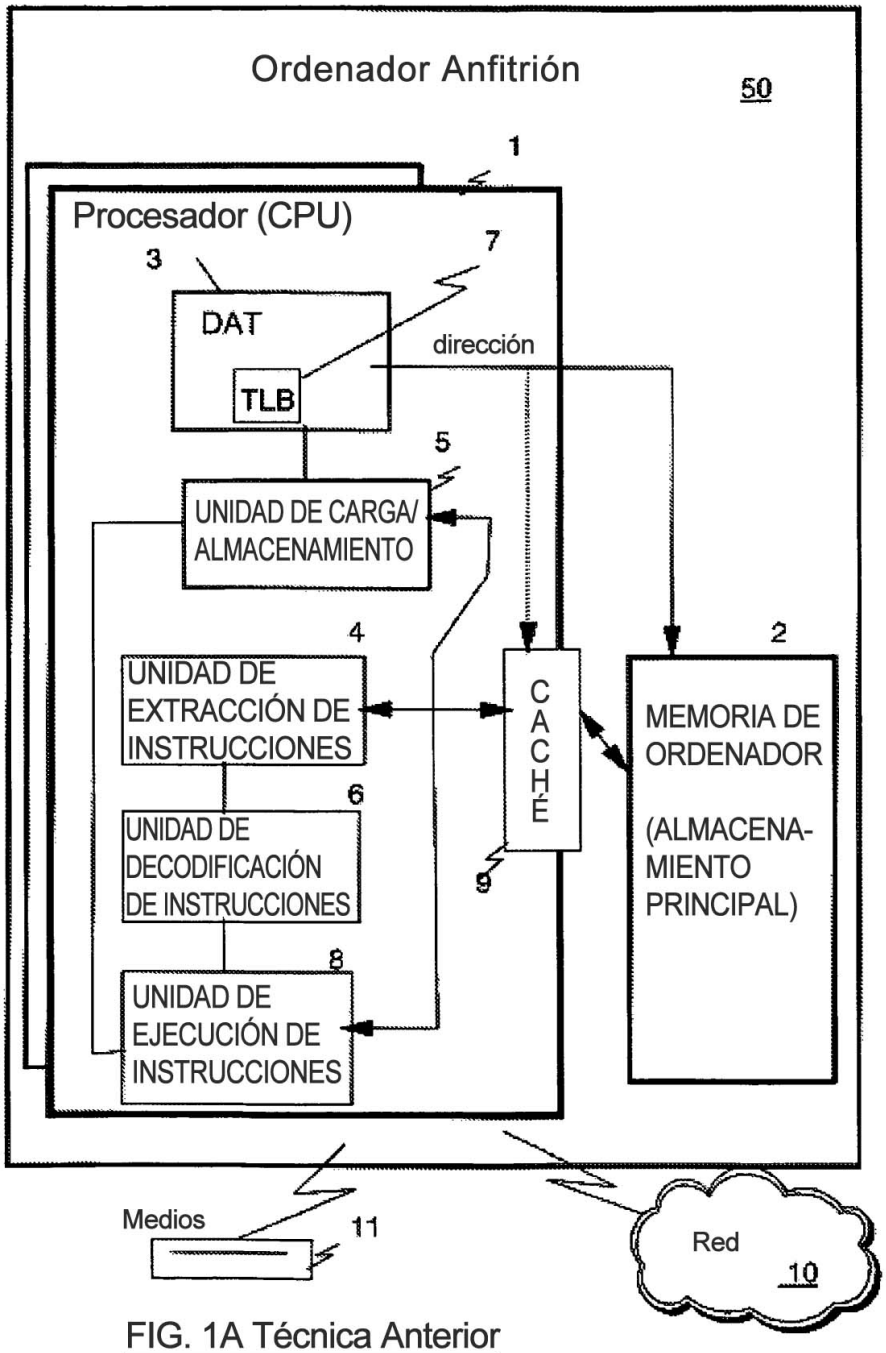

FIG. 1A Técnica Anterior

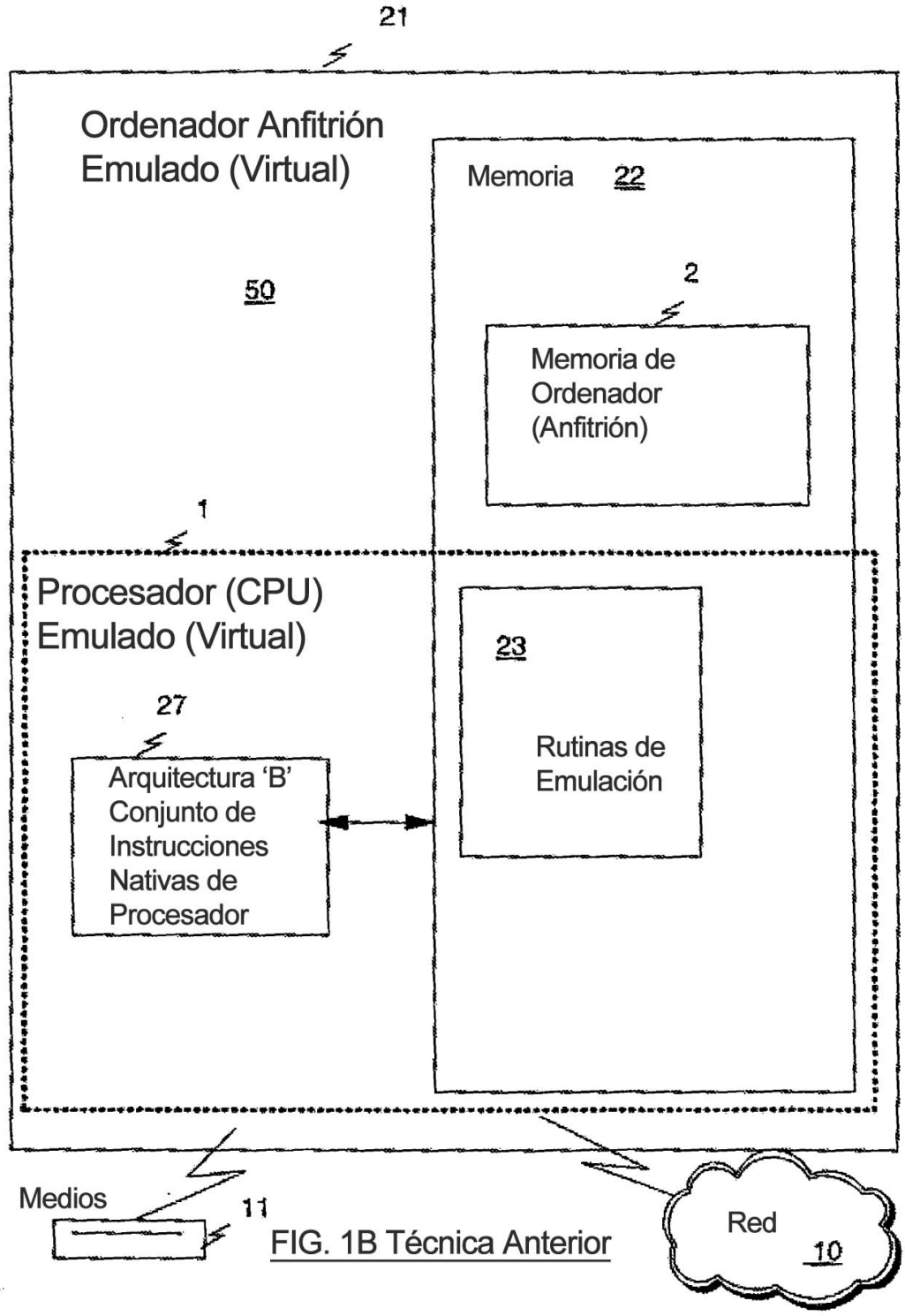

FIG. 1B Técnica Anterior

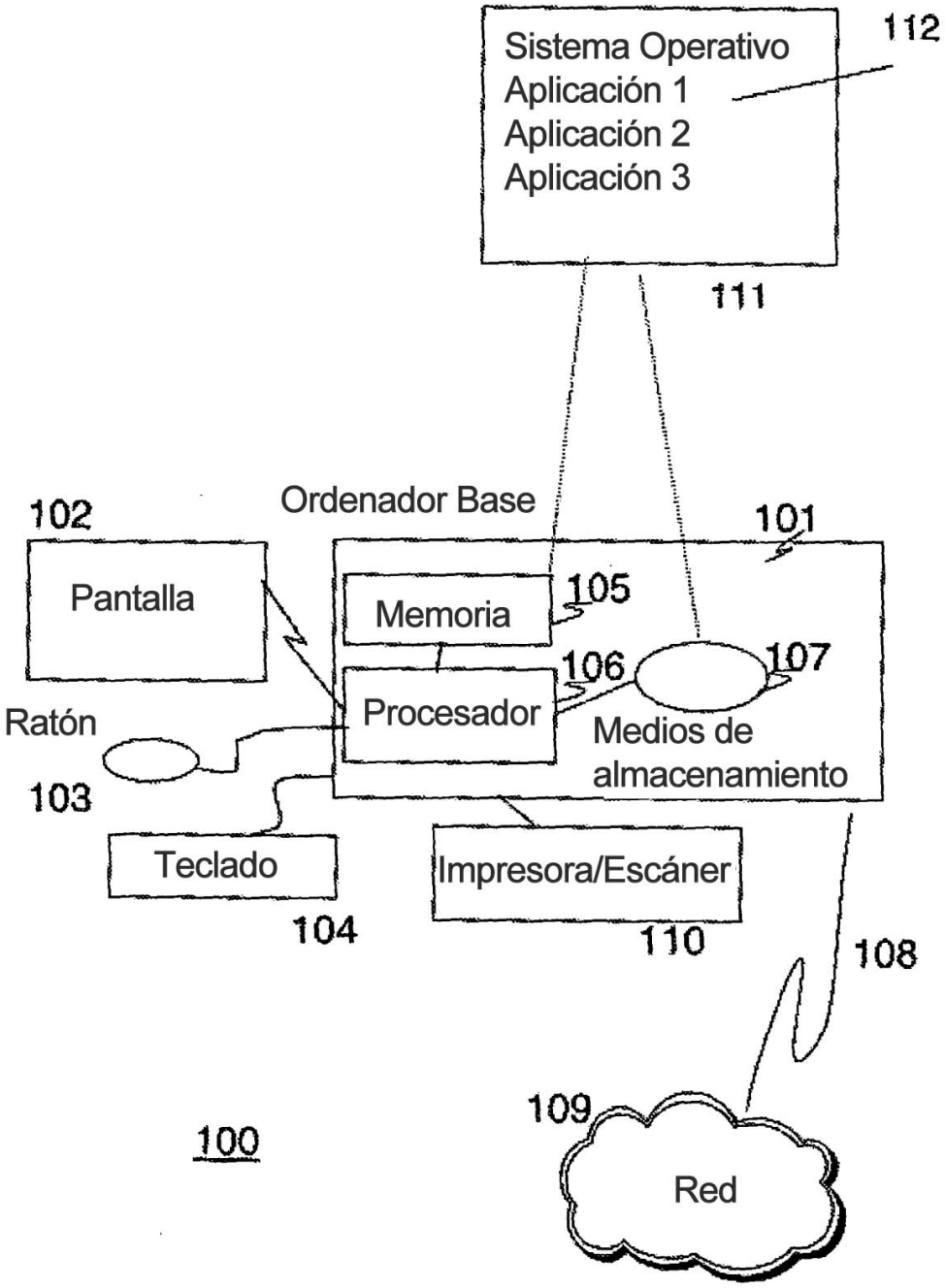

FIG. 1C Técnica Anterior

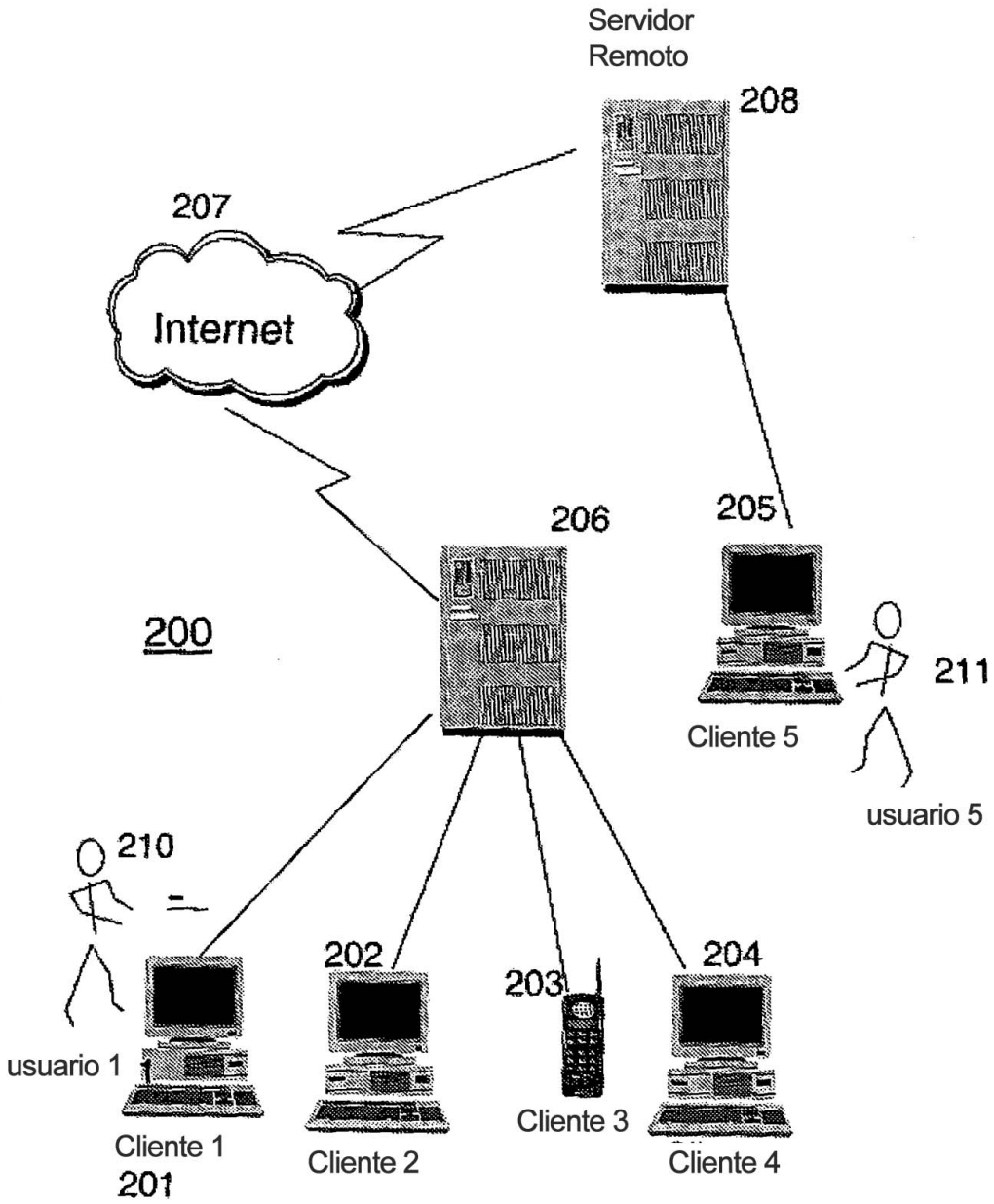

FIG. 2 Técnica Anterior

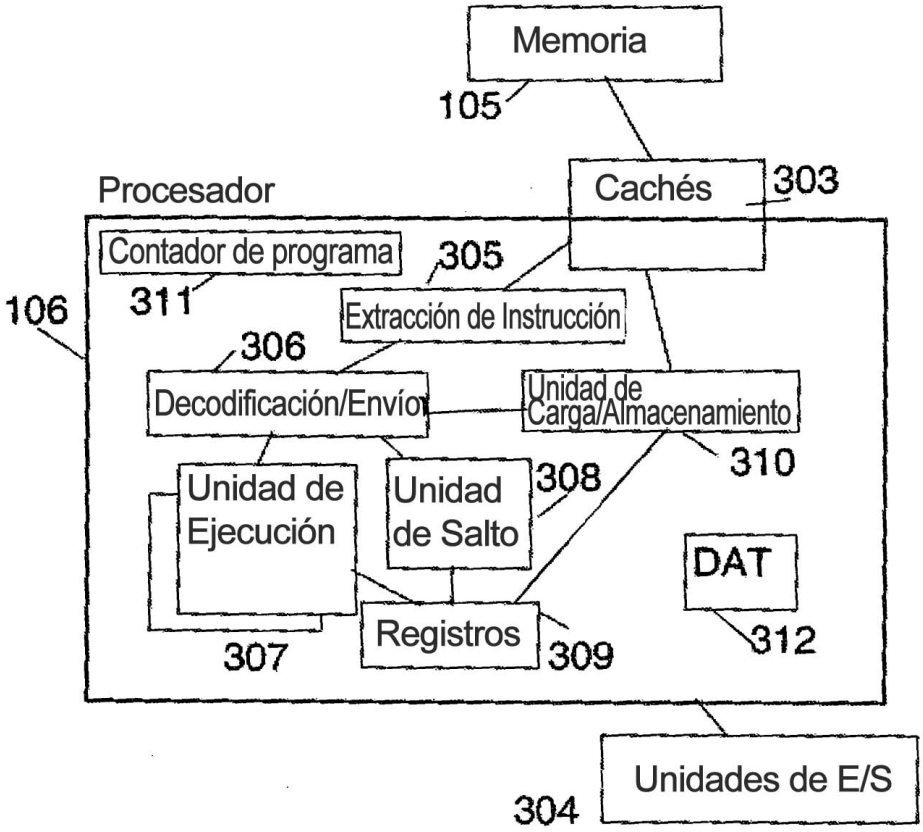

FIG. 3 Técnica Anterior

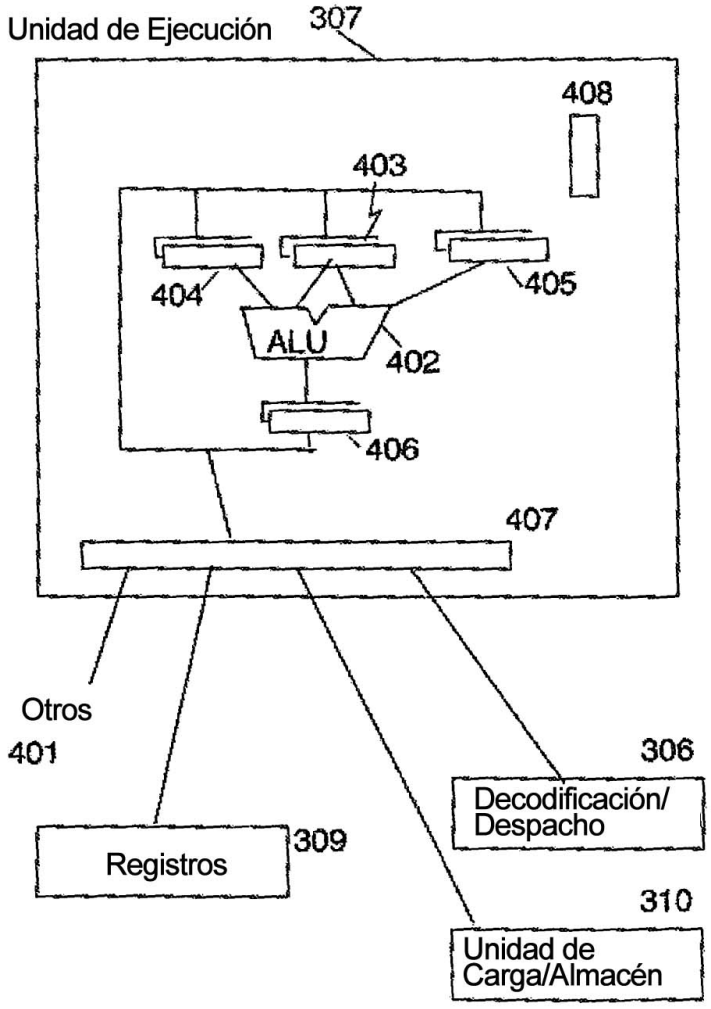

FIG. 4A Técnica Anterior

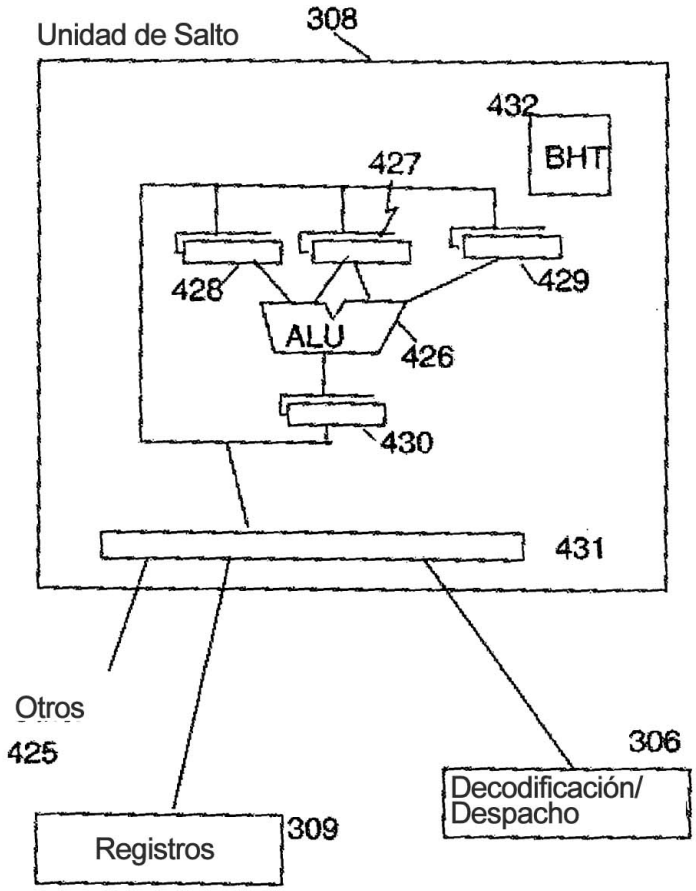

FIG. 4B Técnica Anterior

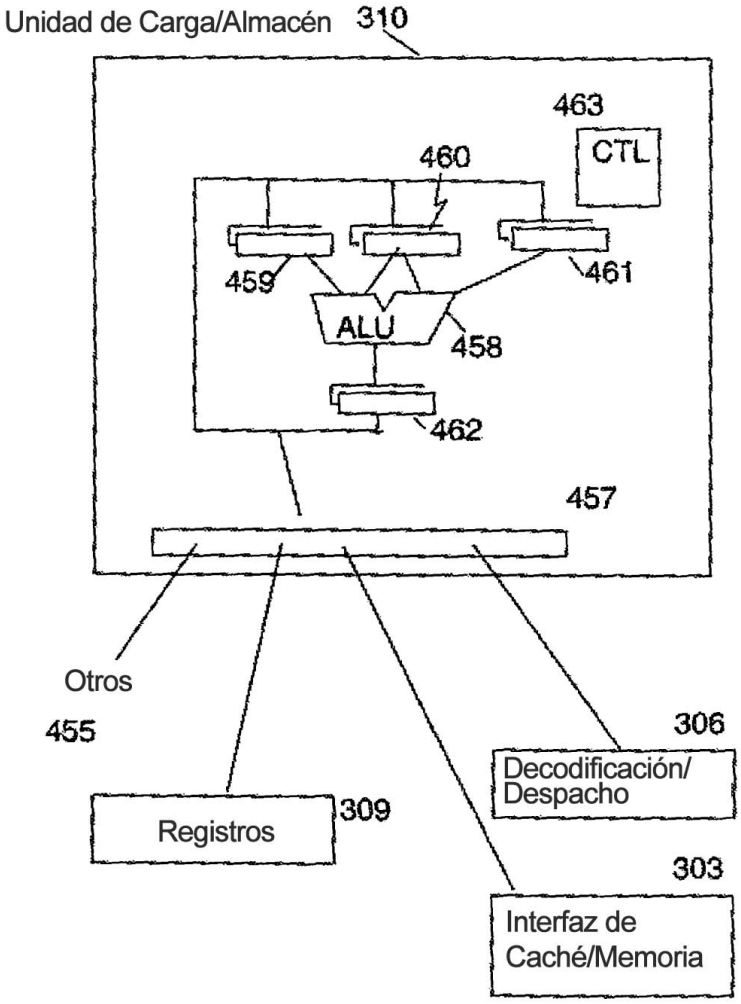

FIG. 4C Técnica Anterior

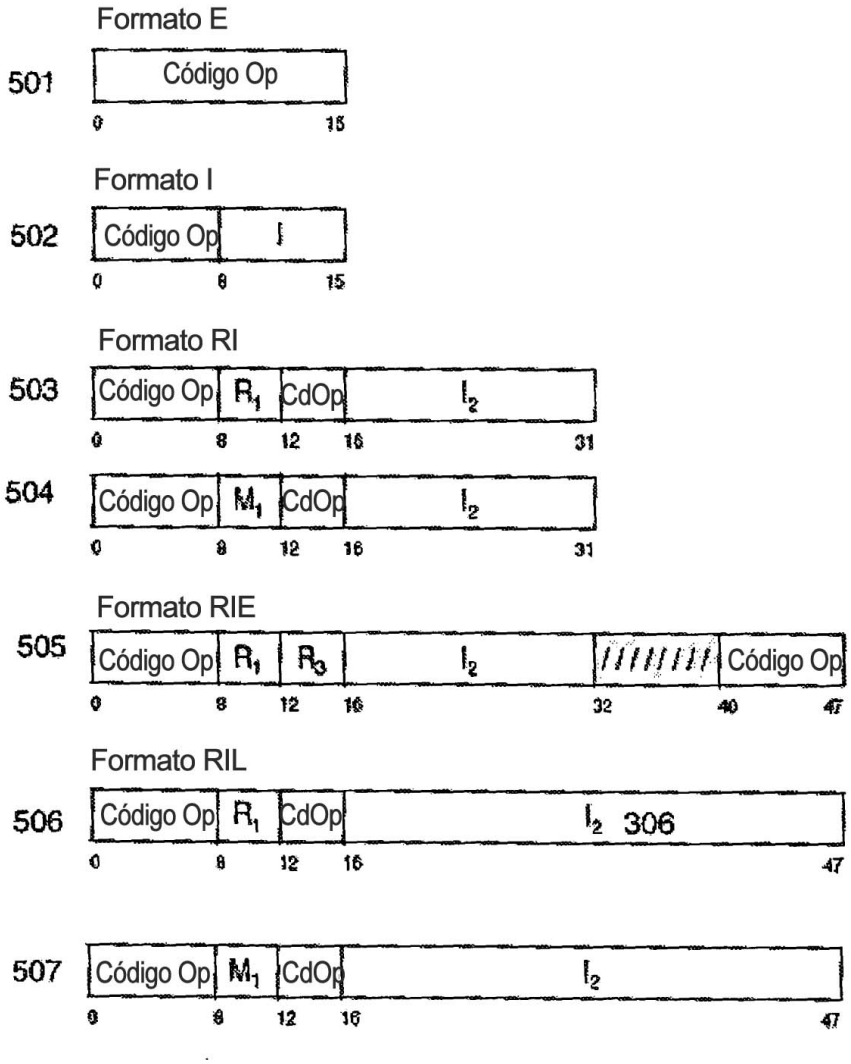
Formato E
501
Código Op
0
15
Formato I
502
Código Op
I
0
8
15
Formato RI
503
Código Op
R1
CdOp
I2
0
8
12
16
31
504
Código Op
M1
CdOp
I2
0
8
12
16
31
Formato RIE
505
Código Op
R1
R3
I2
Código Op
0
8
12
16
32
40
47
Formato RIL
506
Código Op
R1
CdOp
I2 306
0
8
12
16
47
507
Código Op
M1
CdOp
I2
0
8
12
16
47

FIG. 5A

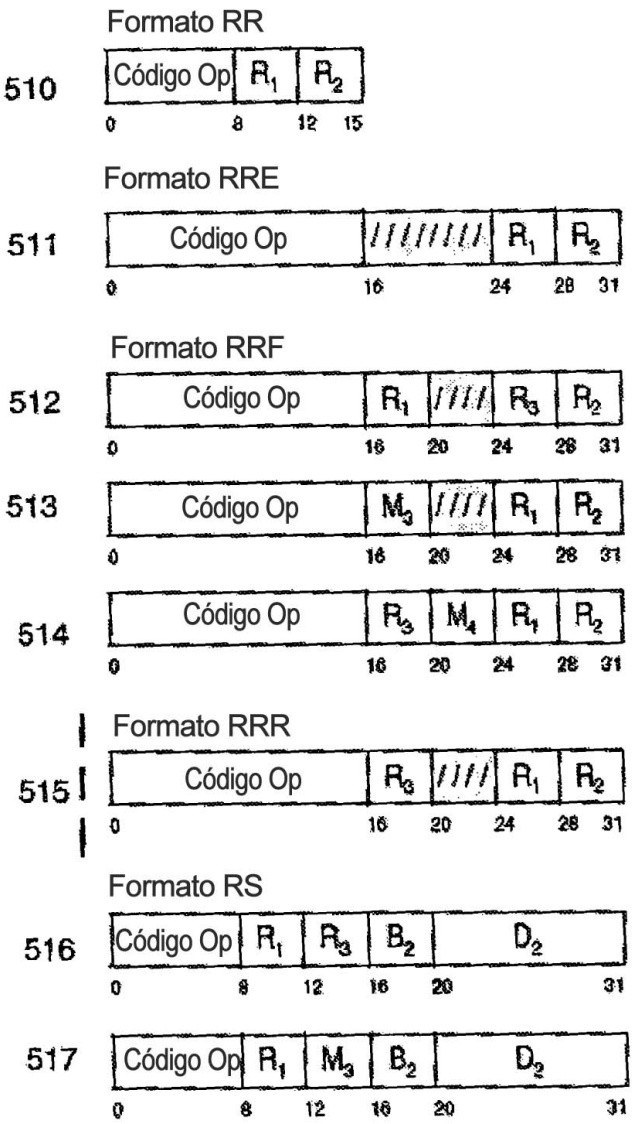
Formato RR
510
Código Op
R1
R2
0
8
12
15
Formato RRE
511
Código Op
R1
R2
0
16
24
28
31
Formato RRF
512
Código Op
R1
R3
R2
0
16
20
24
28
31
513
Código Op
M3
R1
R2
0
16
20
24
28
31
514
Código Op
R3
M4
R1
R2
0
16
20
24
28
31
Formato RRR
515
Código Op
R3
R1
R2
0
16
20
24
28
31
Formato RS
516
Código Op
R1
R3
B2
D2
0
8
12
16
20
31
517
Código Op
R1
M3
B2
D2
0
8
12
16
20
31

FIG. 5B

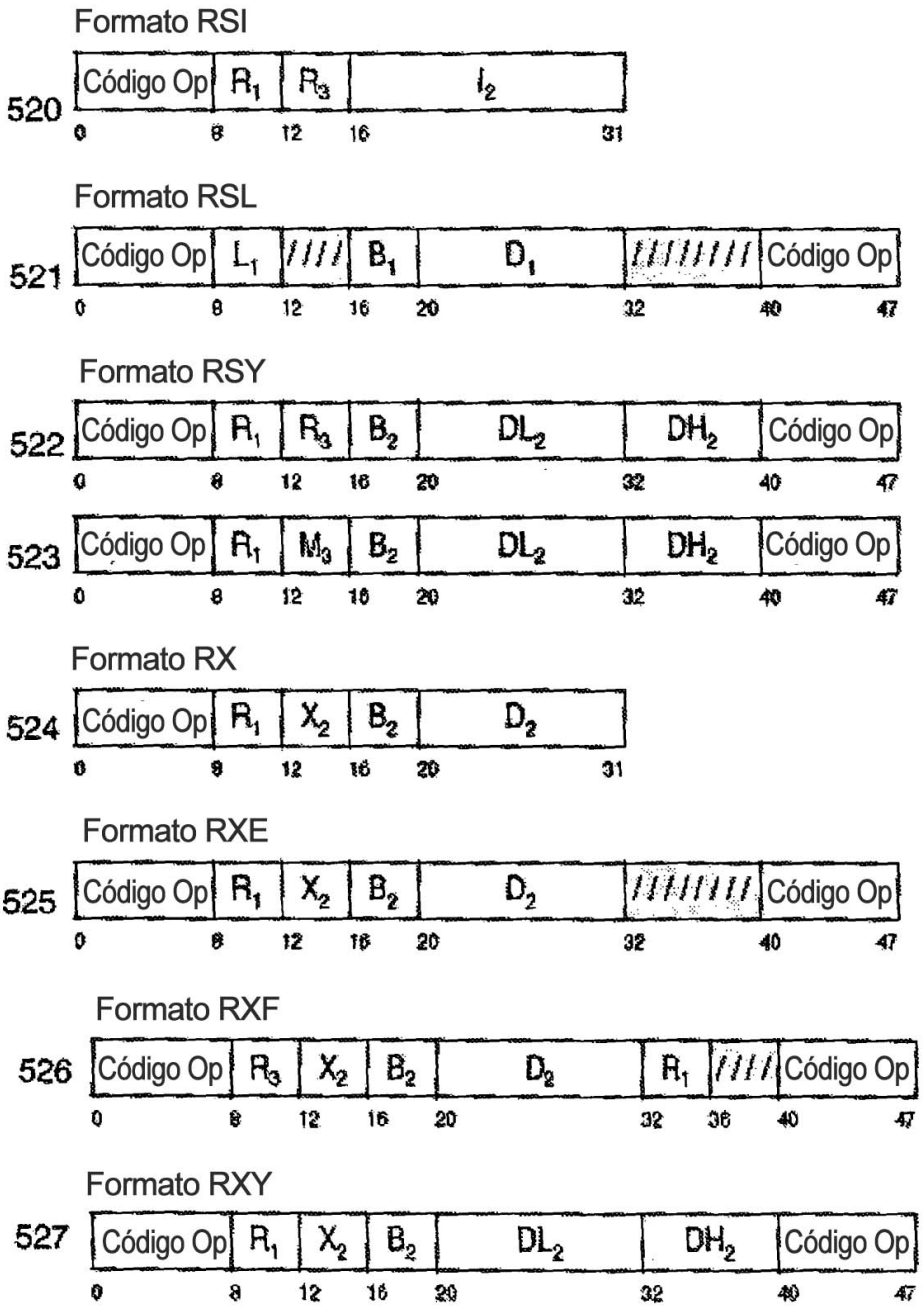
Formato RSI
520
Código Op
R1
R3
I2
0
8
12
16
31
Formato RSL
521
Código Op
L1
////
B1
D1
////////
Código Op
0
8
12
16
20
32
40
47
Formato RSY
522
Código Op
R1
R3
B2
DL2
DH2
Código Op
523
Código Op
R1
M3
B2
DL2
DH2
Código Op
Formato RX
524
Código Op
R1
X2
B2
D2
Formato RXE
525
Código Op
R1
X2
B2
D2
////////
Código Op
Formato RXF
526
Código Op
R3
X2
B2
D2
R1
////
Código Op
36
Formato RXY
527
Código Op
R1
X2
B2
DL2
DH2
Código Op

FIG. 5C

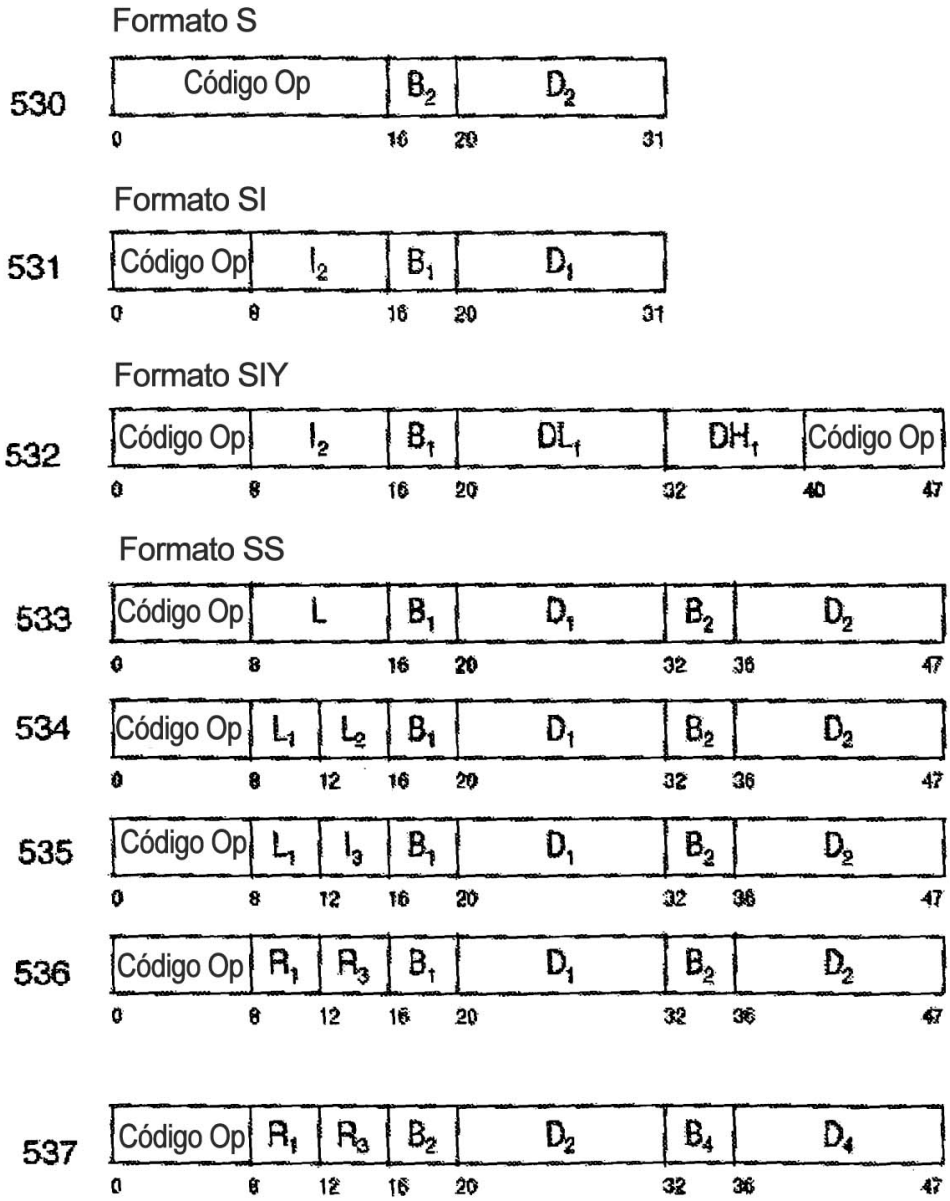
Formato S
530
Código Op
B2
D2
0
16
20
31
Formato SI
531
Código Op
I2
B1
D1
0
8
16
20
31
Formato SIY
532
Código Op
I2
B1
DL1
DH1
Código Op
0
8
16
20
32
40
47
Formato SS
533
Código Op
L
B1
D1
B2
D2
0
8
16
20
32
36
47
534
Código Op
L1
L2
B1
D1
B2
D2
0
8
12
16
20
32
36
47
535
Código Op
L1
I3
B1
D1
B2
D2
0
8
12
16
20
32
36
47
536
Código Op
R1
R3
B1
D1
B2
D2
0
8
12
16
20
32
36
47
537
Código Op
R1
R3
B2
D2
B4
D4
0
8
12
16
20
32
36
47

FIG. 5D

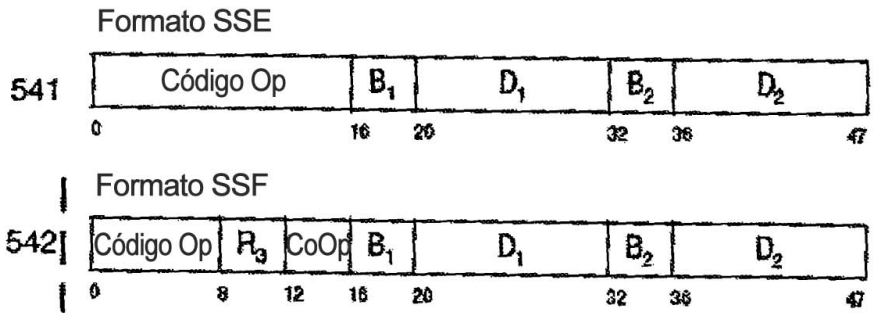
Formato SSE
541
Código Op
B1
D1
B2
D2
0
16
20
32
36
47
Formato SSF
542
Código Op
R3
CoOp
B1
D1
B2
D2
0
8
12
16
20
32
36
47

FIG. 5E

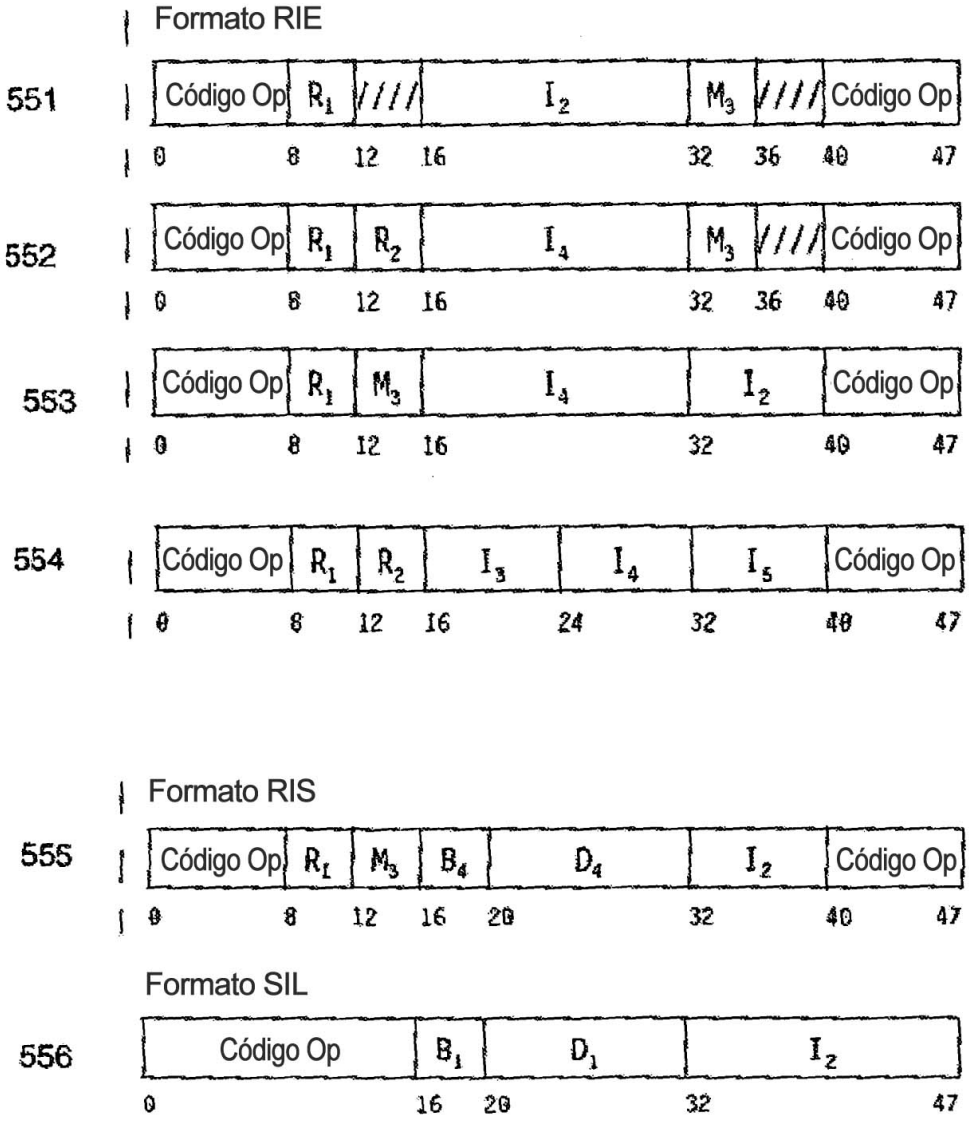
Formato RIE
551
Código Op
R1
////
I2
M3
////
Código Op
0
8
12
16
32
36
40
47
552
Código Op
R1
R2
I4
M3
////
Código Op
0
8
12
16
32
36
40
47
553
Código Op
R1
M3
I4
I2
Código Op
0
8
12
16
32
40
47
554
Código Op
R1
R2
I3
I4
I5
Código Op
0
8
12
16
24
32
40
47
Formato RIS
555
Código Op
R1
M3
B4
D4
I2
Código Op
0
8
12
16
20
32
40
47
Formato SIL
556
Código Op
B1
D1
I2
0
16
20
32
47

FIG. 5F

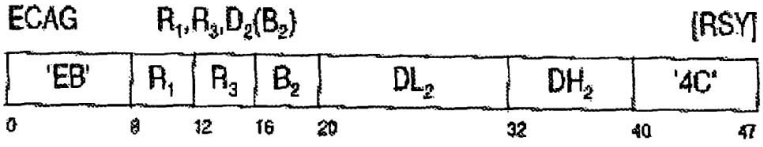
ECAG
R1,R3,D2(B2)
[RSY]
'EB'
R1
R3
B2
DL2
DH2
'4C'
0
8
12
16
20
32
40
47

FIG. 6

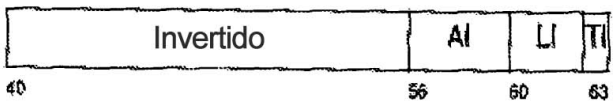
Invertido
AI
LI
TI
40
56
60
63

FIG. 7

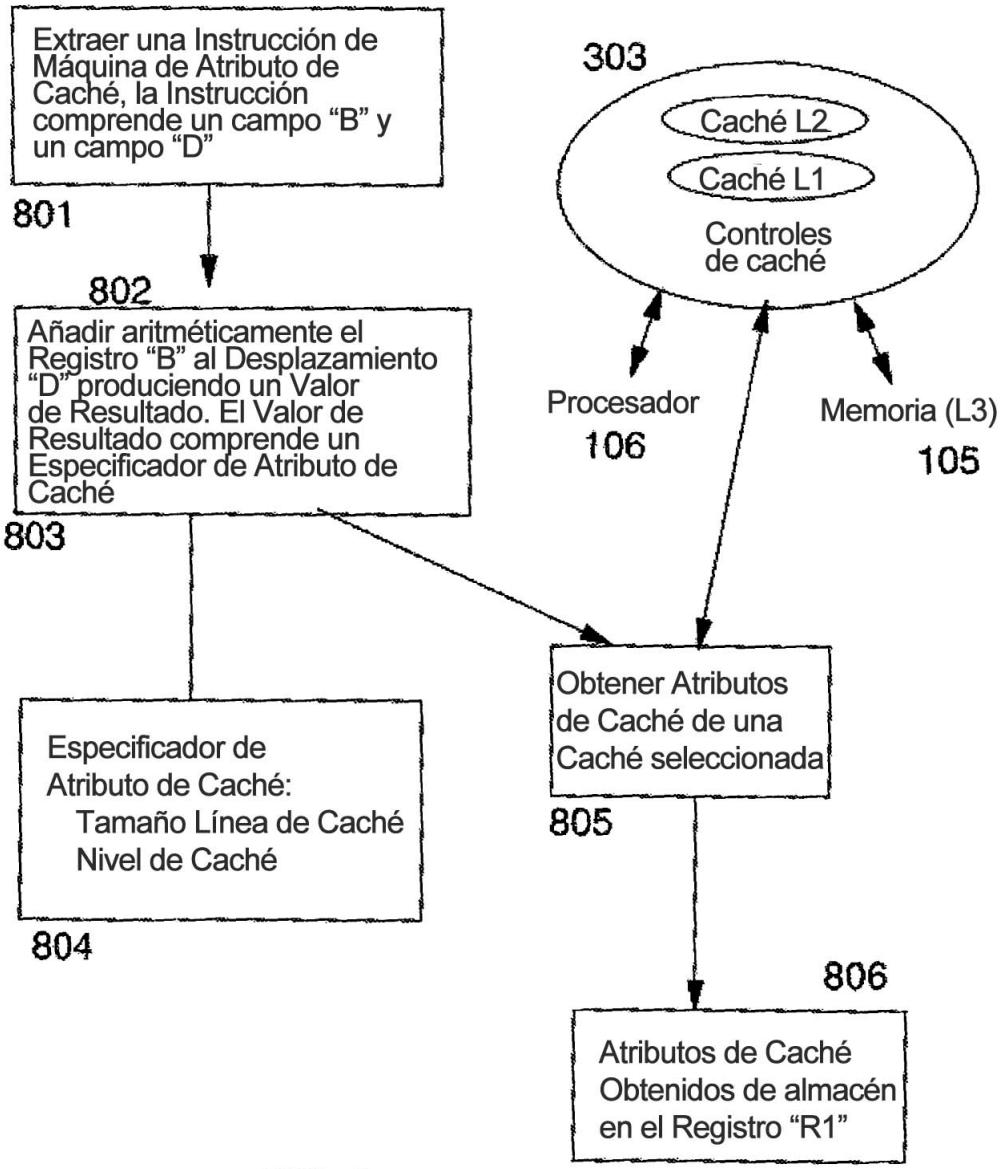
Extraer una Instrucción de Máquina de Atributo de Caché, la Instrucción comprende un campo “B” y un campo “D”
801
802
Añadir aritméticamente el Registro “B” al Desplazamiento “D” produciendo un Valor de Resultado. El Valor de Resultado comprende un Especificador de Atributo de Caché
803
Especificador de Atributo de Caché:
Tamaño Línea de Caché
Nivel de Caché
804
303
Caché L2
Caché L1
Controles de caché
Procesador
106
Memoria (L3)
105
Obtener Atributos de Caché de una Caché seleccionada
805
806
Atributos de Caché Obtenidos de almacén en el Registro “R1”

FIG. 8