(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 906 398**

(51) Int. CI.:

G06F 9/38 (2008.01)
G06T 1/20 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **14.03.2018 E 19166050 (5)**

(97) Fecha y número de publicación de la concesión europea: **01.12.2021 EP 3543845**

(54) Título: **Procesador de múltiples núcleos, presentando cada núcleo una ruta de datos de tipo coma flotante y una ruta de datos de tipo entero independientes**

(30) Prioridad:

24.04.2017 US 201715494773

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
18.04.2022

(73) Titular/es:

INTEL CORPORATION (100.0%)
2200 Mission College Boulevard
Santa Clara, CA 95054, US

(72) Inventor/es:

OULD-AHMED-VALL, ELMOUSTAPHA;
LAKSHMANAN, BARATH;
SHPEISMAN, TATIANA;
RAY, JOYDEEP;
TANG, PING T.;
STRICKLAND, MICHAEL;
CHEN, XIAOMING;
YAO, ANBANG;
ASHBAUGH, BEN J.;
HURD, LINDA L. y
MA, LIWEI

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín Europeo de Patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre Concesión de Patentes Europeas).

DESCRIPCIÓN

Procesador de múltiples núcleos, presentando cada núcleo una ruta de datos de tipo coma flotante y una ruta de datos de tipo entero independientes

CAMPO

Las formas de realización se refieren, en general, a un procesamiento de datos y, más en particular, a un procesamiento de datos a través de una unidad de procesamiento de gráficos de propósito general.

ANTECEDENTES DE LA DESCRIPCIÓN

El procesamiento en paralelo de datos de gráficos actual incluye sistemas y procedimientos desarrollados para realizar operaciones específicas en datos de gráficos tales como, por ejemplo, interpolación lineal, teselación, rasterización, correlación de texturas, pruebas de profundidad, etc. Tradicionalmente, los procesadores de gráficos utilizaban unidades computacionales de función fija para procesar datos de gráficos; sin embargo, más recientemente, partes de los procesadores de gráficos se han hecho programables, lo que permite que dichos procesadores admitan una mayor variedad de operaciones para procesar datos de vértices y fragmentos.

Para aumentar aún más el rendimiento, los procesadores de gráficos típicamente implementan técnicas de procesamiento, tales como un procesamiento en cadena, que intentan procesar, en paralelo, la mayor cantidad de datos de gráficos posible a lo largo de las diferentes partes del procesamiento en cadena de gráficos. Los procesadores de gráficos paralelos con arquitecturas de una sola instrucción y múltiples subprocesos (SIMT) están diseñados para maximizar la cantidad de procesamiento paralelo en la tubería de gráficos. En una arquitectura SIMT, los grupos de subprocesos paralelos intentan ejecutar instrucciones de programa de forma sincronizada con la mayor frecuencia posible para aumentar la eficiencia de procesamiento. Una visión general del software y el hardware para las arquitecturas SIMT se puede encontrar en el documento de Shane Cook, "*CUDA Programming*", Capítulo 3, páginas 37-51 (2013) y/o en el documento de Nicholas Wilt, "*CUDA Handbook, A Comprehensive Guide to GPU Programming*", secciones 2.6.2 a 3.1.2 (junio de 2013).

El documento US 2014/189329A1 se refiere a un conmutador de contexto de granularidad de un conjunto de subprocesos cooperativos durante el manejo de interrupciones en arquitecturas de procesadores paralelos. Por ejemplo, se maneja una interrupción encontrada en un subproceso que es parte de un conjunto de subprocesos que se está ejecutando en una pluralidad de unidades de ejecución. Se actualiza una estructura de datos con un identificador asociado al subproceso para indicar que la interrupción se produjo durante la ejecución del conjunto de subprocesos. Las unidades de ejecución ejecutan una rutina de manejo de interrupciones que incluye un conmutador de contexto. Las unidades de ejecución llevan a cabo este conmutador de contexto para al menos una de las unidades de ejecución como parte de la rutina de manejo de interrupciones, al tiempo que se permite que las unidades de ejecución restantes salgan de la rutina de manejo de interrupciones antes del conmutador de contexto.

El documento WO 2017/049592 A1 se refiere a técnicas para mejorar la eficacia de la memoria compartida. Por ejemplo, en procedimientos y aparatos para mejorar la eficacia de la memoria compartida, se compila una primera versión de un código para acceder a uno o más registros como memoria local compartida, y también se compila una segunda versión del mismo código para acceder a una memoria caché como memoria local compartida. La primera versión del código se ejecuta en respuesta a la comparación de un tamaño de grupo de trabajo del código con un valor umbral.

RESUMEN

La presente invención se define en la reivindicación independiente 1. Las reivindicaciones dependientes definen formas de realización de la misma. De considerarse que cualquier "forma de realización" o "ejemplo" que se divulguen en la siguiente descripción pero que no estén cubiertos por las reivindicaciones solo se presentan con fines ilustrativos.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

Para que las características de la presente invención se puedan entender en detalle, se ofrece una descripción más particular de la invención mediante referencia a formas de realización, algunas de las cuales se ilustran en los dibujos adjuntos. Sin embargo, cabe señalar que los dibujos adjuntos solo ilustran formas de realización típicas y, por lo tanto, no debe considerarse que limitan el alcance de todas las formas de realización.

La FIG. 1 es un diagrama de bloques que ilustra un sistema informático configurado para implementar uno o más aspectos de las formas de realización descritas en el presente documento.

Las FIGS. 2A-2D ilustran componentes de procesadores paralelos, de acuerdo con una forma de realización reivindicada de la invención.

Las FIGS. 3A-3B son diagramas de bloques de multiprocesadores de gráficos, de acuerdo con formas de realización.

Las FIGS. 4A-4F ilustran una arquitectura ejemplar en la que una pluralidad de GPU están acopladas de manera comunicativa a una pluralidad de procesadores de múltiples núcleos.

La FIG. 5 ilustra un procesamiento en cadena de gráficos, de acuerdo con una forma de realización.

La FIG. 6 ilustra una pila de software de aprendizaje automático, de acuerdo con una forma de realización.

La FIG. 7 ilustra una unidad de procesamiento de gráficos de propósito general altamente paralela, de acuerdo con una forma de realización.

La FIG. 8 ilustra un sistema informático de múltiples GPU, de acuerdo con una forma de realización.

Las FIGS. 9A-9B ilustran capas de redes neuronales profundas ejemplares.

La FIG. 10 ilustra una red neuronal recurrente ejemplar.

La FIG. 11 ilustra el entrenamiento y el despliegue de una red neuronal profunda.

La FIG. 12 es un diagrama de bloques que ilustra un aprendizaje distribuido.

La FIG. 13 ilustra un sistema en chip (SOC) de inferencia ejemplar, adecuado para generar inferencias utilizando un modelo entrenado.

La FIG. 14 es un diagrama de bloques de una unidad multiprocesador, de acuerdo con una forma de realización reivindicada de la invención.

La FIG. 15 ilustra un sistema de procesamiento de precisión mixta, de acuerdo con una forma de realización.

La FIG. 16 ilustra un sistema de procesamiento de precisión mixta adicional, de acuerdo con una forma de realización.

La FIG. 17 es un diagrama de flujo de lógica operativa para un sistema de procesamiento de precisión mixta, de acuerdo con una forma de realización.

La FIG. 18 es un diagrama de flujo de lógica operativa para otro sistema de procesamiento de precisión mixta, de acuerdo con una forma de realización.

La FIG. 19 ilustra un sistema de aprendizaje automático, de acuerdo con una forma de realización.

La FIG. 20 ilustra operaciones lógicas de un sistema de aprendizaje automático, de acuerdo con una forma de realización.

La FIG. 21 es un diagrama de bloques de un sistema de procesamiento, de acuerdo con una forma de realización.

La FIG. 22 es un diagrama de bloques de un procesador de acuerdo con una forma de realización.

La FIG. 23 es un diagrama de bloques de un procesador de gráficos, de acuerdo con una forma de realización.

La FIG. 24 es un diagrama de bloques de un motor de procesamiento de gráficos de un procesador de gráficos de acuerdo con algunas formas de realización.

La FIG. 25 es un diagrama de bloques de un procesador de gráficos proporcionado por una forma de realización adicional.

La FIG. 26 ilustra una lógica de ejecución de subprocesos que incluye una formación de elementos de procesamiento empleados en algunas formas de realización.

La FIG. 27 es un diagrama de bloques que ilustra formatos de instrucciones de procesador de gráficos de acuerdo con algunas formas de realización.

La FIG. 28 es un diagrama de bloques de un procesador de gráficos de acuerdo con otra forma de realización.

Las FIGS. 29A-29B ilustran un formato de comando de procesador de gráficos y una secuencia de comandos, de acuerdo con algunas formas de realización.

La FIG. 30 ilustra una arquitectura de software de gráficos ejemplar para un sistema de procesamiento de datos de acuerdo con algunas formas de realización.

La FIG. 31 es un diagrama de bloques que ilustra un sistema de desarrollo de núcleos IP, de acuerdo con una forma de realización.

La FIG. 32 es un diagrama de bloques que ilustra un circuito integrado de sistema en chip ejemplar, de acuerdo con una forma de realización.

La FIG. 33 es un diagrama de bloques que ilustra un procesador de gráficos adicional, de acuerdo con una forma de realización.

La FIG. 34 es un diagrama de bloques que ilustra un procesador de gráficos ejemplar adicional de un circuito integrado de sistema en chip, de acuerdo con una forma de realización.

DESCRIPCIÓN DETALLADA

En algunas formas de realización, una unidad de procesamiento de gráficos (GPU) está acoplada de manera comunicativa a núcleos de ordenador principal/procesador para acelerar las operaciones de gráficos, las operaciones de aprendizaje automático, las operaciones de análisis de patrones y diversas funciones de GPU de propósito general (GPGPU). La GPU puede estar acoplada de manera comunicativa al procesador/núcleos de ordenador principal a través de un bus u otra interconexión (por ejemplo, una interconexión de alta velocidad, tal como PCIe o NVLink). En otras formas de realización, la GPU puede estar integrada en el mismo encapsulado o chip que los núcleos y estar acoplada de manera comunicativa a los núcleos a través de un bus/interconexión de procesador internos (es decir, internos al encapsulado o chip). Independientemente de la manera en que se conecte la GPU, los núcleos de procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidos en un descriptor de trabajo. A continuación, la GPU utiliza circuitos/lógica dedicados para procesar eficazmente estos comandos/instrucciones.

En la siguiente descripción se exponen numerosos detalles específicos para proporcionar un entendimiento más exhaustivo. Sin embargo, para un experto en la técnica será evidente que las formas de realización descritas en el presente documento se pueden llevar a la práctica sin uno o más de estos detalles específicos. En otros casos no se han descrito características ampliamente conocidas para no complicar los detalles de las presentes formas de realización.

Visión general del sistema

La FIG. 1 es un diagrama de bloques que ilustra un sistema informático 100 configurado para implementar uno o más aspectos de las formas de realización descritas en el presente documento. El sistema informático 100 incluye un subsistema de procesamiento 101 que tiene uno o más procesadores 102 y una memoria de sistema 104 que se comunican a través de una ruta de interconexión que puede incluir un concentrador de memoria 105. El concentrador de memoria 105 puede ser otro componente dentro de un componente de conjunto de chips o puede estar integrado dentro del uno o más procesadores 102. El concentrador de memoria 105 está acoplado a un subsistema de E/S 111 a través de un enlace de comunicación 106. El subsistema de E/S 111 incluye un concentrador de E/S 107 que puede permitir que el sistema informático 100 reciba datos de entrada desde uno o más dispositivos de entrada 108. Además, el concentrador de E/S 107 puede permitir que un controlador de visualización, que puede estar incluido en el uno o más procesadores 102, proporcione datos de salida a uno o más dispositivos de visualización 110A. En una forma de realización, el uno o más dispositivos de visualización 110A acoplados al concentrador de E/S 107 pueden incluir un dispositivo de visualización local, interno o integrado.

En una forma de realización, el subsistema de procesamiento 101 incluye uno o más procesadores paralelos 112 acoplados al concentrador de memoria 105 a través de un bus u otro enlace de comunicación 113. El enlace de comunicación 113 puede ser uno de cualquier variedad de tecnologías o protocolos de enlace de comunicación basados en normas, tal como, pero sin limitarse a, PCI Express, o puede ser una interfaz de comunicaciones específica del proveedor o una estructura de comunicaciones. En una forma de realización, el uno o más procesadores paralelos 112 forman un sistema de procesamiento vectorial o en paralelo centrado en la computación que incluye un gran número de núcleos de procesamiento y/o agrupaciones (*clusters*) de procesamiento, tales como un procesador de muchos núcleos integrados (MIC). En una forma de realización, el uno o más procesadores paralelos 112 forman un subsistema de procesamiento de gráficos que puede proporcionar píxeles a un dispositivo de los uno o más dispositivos de visualización 110A acoplados a través del concentrador de E/S 107. El uno o más procesadores paralelos 112 también pueden incluir un controlador de visualización y una interfaz de visualización (no mostrados) para permitir una conexión directa a uno o más dispositivos de visualización 110B.

Dentro del subsistema de E/S 111, una unidad de almacenamiento de sistema 114 puede conectarse al concentrador de E/S 107 para proporcionar un mecanismo de almacenamiento para el sistema informático 100. Se puede utilizar un conmutador de E/S 116 para proporcionar un mecanismo de interfaz para permitir las conexiones entre el

concentrador de E/S 107 y otros componentes, tales como un adaptador de red 118 y/o un adaptador de red inalámbrica 119 que pueden estar integrados en la plataforma, y otros diversos dispositivos que se pueden añadir a través de uno o más dispositivos complementarios 120. El adaptador de red 118 puede ser un adaptador Ethernet u otro adaptador de red alámbrica. El adaptador de red inalámbrica 119 puede incluir una o más de una comunicación Wi-Fi, Bluetooth, de campo cercano (NFC) u otro dispositivo de red que incluya una o más radios inalámbricas.

El sistema informático 100 puede incluir otros componentes que no se muestran explícitamente, que incluyen USB u otras conexiones de puerto, unidades de almacenamiento óptico, dispositivos de captura de vídeo y similares, que también se pueden conectar al concentrador de E/S 107. Las rutas de comunicación que interconectan los diversos componentes en la FIG. 1 se puede implementar utilizando cualquier protocolo adecuado, tal como protocolos basados en PCI (interconexión de componentes periféricos) (por ejemplo, PCI-Express), o cualquier otro bus o interfaz de comunicación de punto a punto y/o protocolo, tal como la interconexión de alta velocidad NV-Link, o protocolos de interconexión conocidos en la técnica.

En una forma de realización, el uno o más procesadores paralelos 112 incorporan circuitos optimizados para el procesamiento de gráficos y vídeo, incluidos, por ejemplo, circuitos de salida de vídeo, y constituyen una unidad de procesamiento de gráficos (GPU). En otra forma de realización, el uno o más procesadores paralelos 112 incorporan circuitos optimizados para un procesamiento de propósito general, al tiempo que preservan la arquitectura computacional subyacente, descrita en mayor detalle en el presente documento. Aún en otra forma de realización, los componentes del sistema informático 100 pueden integrarse con otro u otros elementos de sistema en un único circuito integrado. Por ejemplo, el uno o más procesadores paralelos 112, el concentrador de memoria 105, el/los procesador(es) 102 y el concentrador de E/S 107 se pueden integrar en un circuito integrado de sistema en chip (SoC). De manera alternativa, los componentes del sistema informático 100 se pueden integrar en un solo encapsulado para formar una configuración de sistema en encapsulado (SIP). En una forma de realización, al menos una parte de los componentes del sistema informático 100 puede integrarse en un módulo de múltiples chips (MCM), que puede interconectarse con otros módulos de múltiples chips en un sistema informático modular.

Se apreciará que el sistema informático 100 mostrado en el presente documento es ilustrativo y que pueden realizarse variaciones y modificaciones. La topología de conexión, incluidos el número y la disposición de puentes, el número de procesadores 102 y el número de procesadores paralelos 112, puede modificarse según se desee. Por ejemplo, en algunas formas de realización, la memoria de sistema 104 está conectada al o a los procesadores 102 directamente, en lugar de a través de un puente, mientras que otros dispositivos se comunican con la memoria de sistema 104 a través del concentrador de memoria 105 y el o los procesadores 102. En otras topologías alternativas, el/los procesador(es) paralelo(s) 112 está(n) conectado(s) al concentrador de E/S 107 o directamente a un procesador de los uno o más procesadores 102, en lugar de al concentrador de memoria 105. En otras formas de realización, el concentrador de E/S 107 y el concentrador de memoria 105 pueden estar integrados en un solo chip. Algunas formas de realización pueden incluir dos o más conjuntos de procesadores 102 unidos a través de múltiples zócalos, que pueden acoplarse a dos o más instancias del/de los procesador(es) paralelo(s) 112.

Algunos de los componentes particulares que se muestran en el presente documento son opcionales y pueden no incluirse en todas las implementaciones del sistema informático 100. Por ejemplo, se puede admitir cualquier cantidad de tarjetas complementarias o periféricos, o se pueden eliminar algunos componentes. Además, algunas arquitecturas pueden utilizar terminología diferente para componentes similares a los ilustrados en la FIG. 1. Por ejemplo, el concentrador de memoria 105 puede denominarse "puente norte" en algunas arquitecturas, mientras que el concentrador de E/S 107 puede denominarse "puente sur".

La FIG. 2A ilustra un procesador paralelo 200, de acuerdo con una forma de realización. Los diversos componentes del procesador paralelo 200 pueden implementarse utilizando uno o más dispositivos de circuito integrado, tales como procesadores programables, circuitos integrados específicos de la aplicación (ASIC) o matrices de puertas programables *in situ* (FPGA). El procesador paralelo 200 ilustrado es una variante del uno o más procesadores paralelos 112 mostrados en la FIG. 1, de acuerdo con una forma de realización.

En una forma de realización, el procesador paralelo 200 incluye una unidad de procesamiento paralelo 202. La unidad de procesamiento paralelo incluye una unidad de E/S 204 que permite la comunicación con otros dispositivos, incluidas otras instancias de la unidad de procesamiento paralelo 202. La unidad de E/S 204 puede estar conectada directamente a otros dispositivos. En una forma de realización, la unidad de E/S 204 se conecta a otros dispositivos a través del uso de un concentrador o interfaz de conmutación, tal como el concentrador de memoria 105. Las conexiones entre el concentrador de memoria 105 y la unidad de E/S 204 forman un enlace de comunicación 113. Dentro de la unidad de procesamiento paralelo 202, la unidad de E/S 204 está conectada a una interfaz de ordenador principal 206 y a una barra cruzada de memoria 216, donde la interfaz de ordenador principal 206 recibe comandos dirigidos a realizar operaciones de procesamiento y la barra cruzada de memoria 216 recibe comandos dirigidos a realizar operaciones de memoria.

Cuando la interfaz de ordenador principal 206 recibe un *búfer* de comandos a través de la unidad de E/S 204, la interfaz de ordenador principal 206 puede enviar a una sección de entrada 208 tareas para realizar esos comandos. En una forma de realización, la sección de entrada 208 está acoplada a un planificador 210, que está configurado para

distribuir comandos u otros elementos de trabajo a una formación de agrupaciones de procesamiento 212. En una forma de realización, el planificador 210 garantiza que la formación de agrupaciones de procesamiento 212 esté configurado adecuadamente y en un estado válido antes de que las tareas se distribuyan a las agrupaciones de procesamiento de la formación de agrupaciones de procesamiento 212. En una forma de realización, el planificador 210 se implementa a través de lógica de firmware que se ejecuta en un microcontrolador. El planificador implementado por microcontrolador 210 puede configurarse para realizar operaciones complejas de planificación y distribución de trabajo con granularidad gruesa y fina, lo que permite una preparación y cambio de contexto rápidos de los subprocesos que se ejecutan en la formación de procesamiento 212. En una forma de realización, el software de ordenador principal puede probar cargas de trabajo para su planificación en la formación de procesamiento 212 a través de uno de múltiples timbres de procesamiento de gráficos. Las cargas de trabajo se pueden distribuir automáticamente a través de la formación de procesamiento 212 mediante la lógica del planificador 210 dentro del microcontrolador de planificador.

La formación de agrupaciones de procesamiento 212 puede incluir hasta "N" agrupaciones de procesamiento (por ejemplo, agrupación 214A, agrupación 214B, hasta la agrupación 214N). Cada agrupación 214A-214N de la formación de agrupaciones de procesamiento 212 puede ejecutar una gran cantidad de subprocesos concurrentes. El planificador 210 puede asignar trabajo a las agrupaciones 214A-214N de la formación de agrupaciones de procesamiento 212 utilizando diversos algoritmos de planificación y/o distribución de trabajo, que pueden variar dependiendo de la carga de trabajo que surja para cada tipo de programa o cálculo. La planificación puede manejarse dinámicamente por el planificador 210, o puede ser asistida en parte por lógica de compilador durante la compilación de la lógica de programa configurada para su ejecución por la formación de agrupaciones de procesamiento 212. En una forma de realización, diferentes agrupaciones 214A-214N de la formación de agrupaciones de procesamiento 212 se pueden asignar para procesar diferentes tipos de programas o para realizar diferentes tipos de cálculos.

La formación de agrupaciones de procesamiento 212 puede configurarse para realizar diversos tipos de operaciones de procesamiento paralelo. En una forma de realización, la formación de agrupaciones de procesamiento 212 está configurada para realizar operaciones de cálculo en paralelo de propósito general. Por ejemplo, la formación de agrupaciones de procesamiento 212 puede incluir lógica para ejecutar tareas de procesamiento, que incluyen filtrar datos de vídeo y/o audio, realizar operaciones de modelado, incluidas operaciones físicas, y realizar transformaciones de datos.

En una forma de realización, la formación de agrupaciones de procesamiento 212 está configurada para realizar operaciones de procesamiento de gráficos en paralelo. En formas de realización en las que el procesador paralelo 200 está configurado para realizar operaciones de procesamiento de gráficos, la formación de agrupaciones de procesamiento 212 puede incluir lógica adicional para respaldar la ejecución de dichas operaciones de procesamiento de gráficos, que incluyen, pero sin limitarse a, lógica de muestreo de texturas para realizar operaciones de texturas, así como lógica de teselación y otra lógica de procesamiento de vértices. Además, la formación de agrupaciones de procesamiento 212 puede configurarse para ejecutar programas de sombreado relacionados con el procesamiento de gráficos, tales como, pero sin limitarse a, sombreadores de vértices, sombreadores de teselación, sombreadores de geometría y sombreadores de píxeles. La unidad de procesamiento paralelo 202 puede transferir datos de la memoria de sistema a través de la unidad de E/S 204 para su procesamiento. Durante el procesamiento, los datos transferidos se pueden almacenar en memoria en chip (por ejemplo, memoria de procesador paralelo 222) durante el procesamiento, y después se pueden volver a escribir en la memoria de sistema.

En una forma de realización, cuando la unidad de procesamiento paralelo 202 se utiliza para realizar el procesamiento de gráficos, el planificador 210 puede estar configurado para dividir la carga de trabajo de procesamiento en tareas de aproximadamente el mismo tamaño, para permitir una mejor distribución de las operaciones de procesamiento de gráficos a múltiples agrupaciones 214A-214N de la formación de agrupaciones de procesamiento 212. En algunas formas de realización, partes de la formación de agrupaciones de procesamiento 212 pueden estar configuradas para realizar diferentes tipos de procesamiento. Por ejemplo, una primera parte puede estar configurada para realizar sombreado de vértices y generación de topología, una segunda parte puede estar configurada para realizar sombreado de teselación y geometría, y una tercera parte puede estar configurada para realizar sombreado de píxeles u otras operaciones de espacio de pantalla, para producir una imagen renderizada para su visualización. Los datos intermedios producidos por una o más de las agrupaciones 214A-214N se pueden almacenar en *búferes* para permitir que los datos intermedios se transmitan entre las agrupaciones 214A-214N para su procesamiento adicional.

Durante el funcionamiento, la formación de agrupaciones de procesamiento 212 puede recibir tareas de procesamiento que se ejecutarán a través del planificador 210, que recibe comandos que definen tareas de procesamiento desde la sección de entrada 208. En cuanto a las operaciones de procesamiento de gráficos, las tareas de procesamiento pueden incluir índices de datos a procesar, por ejemplo, datos de superficie (parche), datos de primitivas, datos de vértices y/o datos de píxeles, así como parámetros de estado y comandos que definen cómo se deben procesar los datos (por ejemplo, qué programa se debe ejecutar). El planificador 210 puede estar configurado para obtener los índices correspondientes a las tareas o puede recibir los índices desde la sección de entrada 208. La sección de entrada 208 puede estar configurada para garantizar que la formación de agrupaciones de procesamiento 212 esté configurada en un estado válido antes de que se inicie la carga de trabajo especificada por *búferes* de comandos entrantes (por ejemplo, *búferes* de lotes, *búferes* de carga, etc.).

Cada instancia de las una o más instancias de la unidad de procesamiento paralelo 202 puede acoplarse a la memoria de procesador paralelo 222. Se puede acceder a la memoria de procesador paralelo 222 a través de la barra cruzada de memoria 216, que puede recibir solicitudes de memoria desde la formación de agrupaciones de procesamiento 212, así como de la unidad de E/S 204. La barra cruzada de memoria 216 puede acceder a la memoria de procesador paralelo 222 a través de una interfaz de memoria 218. La interfaz de memoria 218 puede incluir múltiples unidades de partición (por ejemplo, unidad de partición 220A, unidad de partición 220B, hasta la unidad de partición 220N), donde cada una puede acoplarse a una parte (por ejemplo, unidad de memoria) de la memoria de procesador paralelo 222. En una implementación, el número de unidades de partición 220A-220N está configurado para ser igual al número de unidades de memoria, de modo que una primera unidad de partición 220A tiene una primera unidad de memoria 224A correspondiente, una segunda unidad de partición 220B tiene una unidad de memoria 224B correspondiente, y una enésima unidad de partición 220N tiene una enésima unidad de memoria 224N correspondiente. En otras formas de realización, el número de unidades de partición 220A-220N puede no ser igual al número de dispositivos de memoria.

En varias formas de realización, las unidades de memoria 224A-224N pueden incluir varios tipos de dispositivos de memoria, que incluyen memoria de acceso aleatorio dinámica (DRAM) o memoria de acceso aleatorio de gráficos, tal como memoria de acceso aleatorio de gráficos síncrona (SGRAM), que incluye memoria de doble velocidad de datos de gráficos (GDDR). En una forma de realización, las unidades de memoria 224A-224N también pueden incluir memoria apilada 3D que incluye, pero no se limita a, una memoria de alto ancho de banda (HBM). Los expertos en la técnica apreciarán que la implementación específica de las unidades de memoria 224A-224N puede variar y que puede seleccionarse de entre uno de varios diseños convencionales. Objetivos de renderizado, tales como *búferes* de fotogramas o correlaciones de textura, pueden almacenarse a través de las unidades de memoria 224A-224N, lo que permite que las unidades de partición 220A-220N escriban partes de cada objetivo de renderizado en paralelo para utilizar de manera eficaz el ancho de banda disponible de la memoria de procesador paralelo 222. En algunas formas de realización, una instancia local de la memoria de procesador paralelo 222 puede excluirse en favor de un diseño de memoria unificado que utilice memoria de sistema junto con memoria caché local.

En una forma de realización, una cualquiera de las agrupaciones 214A-214N de la formación de agrupaciones de procesamiento 212 puede procesar datos que se escribirán en cualquiera de las unidades de memoria 224A-224N dentro de la memoria de procesador paralelo 222. La barra cruzada de memoria 216 puede estar configurada para transferir la salida de cada agrupación 214A-214N a cualquier unidad de partición 220A-220N o a otra agrupación 214A-214N que puede realizar operaciones de procesamiento adicionales en la salida. Cada agrupación 214A-214N puede comunicarse con la interfaz de memoria 218 a través de la barra cruzada de memoria 216 para leer desde o escribir en varios dispositivos de memoria externos. En una forma de realización, la barra cruzada de memoria 216 tiene una conexión a la interfaz de memoria 218 para comunicarse con la unidad de E/S 204, así como una conexión a una instancia local de la memoria de procesador paralelo 222, lo que permite que las unidades de procesamiento dentro de las diferentes agrupaciones de procesamiento 214A-214N se comuniquen con la memoria de sistema u otra memoria que no sea local a la unidad de procesamiento paralelo 202. En una forma de realización, la barra cruzada de memoria 216 puede utilizar canales virtuales para separar flujos de tráfico entre las agrupaciones 214A-214N y las unidades de partición 220A-220N.

Si bien una única instancia de la unidad de procesamiento paralelo 202 se ilustra dentro del procesador paralelo 200, se puede incluir cualquier número de instancias de la unidad de procesamiento paralelo 202. Por ejemplo, se pueden proporcionar múltiples instancias de la unidad de procesamiento paralelo 202 en una única tarjeta complementaria, o se pueden interconectar múltiples tarjetas complementarias. Las diferentes instancias de la unidad de procesamiento paralelo 202 pueden estar configuradas para interfuncionar incluso si las diferentes instancias tienen diferentes cantidades de núcleos de procesamiento, diferentes cantidades de memoria de procesador paralelo local y/u otras diferencias de configuración. Por ejemplo, y en una forma de realización, algunas instancias de la unidad de procesamiento paralelo 202 pueden incluir unidades de coma flotante de mayor precisión con respecto a otras instancias. Los sistemas que incorporan una o más instancias de la unidad de procesamiento paralelo 202 o del procesador paralelo 200 se pueden implementar en diversas configuraciones y factores de forma, que incluyen, pero sin limitarse a, ordenadores personales de escritorio, portátiles o de mano, servidores, estaciones de trabajo, consolas de juegos y/o sistemas integrados.

La FIG. 2B es un diagrama de bloques de una unidad de partición 220, de acuerdo con una forma de realización. En una forma de realización, la unidad de partición 220 es una instancia de una de las unidades de partición 220A-220N de la FIG. 2A. Tal como se ilustra, la unidad de partición 220 incluye una memoria caché L2 221, una interfaz de *búfer* de fotogramas 225 y una ROP 226 (unidad de operaciones de rasterización). La memoria caché L2 221 es una memoria caché de lectura/escritura que está configurada para realizar operaciones de carga y almacenamiento recibidas desde la barra cruzada de memoria 216 y la ROP 226. La memoria caché L2 221 transfiere errores de lectura y solicitudes de reescritura urgentes a la interfaz de *búfer* de fotogramas 225 para su procesamiento. También se pueden enviar actualizaciones al *búfer* de fotogramas a través de la interfaz de *búfer* de fotogramas 225 para su procesamiento. En una forma de realización, la interfaz de *búfer* de fotogramas 225 interactúa con una de las unidades de memoria de la memoria de procesador paralelo, tal como las unidades de memoria 224A-224N de la FIG. 2 (por ejemplo, dentro de la memoria de procesador paralelo 222).

En aplicaciones gráficas, la ROP 226 es una unidad de procesamiento que realiza operaciones de rasterización tales como estarcido, prueba Z, mezcla y similares. A continuación, la ROP 226 proporciona datos de gráficos procesados que se almacenan en la memoria de gráficos. En algunas formas de realización, la ROP 226 incluye lógica de compresión para comprimir datos de profundidad o color que se escriben en memoria y descomprimir datos de profundidad o color que se leen de la memoria. La lógica de compresión puede ser lógica de compresión sin pérdidas que utiliza uno o más de múltiples algoritmos de compresión. El tipo de compresión realizado por la ROP 226 puede variar en función de las características estadísticas de los datos a comprimir. Por ejemplo, en una forma de realización, la compresión de color delta se realiza en datos de profundidad y color por cada mosaico.

En algunas formas de realización, la ROP 226 está incluida dentro de cada agrupación de procesamiento (por ejemplo, agrupación 214A-214N de la FIG. 2) en lugar de dentro de la unidad de partición 220. En una forma de realización de este tipo, las solicitudes de lectura y escritura de datos de píxeles se transmiten a través de la barra cruzada de memoria 216 en lugar de datos de fragmentos de píxeles. Los datos de gráficos procesados pueden mostrarse en un dispositivo de visualización, tal como un dispositivo de los uno o más dispositivos de visualización 110 de la FIG. 1, encaminarse para un procesamiento adicional mediante el/los procesador(es) 102, o encaminarse para un procesamiento adicional por una de las entidades de procesamiento dentro del procesador paralelo 200 de la FIG. 2A.

La FIG. 2C es un diagrama de bloques de una agrupación de procesamiento 214 dentro de una unidad de procesamiento paralelo, de acuerdo con una forma de realización. En una forma de realización, la agrupación de procesamiento es una instancia de una de las agrupaciones de procesamiento 214A-214N de la FIG. 2. La agrupación de procesamiento 214 puede estar configurada para ejecutar muchos subprocesos en paralelo, donde el término "subproceso" se refiere a una instancia de un programa particular que se ejecuta en un conjunto particular de datos de entrada. En algunas formas de realización se utilizan técnicas de emisión de instrucciones de tipo "una sola instrucción, múltiples datos" (SIMD) para respaldar la ejecución en paralelo de una gran cantidad de subprocesos sin proporcionar múltiples unidades de instrucciones independientes. En otras formas de realización se utilizan técnicas de tipo "una sola instrucción, múltiples subprocesos" (SIMT) para respaldar la ejecución en paralelo de una gran cantidad de subprocesos generalmente sincronizados, utilizando una unidad de instrucciones común configurada para emitir instrucciones a un conjunto de motores de procesamiento dentro de cada una de las agrupaciones de procesamiento. A diferencia de un régimen de ejecución SIMD, donde todos los motores de procesamiento ejecutan típicamente instrucciones idénticas, la ejecución SIMT permite que diferentes subprocesos sigan más fácilmente rutas de ejecución divergentes a través de un programa de subproceso dado. Los expertos en la técnica entenderán que un régimen de procesamiento SIMD representa un subconjunto funcional de un régimen de procesamiento SIMT.

El funcionamiento de la agrupación de procesamiento 214 se puede controlar a través de un gestor de procesamiento en cadena 232 que distribuye tareas de procesamiento a procesadores paralelos SIMT. El gestor de procesamiento en cadena 232 recibe instrucciones desde el planificador 210 de la FIG. 2 y gestiona la ejecución de esas instrucciones a través de un multiprocesador de gráficos 234 y/o una unidad de textura 236. El multiprocesador de gráficos 234 ilustrado es una instancia ejemplar de un procesador paralelo SIMT. Sin embargo, varios tipos de procesadores paralelos SIMT de diferentes arquitecturas se pueden incluir dentro de la agrupación de procesamiento 214. Una o más instancias del multiprocesador de gráficos 234 se pueden incluir dentro de una agrupación de procesamiento 214. El multiprocesador de gráficos 234 puede procesar datos y se puede utilizar una barra cruzada de datos 240 para distribuir los datos procesados a uno de múltiples destinos posibles, incluidas otras unidades de sombreado. El gestor de procesamiento en cadena 232 puede facilitar la distribución de datos procesados mediante la especificación de destinos para los datos procesados que se distribuirán a través de la barra cruzada de datos 240.

Cada multiprocesador de gráficos 234 dentro de la agrupación de procesamiento 214 puede incluir un conjunto idéntico de lógica de ejecución funcional (por ejemplo, unidades de lógica aritmética, unidades de almacenamiento de carga, etc.). La lógica de ejecución funcional se puede configurar en forma de procesamiento en cadena, en donde se pueden emitir nuevas instrucciones antes de que se completen las instrucciones anteriores. La lógica de ejecución funcional admite una variedad de operaciones, incluidas la aritmética de números enteros y de coma flotante, operaciones de comparación, operaciones lógicas, desplazamiento de bits y cálculo de varias funciones algebraicas. En una forma de realización, el mismo hardware de unidad funcional puede aprovecharse para realizar diferentes operaciones, y cualquier combinación de unidades funcionales puede estar presente.

Las instrucciones transmitidas a la agrupación de procesamiento 214 constituyen un subproceso. Un conjunto de subprocesos que se ejecuta a través del conjunto de motores de procesamiento paralelo es un grupo de subprocesos. Un grupo de subprocesos ejecuta el mismo programa con diferentes datos de entrada. Cada subproceso dentro de un grupo de subprocesos se puede asignar a un motor de procesamiento diferente dentro de un multiprocesador de gráficos 234. Un grupo de subprocesos puede incluir menos subprocesos que el número de motores de procesamiento dentro del multiprocesador de gráficos 234. Cuando un grupo de subprocesos incluye menos subprocesos que el número de motores de procesamiento, uno o más de los motores de procesamiento pueden estar inactivos durante los ciclos en los que se procesa ese grupo de subprocesos. Un grupo de subprocesos también puede incluir más subprocesos que el número de motores de procesamiento dentro del multiprocesador de gráficos 234. Cuando el grupo de subprocesos incluye más subprocesos que el número de motores de procesamiento dentro del multiprocesador de gráficos 234, el procesamiento se puede realizar durante ciclos de reloj consecutivos. En una

forma de realización se pueden ejecutar múltiples grupos de subprocesos simultáneamente en un multiprocesador de gráficos 234.

En una forma de realización, el multiprocesador de gráficos 234 incluye una memoria caché interna para realizar operaciones de carga y almacenamiento. En una forma de realización, el multiprocesador de gráficos 234 puede renunciar a una memoria caché interna y usar una memoria caché (por ejemplo, la memoria caché L1 308) dentro del grupo de procesamiento 214. Cada multiprocesador de gráficos 234 también tiene acceso a memorias caché L2 dentro de las unidades de partición (por ejemplo, unidades de partición 220A-220N de la FIG. 2) compartidas entre todas las agrupaciones de procesamiento 214 y se puede utilizar para transferir datos entre subprocesos. El multiprocesador de gráficos 234 también puede acceder a memoria global fuera de chip, que puede incluir una o más de entre memoria de procesador paralelo local y/o memoria de sistema. Cualquier memoria externa a la unidad de procesamiento paralelo 202 puede utilizarse como memoria global. Las formas de realización en las que la agrupación de procesamiento 214 incluye múltiples instancias del multiprocesador de gráficos 234 pueden compartir instrucciones y datos comunes, que se pueden almacenar en la memoria caché L1 308.

Cada agrupación de procesamiento 214 puede incluir una MMU 245 (unidad de gestión de memoria) que está configurada para correlacionar direcciones virtuales con direcciones físicas. En otras formas de realización, una o más instancias de la MMU 245 pueden residir dentro de la interfaz de memoria 218 de la FIG. 2. La MMU 245 incluye un conjunto de entradas de tabla de páginas (PTE) utilizadas para correlacionar una dirección virtual con una dirección física de un mosaico y, opcionalmente, un índice de línea de memoria caché. La MMU 245 puede incluir *búferes* de conversión anticipada de direcciones (TLB) o memorias caché que pueden residir dentro del multiprocesador de gráficos 234 o la memoria caché L1 o agrupación de procesamiento 214. La dirección física se procesa para distribuir la ubicación de acceso a datos de superficie para permitir una intercalación de solicitudes eficaz entre unidades de partición. El índice de línea de memoria caché se puede utilizar para determinar si una solicitud de una línea de memoria caché es un resultado positivo o un fallo.

En aplicaciones gráficas y de cálculo, una agrupación de procesamiento 214 puede configurarse de manera que cada multiprocesador de gráficos 234 esté acoplado a una unidad de textura 236 para realizar operaciones de correlación de textura, por ejemplo, determinar posiciones de muestras de textura, leer datos de textura y filtrar los datos de textura. Los datos de textura se leen de una memoria caché L1 de textura interna (no mostrada) o, en algunas formas de realización de la memoria caché L1, dentro del multiprocesador de gráficos 234, y se obtienen de una memoria caché L2, una memoria de procesador paralelo local o una memoria de sistema, según sea necesario. Cada multiprocesador de gráficos 234 proporciona tareas procesadas a la barra cruzada de datos 240 para proporcionar la tarea procesada a otra agrupación de procesamiento 214 para un procesamiento adicional o para almacenar la tarea procesada en una memoria caché L2, una memoria de procesador paralelo local o una memoria de sistema a través de la barra cruzada de memoria 216. Una preROP 242 (unidad de operaciones de rasterización previa) está configurada para recibir datos del multiprocesador de gráficos 234, dirigir los datos a unidades ROP, que pueden estar ubicadas junto con unidades de partición descritas en el presente documento (por ejemplo, unidades de partición 220A-220N de la FIG. 2). La unidad preROP 242 puede realizar optimizaciones para la mezcla de colores, organizar datos de color de píxeles y realizar conversiones de direcciones.

Se apreciará que la arquitectura principal descrita en el presente documento es ilustrativa y que pueden realizarse variaciones y modificaciones. Cualquier número de unidades de procesamiento, por ejemplo, multiprocesador de gráficos 234, unidades de textura 236, preROP 242, etc., puede incluirse dentro de una agrupación de procesamiento 214. Además, si bien solo se muestra una agrupación de procesamiento 214, una unidad de procesamiento paralelo como la descrita en el presente documento puede incluir cualquier número de instancias de la agrupación de procesamiento 214. En una forma de realización, cada agrupación de procesamiento 214 puede estar configurada para funcionar de forma independiente a otras agrupaciones de procesamiento 214 usando unidades de procesamiento, memorias caché L1, etc., individuales y distintas.

La FIG. 2D muestra un multiprocesador de gráficos 234, de acuerdo con una forma de realización. En dicha forma de realización, el multiprocesador de gráficos 234 está acoplado al gestor de procesamiento en cadena 232 de la agrupación de procesamiento 214. El multiprocesador de gráficos 234 tiene una ejecución mediante procesamiento en cadena que incluye, pero sin limitarse a, una memoria caché de instrucciones 252, una unidad de instrucciones 254, una unidad de correlación de direcciones 256, un archivo de registros 258, uno o más núcleos de unidad de procesamiento de gráficos de propósito general (GPGPU) 262 y una o más unidades de carga/almacenamiento 266. Los núcleos de GPGPU 262 y las unidades de carga/almacenamiento 266 están acopladas a una memoria caché 272 y una memoria compartida 270 a través de una interconexión de memoria caché y de memoria 268.

En una forma de realización, la memoria caché de instrucciones 252 recibe desde el gestor de procesamiento en cadena 232 un flujo de instrucciones a ejecutar. Las instrucciones se almacenan en la memoria caché de instrucciones 252 y se envían para su ejecución mediante la unidad de instrucciones 254. La unidad de instrucciones 254 puede enviar instrucciones como grupos de subprocesos (por ejemplo, deformaciones (*warps*)), donde cada subproceso del grupo de subprocesos está asignado a una unidad de ejecución diferente dentro del núcleo de GPGPU 262. Una instrucción puede acceder a cualquier espacio de direcciones local, compartido o global especificando una dirección dentro de un espacio de direcciones unificado. La unidad de correlación de direcciones 256 se puede utilizar para

convertir direcciones del espacio de direcciones unificado en una dirección de memoria distinta a la que se puede acceder mediante las unidades de carga/almacenamiento 266.

El archivo de registros 258 proporciona un conjunto de registros para las unidades funcionales del multiprocesador de gráficos 324. El archivo de registros 258 proporciona almacenamiento temporal para los operandos conectados a las rutas de datos de las unidades funcionales (por ejemplo, núcleos de GPGPU 262, unidades de carga/almacenamiento 266) del multiprocesador de gráficos 324. En una forma de realización, el archivo de registros 258 se divide entre cada una de las unidades funcionales de manera que a cada unidad funcional se le asigna una parte dedicada del archivo de registros 258. En una forma de realización, el archivo de registros 258 se divide entre las diferentes deformaciones que ejecuta el multiprocesador de gráficos 324.

Cada núcleo de GPGPU 262 puede incluir unidades de coma flotante (FPU) y/o unidades aritmético-lógicas (ALU) de números enteros que se utilizan para ejecutar instrucciones del multiprocesador de gráficos 324. Los núcleos de GPGPU 262 pueden ser similares en arquitectura o pueden diferir en arquitectura, de acuerdo con las formas de realización. Por ejemplo, y en una forma de realización, una primera parte de los núcleos de GPGPU 262 incluye una FPU de precisión simple y una ALU de números enteros, mientras que una segunda parte de los núcleos de GPGPU incluye una FPU de doble precisión. En una forma de realización, las FPU pueden implementar la norma IEEE 754-2008 para la aritmética de coma flotante o habilitar una aritmética de coma flotante de precisión variable. El multiprocesador de gráficos 324 puede incluir adicionalmente una o más unidades de función fija o función especial para realizar funciones específicas tales como operaciones de combinación de píxeles o rectángulos de copia. En una forma de realización, uno o más de los núcleos de GPGPU también pueden incluir lógica de función fija o especial.

En una forma de realización, los núcleos de GPGPU 262 incluyen lógica SIMD capaz de realizar una única instrucción en múltiples conjuntos de datos. En una forma de realización, los núcleos de GPGPU 262 pueden ejecutar de manera física instrucciones SIMD4, SIMD8 y SIMD16 y ejecutar de manera lógica instrucciones SIMD1, SIMD2 y SIMD32. Las instrucciones SIMD para los núcleos de GPGPU pueden generarse en tiempo de compilación por un compilador de sombreado o generarse automáticamente cuando se ejecutan programas escritos y compilados para arquitecturas de tipo "un solo programa, múltiples datos" (SPMD) o SIMT. Múltiples subprocesos de un programa configurado para el modelo de ejecución SIMT se pueden ejecutar a través de una única instrucción SIMD. Por ejemplo, y en una forma de realización, ocho subprocesos SIMT que realizan las mismas operaciones, u otras similares, se pueden ejecutar en paralelo a través de una sola unidad lógica SIMD8.

La interconexión de memoria caché y de memoria 268 es una red de interconexión que conecta cada una de las unidades funcionales del multiprocesador de gráficos 324 al archivo de registros 258 y a la memoria compartida 270. En una forma de realización, la interconexión de memoria caché y de memoria 268 es una interconexión de barras cruzadas que permite que la unidad de carga/almacenamiento 266 implemente operaciones de carga y almacenamiento entre la memoria compartida 270 y el archivo de registros 258. El archivo de registros 258 puede funcionar a la misma frecuencia que los núcleos de GPGPU 262, por lo que la transferencia de datos entre los núcleos de GPGPU 262 y el archivo de registros 258 tiene una latencia muy baja. La memoria compartida 270 se puede utilizar para permitir la comunicación entre subprocesos que se ejecutan en las unidades funcionales dentro del multiprocesador de gráficos 234. La memoria caché 272 se puede utilizar como una memoria caché de datos, por ejemplo, para almacenar en caché los datos de textura comunicados entre las unidades funcionales y la unidad de textura 236. La memoria compartida 270 también se puede utilizar como un programa gestionado en memoria caché. Los subprocesos que se ejecutan en los núcleos de GPGPU 262 pueden almacenar de manera programática datos dentro de la memoria compartida, además de los datos almacenados automáticamente en memoria caché que se almacenan dentro de la memoria caché 272.

Las FIGS. 3A-3B ilustran multiprocesadores de gráficos adicionales, de acuerdo con las formas de realización. Los multiprocesadores de gráficos ilustrados 325, 350 son variantes del multiprocesador de gráficos 234 de la FIG. 2C. Los multiprocesadores de gráficos ilustrados 325, 350 se pueden configurar como un multiprocesador de transmisión continua (SM) capaz de ejecutar simultáneamente una gran cantidad de subprocesos de ejecución.

La FIG. 3A muestra un multiprocesador de gráficos 325 de acuerdo con una forma de realización adicional. El multiprocesador de gráficos 325 incluye múltiples instancias adicionales de unidades de recursos de ejecución con respecto al multiprocesador de gráficos 234 de la FIG. 2D. Por ejemplo, el multiprocesador de gráficos 325 puede incluir múltiples instancias de unidad de instrucción 332A-332B, de archivo de registros 334A-334B y de unidad de textura 344A-344B. El multiprocesador de gráficos 325 también incluye múltiples conjuntos de unidades gráficas o de ejecución de cálculos (por ejemplo, núcleo de GPGPU 336A-336B, núcleo de GPGPU 337A-337B, núcleo de GPGPU 338A-338B) y múltiples conjuntos de unidades de carga/almacenamiento 340A-340B. En una forma de realización, las unidades de recursos de ejecución tienen una memoria caché de instrucciones común 330, una memoria caché de texturas y/o datos 342 y una memoria compartida 346.

Los diversos componentes pueden comunicarse a través de una estructura de interconexión 327. En una forma de realización, la estructura de interconexión 327 incluye uno o más conmutadores de barras cruzadas para permitir la comunicación entre los diversos componentes del multiprocesador de gráficos 325. En una forma de realización, la estructura de interconexión 327 es una capa de estructura de red de alta velocidad distinta sobre la cual se apila cada

componente del multiprocesador de gráficos 325. Los componentes del multiprocesador de gráficos 325 se comunican con componentes remotos a través de la estructura de interconexión 327. Por ejemplo, cada núcleo de GPGPU 336A-336B, 337A-337B y 3378A-338B puede comunicarse con la memoria compartida 346 a través de la estructura de interconexión 327. La estructura de interconexión 327 puede arbitrar la comunicación dentro del multiprocesador de gráficos 325 para garantizar una asignación de ancho de banda justa entre los componentes.

La FIG. 3B muestra un multiprocesador de gráficos 350 de acuerdo con una forma de realización adicional. El procesador de gráficos incluye múltiples conjuntos de recursos de ejecución 356A-356D, donde cada conjunto de recursos de ejecución incluye múltiples unidades de instrucción, archivos de registros, núcleos de GPGPU y unidades de almacenamiento de carga, como se ilustra en la FIG. 2D y la FIG. 3A. Los recursos de ejecución 356A-356D pueden actuar conjuntamente con la(s) unidad(es) de textura 360A-360D para operaciones de textura, mientras comparten una memoria caché de instrucciones 354 y una memoria compartida 362. En una forma de realización, los recursos de ejecución 356A-356D pueden compartir una memoria caché de instrucciones 354 y una memoria compartida 362, así como múltiples instancias de una memoria caché de texturas y/o datos 358A-358B. Los diversos componentes pueden comunicarse a través de una estructura de interconexión 352 similar a la estructura de interconexión 327 de la FIG. 3A.

Los expertos en la técnica entenderán que la arquitectura descrita en las FIGS. 1, 2A-2D y 3A-3B es descriptiva y no limitante del alcance de las presentes formas de realización. Por lo tanto, las técnicas descritas en el presente documento pueden implementarse en cualquier unidad de procesamiento configurada adecuadamente, incluidos, sin limitación, uno o más procesadores de aplicaciones móviles, una o más unidades de procesamiento central (CPU) de escritorio o servidor, incluidas CPU de múltiples núcleos, una o más unidades de procesamiento paralelo, tal como la unidad de procesamiento paralelo 202 de la FIG. 2, así como uno o más procesadores de gráficos o unidades de procesamiento de propósito especial, sin apartarse del alcance de las formas de realización descritas en el presente documento.

En algunas formas de realización, un procesador paralelo o GPGPU descrito en el presente documento está acoplado de manera comunicativa a núcleos de ordenador principal/procesador para acelerar operaciones de gráficos, operaciones de aprendizaje automático, operaciones de análisis de patrones y diversas funciones de GPU de propósito general (GPGPU). La GPU puede estar acoplada de manera comunicativa al procesador/núcleos de ordenador principal a través de un bus u otra interconexión (por ejemplo, una interconexión de alta velocidad, tal como PCIe o NVLink). En otras formas de realización, la GPU puede estar integrada en el mismo encapsulado o chip que los núcleos y estar acoplada de manera comunicativa a los núcleos a través de un bus/interconexión de procesador internos (es decir, internos al encapsulado o chip). Independientemente de la manera en que se conecte la GPU, los núcleos de procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidos en un descriptor de trabajo. A continuación, la GPU utiliza circuitos/lógica dedicados para procesar eficazmente estos comandos/instrucciones.

Técnicas para la interconexión entre GPU y procesador de ordenador principal

La FIG. 4A ilustra una arquitectura ejemplar en la que una pluralidad de GPU 410-413 están acopladas de manera comunicativa a una pluralidad de procesadores de múltiples núcleos 405-406 a través de enlaces de alta velocidad 440-443 (por ejemplo, buses, interconexiones punto a punto, etc.). En una forma de realización, los enlaces de alta velocidad 440-443 admiten un caudal de tráfico de 4 GB/s, 30 GB/s, 80 GB/s o más, dependiendo de la implementación. Se pueden utilizar diversos protocolos de interconexión que incluyen, pero sin limitarse a, PCIe 4.0 o 5.0 y NVLink 2.0. Sin embargo, los principios subyacentes de la invención no se limitan a ningún protocolo de comunicación o caudal de tráfico particular.

Además, en una forma de realización, dos o más de las GPU 410-413 están interconectadas a través de enlaces de alta velocidad 444-445, que pueden implementarse utilizando los mismos o diferentes protocolos/enlaces que los utilizados para los enlaces de alta velocidad 440-443. De manera similar, dos o más de los procesadores de múltiples núcleos 405-406 pueden conectarse a través del enlace de alta velocidad 433, que puede ser buses de múltiples procesadores simétricos (SMP) que funcionan a 20 GB/s, 30 GB/s, 120 GB/s o más. De manera alternativa, toda la comunicación entre los diversos componentes de sistema mostrados en la FIG. 4A puede lograrse utilizando los mismos protocolos/enlaces (por ejemplo, a través de una estructura de interconexión común). Sin embargo, como se ha mencionado, los principios subyacentes de la invención no se limitan a ningún tipo particular de tecnología de interconexión.

En una forma de realización, cada procesador de múltiples núcleos 405-406 está acoplado de manera comunicativa a una memoria de procesador 401-402, a través de interconexiones de memoria 430-431, respectivamente, y cada GPU 410-413 está acoplada de manera comunicativa a la memoria de GPU 420-423 a través de interconexiones de memoria de GPU 450-453, respectivamente. Las interconexiones de memoria 430-431 y 450-453 pueden utilizar las mismas o diferentes tecnologías de acceso a memoria. A modo de ejemplo, y no de limitación, las memorias de procesador 401-402 y las memorias de GPU 420-423 pueden ser memorias volátiles, tales como memorias de acceso aleatorio dinámicas (DRAM) (incluidas DRAM apiladas), SDRAM de DDR de gráficos (GDDR) (por ejemplo, GDDR5, GDDR6) o memoria de alto ancho de banda (HBM) y/o pueden ser memorias no volátiles tales como 3D XPoint o

Nano-Ram. En una forma de realización, una parte de las memorias puede ser memoria volátil y otra parte puede ser memoria no volátil (por ejemplo, usando una jerarquía de memoria de dos niveles (2LM)).

Tal como se describe a continuación, aunque los diversos procesadores 405-406 y GPU 410-413 pueden estar acoplados físicamente a una memoria particular 401-402, 420-423, respectivamente, se puede implementar una arquitectura de memoria unificada en la que el mismo espacio de direcciones de sistema virtuales (también denominado espacio de "direcciones efectivas") se distribuye entre todas las diversas memorias físicas. Por ejemplo, cada memoria de procesador 401-402 puede comprender 64 GB del espacio de direcciones de memoria de sistema y cada memoria de GPU 420-423 puede comprender 32 GB del espacio de direcciones de memoria de sistema (lo que da como resultado una memoria direccionable con un total de 256 GB en este ejemplo).

La FIG. 4B ilustra detalles adicionales para una interconexión entre un procesador de múltiples núcleos 407 y un módulo de aceleración de gráficos 446 de acuerdo con una forma de realización. El módulo de aceleración de gráficos 446 puede incluir uno o más chips de GPU integrados en una tarjeta de línea que está acoplada al procesador 407 a través del enlace de alta velocidad 440. De manera alternativa, el módulo de aceleración de gráficos 446 puede estar integrado en el mismo encapsulado o chip que el procesador 407.

El procesador 407 ilustrado incluye una pluralidad de núcleos 460A-460D, cada uno con un *búfer* de conversión anticipada de direcciones 461A-461D y una o más memorias caché 462A-462D. Los núcleos pueden incluir otros diversos componentes para ejecutar instrucciones y procesar datos que no se ilustran para no complicar los principios subyacentes de la invención (por ejemplo, unidades de obtención de instrucciones, unidades de predicción de ramas, descodificadores, unidades de ejecución, *búferes* de reordenamiento, etc.). Las memoria caché 462A-462D pueden comprender memoria caché de nivel 1 (L1) y nivel 2 (L2). Además, una o más memorias caché compartidas 426 se pueden incluir en la jerarquía de almacenamiento en caché y se pueden compartir por conjuntos de los núcleos 460A-460D. Por ejemplo, una forma de realización del procesador 407 incluye 24 núcleos, cada uno con su propia memoria caché L1, doce memorias caché L2 compartidas y doce memorias caché L3 compartidas. En esta forma de realización, una de las memorias caché L2 y L3 son compartidas por dos núcleos adyacentes. El procesador 407 y el módulo de integración de acelerador de gráficos 446 están conectados a la memoria de sistema 441, que puede incluir las memorias de procesador 401-402.

Se mantiene la coherencia para los datos e instrucciones almacenados en las diversas memorias caché 462A-462D, 456 y la memoria de sistema 441 mediante una comunicación entre núcleos a través de un bus de coherencia 464. Por ejemplo, cada memoria caché puede tener una lógica/circuitos de coherencia de memoria caché asociados a la misma para comunicarse a través del bus de coherencia 464 en respuesta a lecturas o escrituras detectadas en líneas de memoria caché particulares. En una implementación, se implementa un protocolo de espionaje (*snooping*) de memoria caché a través del bus de coherencia 464 para espiar los accesos a memoria caché. Las técnicas de espionaje/coherencia de memoria caché son ampliamente conocidas por los expertos en la técnica y no se describirán en detalle en el presente documento para no complicar los principios subyacentes de la invención.

En una forma de realización, un circuito intermediario (*proxy*) 425 acopla de forma comunicativa el módulo de aceleración de gráficos 446 al bus de coherencia 464, lo que permite que el módulo de aceleración de gráficos 446 participe en el protocolo de coherencia de memoria caché como un homólogo de los núcleos. En particular, una interfaz 435 proporciona conectividad al circuito intermediario 425 a través del enlace de alta velocidad 440 (por ejemplo, un bus PCIe, NVLink, etc.), y una interfaz 437 conecta el módulo de aceleración de gráficos 446 al enlace de alta velocidad 440.

En una implementación, un circuito de integración de acelerador 436 proporciona gestión de memoria caché, acceso a memoria, gestión de contexto y servicios de gestión de interrupciones en nombre de una pluralidad de motores de procesamiento de gráficos 431, 432, N del módulo de aceleración de gráficos 446. Cada motor de procesamiento de gráficos 431, 432, N puede comprender una unidad de procesamiento de gráficos (GPU) distinta. De manera alternativa, los motores de procesamiento de gráficos 431, 432, N pueden comprender diferentes tipos de motores de procesamiento de gráficos dentro de una GPU, tales como unidades de ejecución de gráficos, motores de procesamiento de medios (por ejemplo, codificadores/descodificadores de vídeo), muestreadores y motores *blit*. En otras palabras, el módulo de aceleración de gráficos puede ser una GPU con una pluralidad de motores de procesamiento de gráficos 431-432, N, o los motores de procesamiento de gráficos 431-432, N pueden ser GPU individuales integradas en un encapsulado, tarjeta de línea o chip común.

En una forma de realización, el circuito de integración de acelerador 436 incluye una unidad de gestión de memoria (MMU) 439 para realizar diversas funciones de gestión de memoria tales como conversiones de memoria virtual a física (también denominadas conversiones de memoria efectiva a real) y protocolos de acceso a memoria para acceder a la memoria de sistema 441. La MMU 439 también puede incluir un *búfer* de conversión anticipada de direcciones (TLB) (no mostrado) para almacenar en caché las conversiones de dirección virtual/efectiva a física/real. En una implementación, una memoria caché 438 almacena comandos y datos para un acceso eficiente por parte de los motores de procesamiento de gráficos 431-432, N. En una forma de realización, los datos almacenados en la memoria caché 438 y las memorias gráficas 433-434, N se mantienen coherentes con las memorias caché principales 462A-462D, 456 y la memoria de sistema 411. Como se mencionó, esto se puede lograr mediante el circuito intermediario

425 que participa en el mecanismo de coherencia de memoria caché en nombre de la memoria caché 438 y las memorias 433-434, N (por ejemplo, enviar actualizaciones a la memoria caché 438 relacionadas con modificaciones/accesos de líneas de memoria caché en las memorias caché de procesador 462A-462D, 456 y recibir actualizaciones desde la memoria caché 438).

Un conjunto de registros 445 almacena datos de contexto para los subprocesos ejecutados por los motores de procesamiento de gráficos 431-432, N y un circuito de gestión de contexto 448 gestiona los contextos de subprocesos. Por ejemplo, el circuito de gestión de contexto 448 puede realizar operaciones de guardado y restauración para guardar y restaurar contextos de los diversos subprocesos durante conmutadores de contexto (por ejemplo, donde se guarda un primer subproceso y se almacena un segundo subproceso de modo que el segundo subproceso se puede ejecutar mediante un motor de procesamiento de gráficos). Por ejemplo, en un conmutador de contexto, el circuito de gestión de contexto 448 puede almacenar valores de registro actuales en una región designada en la memoria (por ejemplo, identificada mediante un puntero de contexto). A continuación, puede restaurar los valores de registro al volver al contexto. En una forma de realización, un circuito de gestión de interrupciones 447 recibe y procesa interrupciones recibidas desde los dispositivos del sistema.

En una implementación, las direcciones virtuales/efectivas de un motor de procesamiento de gráficos 431 se convierten en direcciones reales/físicas en la memoria de sistema 411 mediante la MMU 439. Una forma de realización del circuito de integración de acelerador 436 admite múltiples (por ejemplo, 4, 8, 16) módulos de acelerador de gráficos 446 y/u otros dispositivos aceleradores. El módulo acelerador de gráficos 446 puede estar dedicado a una única aplicación ejecutada en el procesador 407 o puede estar compartido entre múltiples aplicaciones. En una forma de realización, se presenta un entorno de ejecución de gráficos virtualizado en el que los recursos de los motores de procesamiento de gráficos 431-432, N se comparten con múltiples aplicaciones o máquinas virtuales (VM). Los recursos se pueden subdividir en "sectores" que se asignan a diferentes VM y/o aplicaciones en función de los requisitos de procesamiento y las prioridades asociadas a las VM y/o aplicaciones.

Por lo tanto, el circuito de integración de acelerador actúa como un puente al sistema para el módulo de aceleración de gráficos 446 y proporciona servicios de conversión de direcciones y de memoria caché de sistema. Además, el circuito de integración de acelerador 436 puede proporcionar unidades de virtualización para que el procesador de ordenador principal gestione la virtualización de los motores de procesamiento de gráficos, interrupciones y gestión de memoria.

Debido a que los recursos de hardware de los motores de procesamiento de gráficos 431-432, N se correlacionan explícitamente con el espacio de direcciones reales visto por el procesador de ordenador principal 407, cualquier procesador de ordenador principal puede dirigirse a estos recursos directamente usando un valor de dirección efectivo. Una función del circuito de integración de acelerador 436, en una forma de realización, es la separación física de los motores de procesamiento de gráficos 431-432, N de modo al sistema le parezca que son unidades independientes.

Como se mencionó, en la forma de realización ilustrada, una o más memorias gráficas 433-434, M están acopladas a cada uno de los motores de procesamiento de gráficos 431-432, N, respectivamente. Las memorias gráficas 433-434, M almacenan instrucciones y datos que son procesados por cada uno de los motores de procesamiento de gráficos 431-432, N. Las memorias gráficas 433-434, M pueden ser memorias volátiles, tales como DRAM (incluidas DRAM apiladas), memoria GDDR (por ejemplo, GDDR5, GDDR6) o HBM, y/o pueden ser memorias no volátiles, tales como 3D XPoint o Nano-Ram.

En una forma de realización, para reducir el tráfico de datos a través del enlace de alta velocidad 440, se utilizan técnicas de sesgado para garantizar que los datos almacenados en las memorias gráficas 433-434, M sean datos que serán utilizados con mayor frecuencia por los motores de procesamiento de gráficos 431-432, N y, preferentemente, no utilizados por los núcleos 460A-460D (al menos no con frecuencia). De manera similar, el mecanismo de sesgado intenta mantener los datos necesarios por los núcleos (y, preferentemente, no los motores de procesamiento de gráficos 431-432, N) dentro de las memorias caché 462A-462D, 456 de los núcleos y la memoria de sistema 411.

La FIG. 4C ilustra otra forma de realización en la que el circuito de integración de acelerador 436 está integrado dentro del procesador 407. En esta forma de realización, los motores de procesamiento de gráficos 431-432, N se comunican directamente a través del enlace de alta velocidad 440 con el circuito de integración de acelerador 436 a través de la interfaz 437 y la interfaz 435 (que, nuevamente, puede utilizar cualquier forma de bus o protocolo de interfaz). El circuito de integración de acelerador 436 puede realizar las mismas operaciones que las descritas con respecto a la FIG. 4B, pero, posiblemente, con un mayor caudal de tráfico dada su proximidad al bus de coherencia 462 y las memorias caché 462A-462D, 426.

Una forma de realización admite diferentes modelos de programación, que incluyen un modelo de programación de procesos dedicados (sin virtualización de módulo de aceleración de gráficos) y modelos de programación compartida (con virtualización). Estos últimos puede incluir modelos de programación que están controlados por el circuito de integración de acelerador 436 y modelos de programación que están controlados por el módulo de aceleración de gráficos 446.

En una forma de realización del modelo de proceso dedicado, los motores de procesamiento de gráficos 431-432, N están dedicados a una única aplicación o proceso bajo un único sistema operativo. La aplicación única puede canalizar otras solicitudes de aplicación hacia los motores gráficos 431-432, N, proporcionando virtualización dentro de una VM/partición.

En los modelos de programación de procesos dedicados, los motores de procesamiento de gráficos 431-432, N, pueden estar compartidos por múltiples particiones de VM/aplicación. Los modelos compartidos requieren un hipervisor de sistema para virtualizar los motores de procesamiento de gráficos 431-432, N para permitir el acceso por parte de cada sistema operativo. En cuanto a sistemas de partición única sin hipervisor, los motores de procesamiento de gráficos 431-432, N pertenecen al sistema operativo. En ambos casos, el sistema operativo puede virtualizar los motores de procesamiento de gráficos 431-432, N para proporcionar acceso a cada proceso o aplicación.

En cuanto al modelo de programación compartida, el módulo de aceleración de gráficos 446 o un motor de procesamiento de gráficos individual 431-432, N selecciona un elemento de proceso usando un gestor de procesos. En una forma de realización, los elementos de proceso se almacenan en la memoria de sistema 411 y pueden referenciarse usando las técnicas de conversión de direcciones efectivas a direcciones reales descritas en el presente documento. El gestor de procesos puede ser un valor específico de implementación proporcionado al proceso de ordenador principal cuando se registra su contexto con el motor de procesamiento de gráficos 431-432, N (es decir, cuando se llama al software de sistema para añadir el elemento de proceso a la lista enlazada de elementos de proceso). Los 16 bits inferiores del gestor de procesos pueden ser el desplazamiento del elemento de proceso dentro de la lista enlazada de elementos de proceso.

La FIG. 4D ilustra un sector de integración de acelerador 490 ejemplar. Tal como se usa en el presente documento, un "sector" comprende una porción específica de los recursos de procesamiento del circuito de integración de acelerador 436. El espacio de direcciones efectivas de aplicación 482 dentro de la memoria de sistema 411 almacena elementos de proceso 483. En una forma de realización, los elementos de proceso 483 se almacenan en respuesta a las invocaciones de GPU 481 de las aplicaciones 480 ejecutadas en el procesador 407. Un elemento de proceso 483 contiene el estado de proceso para la aplicación 480 correspondiente. Un descriptor de trabajo (WD) 484 contenido en el elemento de proceso 483 puede ser una sola tarea solicitada por una aplicación o puede contener un puntero a una cola de tareas. En el segundo caso, el WD 484 es un puntero a la cola de solicitud de tareas en el espacio de direcciones 482 de la aplicación.

El módulo de aceleración de gráficos 446 y/o los motores de procesamiento de gráficos individuales 431-432, N pueden ser compartidos por todos o un subconjunto de los procesos del sistema. Las formas de realización de la invención incluyen una infraestructura para configurar el estado del proceso y enviar un WD 484 a un módulo de aceleración de gráficos 446 para iniciar una tarea en un entorno virtualizado.

En una implementación, el modelo de programación de procesos dedicados es específico de la implementación. En este modelo, un solo proceso acapara el módulo de aceleración de gráficos 446 o un motor de procesamiento de gráficos individual 431. Debido a que un único proceso acapara el módulo de aceleración de gráficos 446, el hipervisor inicializa el circuito de integración de acelerador 436 para la partición acaparadora y el sistema operativo inicializa el circuito de integración de acelerador 436 para el proceso acaparador en el momento en que se asigna el módulo de aceleración de gráficos 446.

Durante el funcionamiento, una unidad de obtención de WD 491 en el sector de integración de acelerador 490 obtiene el siguiente WD 484 que incluye una indicación del trabajo que debe realizar uno de los motores de procesamiento de gráficos del módulo de aceleración de gráficos 446. Los datos del WD 484 pueden almacenarse en registros 445 y usarse por la MMU 439, el circuito de gestión de interrupciones 447 y/o el circuito de gestión de contexto 446, como se ilustra. Por ejemplo, una forma de realización de la MMU 439 incluye circuitos de segmentos/recorridos de página para acceder a tablas de segmentos/páginas 486 dentro del espacio de direcciones virtuales de OS 485. El circuito de gestión de interrupciones 447 puede procesar eventos de interrupción 492 recibidos desde el módulo de aceleración de gráficos 446. Cuando se realizan operaciones de gráficos, una dirección efectiva 493 generada por un motor de procesamiento de gráficos 431-432, N se convierte en una dirección real mediante la MMU 439.

En una forma de realización, el mismo conjunto de registros 445 se duplica para cada motor de procesamiento de gráficos 431-432, N y/o módulo de aceleración de gráficos 446 y puede ser inicializado por el hipervisor o sistema operativo. Cada uno de estos registros duplicados puede incluirse en un sector de integración de acelerador 490. Registros ejemplares que pueden ser inicializados por el hipervisor se muestran en la Tabla 1.

Tabla 1 - Registros inicializados por hipervisor

1	Registro de control de sectores
2	Puntero a área de procesos planificados de direcciones reales (RA)
3	Registro de anulación de máscara de autoridad
4	Desplazamiento de entrada de tabla de vectores de interrupciones
5	Límite de entrada de tabla de vectores de interrupciones
6	Registro de estado
7	ID de partición lógica
8	Puntero de registro de utilización de acelerador de hipervisor de direcciones reales (RA)
9	Registro de descripción de almacenamiento

Registros ejemplares que pueden ser inicializados por el sistema operativo se muestran en la Tabla 2.

Tabla 2 - Registros inicializados del sistema operativo

1	Identificación de procesos y subprocesos
2	Puntero de guardado/restauración de contexto de direcciones efectivas (EA)
3	Puntero de registro de utilización de acelerador de direcciones virtuales (VA)
4	Puntero a tabla de segmentos de almacenamiento de direcciones virtuales (VA)
5	Máscara de autoridad
6	Descriptor de trabajo

En una forma de realización, cada WD 484 es específico de un módulo de aceleración de gráficos particular 446 y/o de un motor de procesamiento de gráficos 431-432, N. Contiene toda la información que un motor de procesamiento de gráficos 431-432, N requiere para realizar su trabajo o puede ser un puntero a una ubicación de memoria donde la aplicación ha configurado una cola de comandos de trabajo a completar.

La FIG. 4E ilustra detalles adicionales para una forma de realización de un modelo compartido. Esta forma de realización incluye un espacio de direcciones reales de hipervisor 498 en el que se almacena una lista de elementos de proceso 499. El espacio de direcciones reales de hipervisor 498 es accesible a través de un hipervisor 496 que virtualiza los motores de módulo de aceleración de gráficos para el sistema operativo 495.

Los modelos de programación compartida permiten que todos o un subconjunto de procesos de todos o un subconjunto de particiones del sistema utilicen un módulo de aceleración de gráficos 446. Hay dos modelos de programación en los que el módulo de aceleración de gráficos 446 es compartido por múltiples procesos y particiones: compartición en fracciones de tiempo y compartición dirigida por gráficos.

En este modelo, el hipervisor de sistema 496 acapara el módulo de aceleración de gráficos 446 y pone sus funciones a disposición de todos los sistemas operativos 495. Para que un módulo de aceleración de gráficos 446 admita la virtualización por parte del hipervisor de sistema 496, el módulo de aceleración de gráficos 446 puede cumplir los siguientes requisitos: 1) La solicitud de tarea de una aplicación debe ser autónoma (es decir, no es necesario mantener el estado entre tareas), o el módulo de aceleración de gráficos 446 debe proporcionar un mecanismo de guardado y restauración de contexto. 2) El módulo de aceleración de gráficos 446 garantiza que la solicitud de tarea de una aplicación se complete en una cantidad de tiempo especificada, incluido cualquier fallo de conversión, o el módulo de aceleración de gráficos 446 proporciona la capacidad de anular el procesamiento de la tarea. 3) El módulo de aceleración de gráficos 446 debe garantizar la equidad entre procesos cuando se está funcionando en el modelo de programación compartida dirigida.

En una forma de realización, para el modelo compartido, se requiere que la aplicación 480 realice una llamada al sistema operativo 495 con un tipo de módulo de aceleración de gráficos 446, un descriptor de trabajo (WD), un valor de registro de máscara de autoridad (AMR) y un puntero a área de guardado/restauración de contexto (CSRP). El tipo de módulo de aceleración de gráficos 446 describe la función de aceleración objetivo para la llamada al sistema. El tipo de módulo de aceleración de gráficos 446 puede ser un valor específico del sistema. El WD tiene un formato específico para el módulo de aceleración de gráficos 446 y puede estar en forma de comando de módulo de aceleración de gráficos 446, puntero de dirección efectiva a una estructura definida por el usuario, puntero de dirección efectiva a una cola de comandos o cualquier otra estructura de datos para describir el trabajo que debe realizar el módulo de aceleración de gráficos 446. En una forma de realización, el valor de AMR es el estado de AMR a utilizar para el proceso actual. El valor transferido al sistema operativo es similar a una configuración de aplicación del AMR.

Si el circuito de integración de acelerador 436 y las implementaciones de módulo de aceleración de gráficos 446 no admiten un registro de anulación de máscara de autoridad de usuario (UAMOR), el sistema operativo puede aplicar el valor de UAMOR actual al valor de AMR antes de pasar el AMR en la llamada al hipervisor. El hipervisor 496 puede aplicar opcionalmente el valor actual del registro de anulación de máscara de autoridad (AMOR) antes de colocar el AMR en el elemento de proceso 483. En una forma de realización, el CSRP es uno de los registros 445 que contiene la dirección efectiva de un área en el espacio de direcciones 482 de la aplicación para que el módulo de aceleración de gráficos 446 guarde y restaure el estado de contexto. Este puntero es opcional si no se requiere que se guarde ningún estado entre tareas o cuando se anula una tarea. El área de guardado/restauración de contexto puede residir en la memoria de sistema.

Al recibir la llamada al sistema, el sistema operativo 495 puede verificar que la aplicación 480 se ha registrado y se le ha otorgado la autoridad para utilizar el módulo de aceleración de gráficos 446. A continuación, el sistema operativo 495 llama al hipervisor 496 con la información mostrada en la Tabla 3**.**

Tabla 3 - Parámetros de llamada del OS al hipervisor

1	Un descriptor de trabajo (WD)
2	Un valor de registro de máscara de autoridad (AMR) (posiblemente enmascarado).
3	Puntero a área de guardado/restauración de contexto (CSRP) de direcciones efectivas (EA)
4	Un ID de proceso (PID) y un ID de subproceso (TID) opcional
5	Un puntero de registro de utilización de acelerador (AURP) de direcciones virtuales (VA)
6	La dirección virtual del puntero a tabla de segmentos de almacenamiento (SSTP)
7	Un número de servicio de interrupción lógica (LISN)

Al recibir la llamada al hipervisor, el hipervisor 496 verifica que el sistema operativo 495 se ha registrado y se le ha otorgado la autoridad para utilizar el módulo de aceleración de gráficos 446. A continuación, el hipervisor 496 coloca el elemento de proceso 483 en la lista enlazada de elementos de proceso para el tipo de módulo de aceleración de gráficos 446 correspondiente. El elemento de proceso puede incluir la información mostrada en la Tabla 4**.**

Tabla 4 - Información de elemento de proceso

1	Un descriptor de trabajo (WD)
2	Un valor de registro de máscara de autoridad (AMR) (posiblemente enmascarado).
3	Puntero a área de guardado/restauración de contexto (CSRP) de direcciones efectivas (EA)
4	Un ID de proceso (PID) y un ID de subproceso (TID) opcional
5	Un puntero de registro de utilización de acelerador (AURP) de direcciones virtuales (VA)
6	La dirección virtual del puntero a tabla de segmentos de almacenamiento (SSTP)
7	Un número de servicio de interrupción lógica (LISN)
8	Tabla de vectores de interrupción, obtenida a partir de los parámetros de llamada al hipervisor
9	Un valor de registro de estado (SR)
10	Un ID de partición lógica (LPID)
11	Un puntero a registro de utilización de acelerador de hipervisor de direcciones reales (RA)
12	El registro de descriptor de almacenamiento (SDR)

En una forma de realización, el hipervisor inicializa una pluralidad de registros 445 de sector de integración de acelerador 490.

Tal como se ilustra en la FIG. 4F, una forma de realización de la invención emplea una memoria unificada direccionable a través de un espacio de direcciones de memoria virtual común utilizado para acceder a las memorias de procesador físicas 401-402 y las memorias de GPU 420-423. En esta implementación, las operaciones ejecutadas en las GPU 410-413 utilizan el mismo espacio de direcciones virtuales/eficaces de memoria para acceder a las memorias de procesador 401-402 y viceversa, simplificando así la programación. En una forma de realización, una primera parte del espacio de direcciones virtuales/eficaces está asignada a la memoria de procesador 401, una segunda parte a la segunda memoria de procesador 402, una tercera parte a la memoria de GPU 420, y así sucesivamente. Todo el espacio de memoria virtual/eficaz (a veces denominado espacio de direcciones efectivas) se distribuye de este modo a través de cada una de las memorias de procesador 401-402 y las memorias de GPU 420-423, lo que permite que

cualquier procesador o GPU acceda a cualquier memoria física con una dirección virtual correlacionada con esa memoria.

En una forma de realización, los circuitos de gestión de sesgo/coherencia 494A-494E dentro de una o más de las MMU 439A-439E garantizan la coherencia de memoria caché entre las memorias caché de los procesadores de ordenador principal (por ejemplo, 405) y las GPU 410-413, e implementan técnicas de sesgado que indican las memorias físicas en las que se deben almacenar determinados tipos de datos. Si bien se ilustran múltiples instancias de circuitos de gestión de sesgo/coherencia 494A-494E en la FIG. 4F, los circuitos de sesgo/coherencia se pueden implementar dentro de la MMU de uno o más procesadores de ordenador principal 405 y/o dentro del circuito de integración de acelerador 436.

Una forma de realización permite que una memoria conectada a GPU 420-423 se correlacione como parte de la memoria del sistema y que se acceda a la misma mediante tecnología de memoria virtual compartida (SVM), pero sin experimentar los inconvenientes de rendimiento típicos asociados a la coherencia de memoria caché del sistema global. La capacidad de acceder a memoria conectada a GPU 420-423 como memoria de sistema sin sobrecarga onerosa de coherencia de memoria caché proporciona un entorno operativo beneficioso para la descarga hacia una GPU. Esta disposición permite que el software de procesador de ordenador principal 405 configure operandos y acceda a resultados de cálculo, sin la sobrecarga de copias de datos de DMA de E/S habituales. Tales copias tradicionales implican llamadas al controlador, interrupciones y accesos de E/S mapeada en memoria (MMIO), que son todos ineficaces en relación con simples accesos a memoria. Al mismo tiempo, la capacidad de acceder a memoria conectada a GPU 420-423 sin sobrecargas de coherencia de memoria caché puede ser crítica en el tiempo de ejecución de un cálculo descargado. En caso de tráfico de memoria sustancial de escritura de transmisión continua, por ejemplo, la sobrecarga de coherencia de memoria caché puede reducir significativamente el ancho de banda de escritura efectivo visto por una GPU 410-413. La eficacia de la configuración de los operandos, la eficacia del acceso a los resultados y la eficiencia del cálculo de la GPU juegan un papel en la determinación de la efectividad de la descarga hacia la GPU.

En una implementación, la selección entre el sesgo de GPU y el sesgo de procesador de ordenador principal es determinada por una estructura de datos de rastreador de sesgo. Se puede utilizar una tabla de sesgo, por ejemplo, que puede ser una estructura granular de página (es decir, controlada en la granularidad de una página de memoria) que incluye 1 o 2 bits por página de memoria conectada a GPU. La tabla de sesgo puede implementarse en un intervalo de memoria robado de una o más memorias conectadas a GPU 420-423, con o sin una memoria caché de sesgo en la GPU 410-413 (por ejemplo, para almacenar en caché entradas utilizadas con frecuencia/recientemente de la tabla de sesgo). De forma alternativa, toda la tabla de sesgo se puede mantener dentro de la GPU.

En una implementación, se accede a la entrada de tabla de sesgo asociada a cada acceso a la memoria conectada a GPU 420-423 antes del acceso real a la memoria de GPU, lo que provoca las siguientes operaciones. En primer lugar, las solicitudes locales de la GPU 410-413 que encuentran su página en el sesgo de GPU se reenvían directamente a una memoria de GPU correspondiente 420-423. Las solicitudes locales de la GPU que encuentran su página en el sesgo de ordenador principal se reenvían al procesador 405 (por ejemplo, a través de un enlace de alta velocidad como el analizado anteriormente). En una forma de realización, las solicitudes del procesador 405 que encuentran la página solicitada en el sesgo de procesador de ordenador principal completan la solicitud como una lectura de memoria normal. De forma alternativa, las solicitudes dirigidas a una página sesgada por GPU se pueden reenviar a la GPU 410-413. La GPU puede hacer entonces que la página pase a un sesgo de procesador de ordenador principal si no está utilizando actualmente la página.

El estado de sesgo de una página puede cambiarse mediante un mecanismo basado en software, un mecanismo basado en software asistido por hardware o, para un conjunto limitado de casos, un mecanismo basado puramente en hardware.

Un mecanismo para cambiar el estado de sesgo emplea una llamada a API (por ejemplo, OpenCL), que, a su vez, llama al controlador de dispositivo de la GPU que, a su vez, envía un mensaje (o pone en cola un descriptor de comando) a la GPU indicándole que cambie el estado de sesgo y, para algunas transiciones, que realice una operación de vaciado de memoria caché en el ordenador principal. La operación de vaciado de memoria caché es necesaria para una transición del sesgo de procesador de ordenador principal 405 al sesgo de GPU, pero no es necesaria para la transición opuesta.

En una forma de realización, la coherencia de memoria caché se mantiene haciendo temporalmente que las páginas sesgadas por GPU no puedan almacenarse en memoria caché mediante el procesador de ordenador principal 405. Para acceder a estas páginas, el procesador 405 puede solicitar acceso desde la GPU 410, que puede, o no, otorgar acceso de inmediato, dependiendo de la implementación. Por lo tanto, para reducir la comunicación entre el procesador 405 y la GPU 410, es beneficioso garantizar que las páginas sesgadas por GPU sean las que requiere la GPU pero no el procesador de ordenador principal 405, y viceversa.

Procesamiento en cadena de gráficos

La FIG. 5 ilustra un procesamiento en cadena de gráficos 500, de acuerdo con una forma de realización. En una forma de realización, un procesador de gráficos puede implementar el procesamiento en cadena de gráficos 500 ilustrado. El procesador de gráficos puede estar incluir dentro de los subsistemas de procesamiento paralelo descritos en el presente documento, tal como el procesador paralelo 200 de la FIG. 2, que, en una forma de realización, es una variante del/de los procesador(es) paralelo(s) 112 de la FIG. 1. Los diversos sistemas de procesamiento paralelo pueden implementar el procesamiento en cadena de gráficos 500 a través de una o más instancias de la unidad de procesamiento paralelo (por ejemplo, la unidad de procesamiento paralelo 202 de la FIG. 2) como se describe en el presente documento. Por ejemplo, una unidad de sombreado (por ejemplo, el multiprocesador de gráficos 234 de la FIG. 3) puede estar configurada para realizar las funciones de una o más de una unidad de procesamiento de vértices 504, una unidad de procesamiento de control de teselación 508, una unidad de procesamiento de evaluación de teselación 512, una unidad de procesamiento de geometría 516 y una unidad de procesamiento de fragmentos/píxeles 524. Las funciones de ensamblador de datos 502, de ensambladores de primitivas 506, 514, 518, de unidad de teselación 510, de rasterizador 522 y de unidad de operaciones de rasterización 526 también se pueden realizar mediante otros motores de procesamiento dentro de una agrupación de procesamiento (por ejemplo, agrupación de procesamiento 214 de la FIG. 3) y una unidad de partición correspondiente (por ejemplo, la unidad de partición 220A-220N de la FIG. 2). El procesamiento en cadena de gráficos 500 también puede implementarse utilizando unidades de procesamiento dedicadas para una o más funciones. En una forma de realización, una o más partes del procesamiento en cadena de gráficos 500 se pueden realizar mediante lógica de procesamiento paralelo dentro de un procesador de propósito general (por ejemplo, una CPU). En una forma de realización, una o más partes del procesamiento en cadena de gráficos 500 pueden acceder a memoria en chip (por ejemplo, la memoria de procesador paralelo 222 de la FIG. 2) a través de una interfaz de memoria 528, que puede ser una instancia de la interfaz de memoria 218 de la FIG. 2.

En una forma de realización, el ensamblador de datos 502 es una unidad de procesamiento que recopila datos de vértices para superficies y primitivas. A continuación, el ensamblador de datos 502 proporciona los datos de vértice, incluidos los atributos de vértice, a la unidad de procesamiento de vértices 504. La unidad de procesamiento de vértices 504 es una unidad de ejecución programable que ejecuta programas de sombreado de vértices, iluminando y transformando datos de vértices como se especifica por los programas de sombreado de vértices. La unidad de procesamiento de vértices 504 lee datos que se almacenan en memoria caché, local o de sistema para su uso en el procesamiento de los datos de vértices y se puede programar para transformar los datos de vértices desde una representación de coordenadas basada en objetos a un espacio de coordenadas de espacio mundial o un espacio de coordenadas de dispositivo normalizado.

Una primera instancia de un ensamblador de primitivas 506 recibe atributos de vértice desde la unidad de procesamiento de vértices 50. Las lecturas del ensamblador de primitivas 506 almacenan atributos de vértice según sea necesario y generan primitivas gráficas para su procesamiento mediante la unidad de procesamiento de control de teselación 508. Las primitivas gráficas incluyen triángulos, segmentos de línea, puntos, parches, etc., según lo admitido por varias interfaces de programación de aplicaciones (API) de procesamiento gráfico.

La unidad de procesamiento de control de teselación 508 trata los vértices de entrada como puntos de control para un parche geométrico. Los puntos de control se transforman desde una representación de datos de entrada a partir del parche (por ejemplo, las bases del parche) a una representación que sea adecuada para su uso en la evaluación de superficies por parte de la unidad de procesamiento de evaluación de teselación 512. La unidad de procesamiento de control de teselación 508 también puede calcular factores de teselación para bordes de parches geométricos. Un factor de teselación se aplica a un solo borde y cuantifica un nivel de detalle dependiente de vista asociado al borde. Una unidad de teselación 510 está configurada para recibir los factores de teselación para los bordes de un parche y para teselar el parche en múltiples primitivas geométricas tales como primitivas de línea, triángulo o cuadrilátero, que se transmiten a una unidad de procesamiento de evaluación de teselación 512. La unidad de procesamiento de evaluación de teselación 512 realiza operaciones en coordenadas parametrizadas del parche subdividido para generar una representación de superficie y atributos de vértice para cada vértice asociado a las primitivas geométricas.

Una segunda instancia de un ensamblador de primitivas 514 recibe atributos de vértice desde la unidad de procesamiento de evaluación de teselación 512, lee atributos de vértice almacenados según sea necesario y genera primitivas gráficas para su procesamiento mediante la unidad de procesamiento de geometría 516. La unidad de procesamiento de geometría 516 es una unidad de ejecución programable que ejecuta programas de sombreado de geometría para transformar primitivas de gráficos recibidas desde el ensamblador de primitivas 514 como se especifica por los programas de sombreado de geometría. En una forma de realización, la unidad de procesamiento de geometría 516 está programada para subdividir las primitivas gráficas en una o más primitivas gráficas nuevas y calcular parámetros utilizados para rasterizar las nuevas primitivas gráficas.

En algunas formas de realización, la unidad de procesamiento de geometría 516 puede añadir o eliminar elementos en la secuencia de geometría. La unidad de procesamiento de geometría 516 proporciona al ensamblador de primitivas 518 los parámetros y vértices que especifican nuevas primitivas gráficas. El ensamblador de primitivas 518 recibe los

parámetros y vértices desde la unidad de procesamiento de geometría 516 y genera primitivas gráficas para su procesamiento mediante una unidad de escalado, selección y recorte de ventana gráfica 520. La unidad de procesamiento de geometría 516 lee datos almacenados en memoria de procesador paralelo o memoria de sistema para su uso en el procesamiento de los datos de geometría. La unidad de escalado, selección y recorte de ventana gráfica 520 realiza el recorte, la selección y el escalado de ventana gráfica y proporciona primitivas gráficas procesadas a un rasterizador 522.

El rasterizador 522 puede realizar una selección de profundidad y otras optimizaciones basadas en profundidad. El rasterizador 522 también realiza una conversión por barrido en las nuevas primitivas gráficas para generar fragmentos y proporcionar esos fragmentos y datos de cobertura asociados a la unidad de procesamiento de fragmentos/píxeles 524. La unidad de procesamiento de fragmentos/píxeles 524 es una unidad de ejecución programable que está configurada para ejecutar programas de sombreado de fragmentos o programas de sombreado de píxeles. La unidad de procesamiento de fragmentos/píxeles 524 transforma fragmentos o píxeles recibidos desde el rasterizador 522, como se especifica por los programas de sombreado de fragmentos o píxeles. Por ejemplo, la unidad de procesamiento de fragmentos/píxeles 524 se puede programar para realizar operaciones que incluyen, pero sin limitarse a, correlación de textura, sombreado, mezcla, corrección de textura y corrección de perspectiva para producir fragmentos o píxeles sombreados que se proporcionan a una unidad de operaciones de rasterización 526. La unidad de procesamiento de fragmentos/píxeles 524 puede leer datos almacenados en la memoria de procesador paralelo o la memoria de sistema para su uso cuando se procesan los datos de fragmentos. Los programas de sombreado de fragmentos o píxeles pueden configurarse para sombrear en granularidades de muestra, píxel, mosaico u otras granularidades, dependiendo de la frecuencia de muestreo configurada para las unidades de procesamiento.

La unidad de operaciones de rasterización 526 es una unidad de procesamiento que realiza operaciones de rasterización que incluyen, pero sin limitarse a, estarcido, prueba Z, mezcla y similares, y proporciona datos de píxel como datos de gráficos procesados que se almacenarán en una memoria de gráficos (por ejemplo, la memoria de procesador paralelo 222 de la FIG. 2 y/o la memoria de sistema 104 de la FIG. 1), y se mostrarán en el uno o más dispositivos de visualización 110 o para un procesamiento adicional por un procesador del uno o más procesadores 102 o procesadores paralelos 112. En algunas formas de realización, la unidad de operaciones de rasterización 526 está configurada para comprimir datos Z o de color que se escriben en memoria y descomprimir datos Z o de color que se leen de la memoria.

Visión general del aprendizaje automático

Un algoritmo de aprendizaje automático es un algoritmo que puede aprender basándose en un conjunto de datos. Las formas de realización de los algoritmos de aprendizaje automático se pueden diseñar para modelar abstracciones de alto nivel dentro de un conjunto de datos. Por ejemplo, se pueden utilizar algoritmos de reconocimiento de imágenes para determinar a cuál de diversas categorías pertenece una entrada dada; los algoritmos de regresión pueden proporcionar un valor numérico dada una entrada; y se pueden utilizar algoritmos de reconocimiento de patrones para generar texto traducido o realizar reconocimiento de texto a voz y/o habla.

Un tipo ejemplar de algoritmo de aprendizaje automático es una red neuronal. Hay muchos tipos de redes neuronales; un tipo simple de red neuronal es una red de alimentación directa. Una red de alimentación directa puede implementarse como un gráfico acíclico en el que los nodos están dispuestos en capas. Típicamente, una topología de red anticipada incluye una capa de entrada y una capa de salida que están separadas por al menos una capa oculta. La capa oculta transforma la entrada recibida por la capa de entrada en una representación que es útil para generar una salida en la capa de salida. Los nodos de red están completamente conectados a través de los bordes a los nodos en capas adyacentes, pero no hay bordes entre los nodos dentro de cada capa. Los datos recibidos en los nodos de una capa de entrada de una red de alimentación directa se propagan (es decir, "se reenvían") a los nodos de la capa de salida a través de una función de activación que calcula los estados de los nodos de cada capa sucesiva en la red en función de coeficientes ("pesos") asociados respectivamente a cada uno de los bordes que conectan las capas. Dependiendo del modelo específico representado por el algoritmo que se está ejecutando, la salida del algoritmo de red neuronal puede adoptar varias formas.

Antes de que se pueda utilizar un algoritmo de aprendizaje automático para modelar un problema particular, el algoritmo se entrena utilizando un conjunto de datos de entrenamiento. El entrenamiento de una red neuronal implica seleccionar una topología de red, usar un conjunto de datos de entrenamiento que representan un problema que está siendo modelado por la red y ajustar los pesos hasta que el modelo de red funcione con un error mínimo para todas las instancias del conjunto de datos de entrenamiento. Por ejemplo, durante un proceso de entrenamiento de aprendizaje supervisado para una red neuronal, la salida producida por la red en respuesta a la entrada que representa una instancia en un conjunto de datos de entrenamiento se compara con la salida etiquetada como "correcta" para esa instancia, se calcula una señal de error que representa la diferencia entre la salida y la salida etiquetada, y los pesos asociados a las conexiones se ajustan para minimizar ese error a medida que la señal de error se propaga hacia atrás a través de las capas de la red. La red se considera “entrenada” cuando se minimizan los errores para cada una de las salidas generadas a partir de las instancias del conjunto de datos de entrenamiento.

La precisión de un algoritmo de aprendizaje automático puede verse afectada significativamente por la calidad del conjunto de datos utilizado para entrenar el algoritmo. El proceso de entrenamiento puede ser computacionalmente intenso y puede requerir una cantidad significativa de tiempo en un procesador convencional de propósito general. Por consiguiente, el hardware de procesamiento paralelo se utiliza para entrenar muchos tipos de algoritmos de aprendizaje automático. Esto es particularmente útil para optimizar el entrenamiento de redes neuronales, ya que los cálculos realizados para ajustar los coeficientes en redes neuronales se prestan naturalmente a implementaciones paralelas. Específicamente, muchos algoritmos de aprendizaje automático y aplicaciones de software se han adaptado para hacer uso del hardware de procesamiento paralelo dentro de dispositivos de procesamiento de gráficos de propósito general.

La FIG. 6 es un diagrama generalizado de una pila de software de aprendizaje automático 600. Una aplicación de aprendizaje automático 602 puede estar configurada para entrenar una red neuronal utilizando un conjunto de datos de entrenamiento o para usar una red neuronal profunda entrenada para implementar inteligencia de máquina. La aplicación de aprendizaje automático 602 puede incluir funcionalidad de entrenamiento e inferencia para una red neuronal y/o software especializado que se puede utilizar para entrenar una red neuronal antes de la implantación. La aplicación de aprendizaje automático 602 puede implementar cualquier tipo de inteligencia de máquina que incluye, pero sin limitarse a, reconocimiento de imágenes, correlación y localización, navegación autónoma, síntesis de voz, formación de imágenes médicas o traducción de idiomas.

La aceleración de hardware para la aplicación de aprendizaje automático 602 puede habilitarse a través de un marco estructural de aprendizaje automático 604. El marco estructural de aprendizaje automático 604 puede proporcionar una biblioteca de primitivas de aprendizaje automático. Las primitivas de aprendizaje automático son operaciones básicas realizadas comúnmente por algoritmos de aprendizaje automático. Sin el marco estructural de aprendizaje automático 604, los desarrolladores de algoritmos de aprendizaje automático tendrían que crear y optimizar la lógica computacional principal asociada al algoritmo de aprendizaje automático, y luego volver a optimizar la lógica computacional a medida que se desarrollen nuevos procesadores paralelos. En cambio, la aplicación de aprendizaje automático puede configurarse para realizar los cálculos necesarios utilizando las primitivas proporcionadas por el marco estructural de aprendizaje automático 604. Primitivas ejemplares incluyen convoluciones de tensor, funciones de activación y agrupamiento, que son operaciones computacionales que se realizan mientras se entrena una red neuronal convolucional (CNN). El marco estructural de aprendizaje automático 604 también puede proporcionar primitivas para implementar subprogramas de álgebra lineal básicos realizados por muchos algoritmos de aprendizaje automático, tales como operaciones matriciales y vectoriales.

El marco estructural de aprendizaje automático 604 puede procesar los datos de entrada recibidos desde la aplicación de aprendizaje automático 602 y generar los datos de entrada adecuados para un marco estructural informático 606. El marco estructural informático 606 puede abstraer las instrucciones subyacentes proporcionadas al controlador de GPGPU 608 para permitir que el marco estructural de aprendizaje automático 604 aproveche la aceleración de hardware a través del hardware de GPGPU 610 sin requerir que el marco estructural de aprendizaje automático 604 tenga un conocimiento exhaustivo de la arquitectura del hardware de GPGPU 610. Además, el marco estructural informático 606 puede permitir la aceleración de hardware para el marco estructural de aprendizaje automático 604 a través de una variedad de tipos y generaciones del hardware de GPGPU 610.

Aceleración del aprendizaje automático GPGPU

La FIG. 7 ilustra una unidad de procesamiento de gráficos de propósito general altamente paralela 700, de acuerdo con una forma de realización. En una forma de realización, la unidad de procesamiento de propósito general (GPGPU) 700 puede estar configurada para ser particularmente eficaz en el procesamiento del tipo de cargas de trabajo computacionales asociadas al entrenamiento de redes neuronales profundas. Además, la GPGPU 700 se puede vincular directamente a otras instancias de la GPGPU para crear una agrupación de múltiples GPU para mejorar la velocidad de entrenamiento para redes neuronales particularmente profundas.

La GPGPU 700 incluye una interfaz de ordenador principal 702 para permitir una conexión con un procesador de ordenador principal. En una forma de realización, la interfaz de ordenador principal 702 es una interfaz PCI Express. Sin embargo, la interfaz de ordenador principal también puede ser una interfaz de comunicaciones específica del proveedor o una estructura de comunicaciones. La GPGPU 700 recibe comandos del procesador de ordenador principal y utiliza un planificador global 704 para distribuir subprocesos de ejecución asociados a esos comandos a un conjunto de agrupaciones de cálculo 706A-706H. Las agrupaciones de cálculo 706A-706H comparten una memoria caché 708. La memoria caché 708 puede servir como memoria caché de nivel superior para las memorias caché dentro de las agrupaciones de cálculo 706A-706H.

La GPGPU 700 incluye memoria 714A-714B acoplada a las agrupaciones de cálculo 706A-H a través de un conjunto de controladores de memoria 712A-712B. En varias formas de realización, la memoria 714A-714B puede incluir varios tipos de dispositivos de memoria que incluyen memoria de acceso aleatorio dinámica (DRAM) o memoria de acceso aleatorio de gráficos, tal como memoria de acceso aleatorio de gráficos síncrona (SGRAM), que incluye memoria de doble velocidad de datos de gráficos (GDDR) o memoria apilada 3D, que incluye, pero sin limitarse a, memoria de alto ancho de banda (HBM).

En una forma de realización, cada agrupación de cálculo 706A-706H incluye un conjunto de multiprocesadores de gráficos, tal como el multiprocesador de gráficos 400 de la FIG. 4A. Los multiprocesadores de gráficos de la agrupación de cálculo tienen múltiples tipos de unidades lógicas de números enteros y de coma flotante que pueden realizar operaciones computacionales en un rango de precisiones, incluidas las adecuadas para cálculos de aprendizaje automático. Por ejemplo, y en una forma de realización, al menos un subconjunto de las unidades de coma flotante en cada una de las agrupaciones de cálculo 706A-706H puede configurarse para realizar operaciones de coma flotante de 16 bits o 32 bits, mientras que un subconjunto diferente de las unidades de coma flotante puede configurarse para realizar operaciones de coma flotante de 64 bits.

Se pueden configurar múltiples instancias de la GPGPU 700 para que funcionen como una agrupación de cálculo. El mecanismo de comunicación utilizado por la agrupación de cálculo para la sincronización y el intercambio de datos varía según las formas de realización. En una forma de realización, las múltiples instancias de la GPGPU 700 se comunican a través de la interfaz de ordenador principal 702. En una forma de realización, la GPGPU 700 incluye un concentrador de E/S 709 que acopla la GPGPU 700 a un enlace de GPU 710 que permite una conexión directa a otras instancias de la GPGPU. En una forma de realización, el enlace de GPU 710 está acoplado a un puente GPU a GPU dedicado que permite la comunicación y sincronización entre múltiples instancias de la GPGPU 700. En una forma de realización, el enlace de GPU 710 está acoplado a una interconexión de alta velocidad para transmitir y recibir datos a otras GPGPU o procesadores paralelos. En una forma de realización, las múltiples instancias de la GPGPU 700 están ubicadas en distintos sistemas de procesamiento de datos y se comunican a través de un dispositivo de red al que se puede acceder a través de la interfaz de ordenador principal 702. En una forma de realización, el enlace de GPU 710 puede estar configurado para permitir una conexión a un procesador de ordenador principal además de o como una alternativa a la interfaz de ordenador principal 702.

Si bien la configuración ilustrada de la GPGPU 700 se puede configurar para entrenar redes neuronales, una forma de realización proporciona una configuración alternativa de la GPGPU 700 que se puede configurar para implantarse en una plataforma de inferencia de alto rendimiento o baja potencia. En una configuración de inferencia, la GPGPU 700 incluye menos agrupaciones de cálculo 706A-706H con respecto a la configuración de entrenamiento. Además, la tecnología de memorias asociada a la memoria 714A-714B puede diferir entre las configuraciones de inferencia y de entrenamiento. En una forma de realización, la configuración de inferencia de la GPGPU 700 puede admitir instrucciones específicas de inferencia. Por ejemplo, una configuración de inferencia puede proporcionar soporte para una o más instrucciones de productos escalar de números enteros de 8 bits, que se utilizan comúnmente durante las operaciones de inferencia para redes neuronales implantadas.

La FIG. 8 ilustra un sistema informático de múltiples GPU 800, de acuerdo con una forma de realización. El sistema informático de múltiples GPU 800 puede incluir un procesador 802 acoplado a múltiples GPGPU 806A-806D a través de un conmutador de interfaz de ordenador principal 804. El conmutador de interfaz de ordenador principal 804, en una forma de realización, es un dispositivo de conmutación PCI Express que acopla el procesador 802 a un bus PCI Express a través del cual el procesador 802 puede comunicarse con el conjunto de GPGPU 806A-806D. Cada una de las múltiples GPGPU 806A-806D puede ser una instancia de la GPGPU 700 de la FIG. 7. Las GPGPU 806A-806D pueden interconectarse a través de un conjunto de enlaces de GPU a GPU de punto a punto de alta velocidad 816. Los enlaces de GPU a GPU de ata velocidad pueden conectarse a cada una de las GPGPU 806A-806D a través de un enlace de GPU dedicado, tal como el enlace de GPU 710 de la FIG. 7. Los enlaces de GPU P2P 816 permiten la comunicación directa entre cada una de las GPGPU 806A-806D sin requerir comunicación a través del bus de interfaz de ordenador principal al que está conectado el procesador 802. Con el tráfico de GPU a GPU dirigido a los enlaces de GPU P2P, el bus de interfaz de ordenador principal permanece disponible para el acceso a memoria del sistema o para comunicarse con otras instancias del sistema informático de múltiples GPU 800, por ejemplo, a través de uno o más dispositivos de red. Mientras que en la forma de realización ilustrada las GPGPU 806A-806D están conectadas al procesador 802 a través del conmutador de interfaz de ordenador principal 804, en una forma de realización el procesador 802 incluye soporte directo para los enlaces de GPU P2P 816 y puede conectarse directamente a las GPGPU 806A-806D.

Implementaciones de redes neuronales de aprendizaje automático

La arquitectura informática proporcionada por las formas de realización descritas en el presente documento puede configurarse para realizar los tipos de procesamiento paralelo que son particularmente adecuados para entrenar e implantar redes neuronales para el aprendizaje automático. Una red neuronal se puede generalizar como una red de funciones que tienen una relación gráfica. Como es bien conocido en la técnica, hay una variedad de tipos de implementaciones de redes neuronales utilizadas en el aprendizaje automático. Un tipo ejemplar de red neuronal es la red de alimentación directa, como se describió anteriormente.

Un segundo tipo ejemplar de red neuronal es la red neuronal convolucional (CNN). Una CNN es una red neuronal de alimentación directa especializada para procesar datos que tienen una topología conocida, similar a una cuadrícula, tal como datos de imágenes. Por consiguiente, las CNN se utilizan comúnmente para aplicaciones de visión computarizada y reconocimiento de imágenes, pero también se pueden utilizar para otros tipos de reconocimiento de patrones, tales como procesamiento de voz y lenguaje. Los nodos en la capa de entrada de CNN están organizados

en un conjunto de "filtros" (detectores de características inspirados en los campos receptivos que se encuentran en la retina), y la salida de cada conjunto de filtros se propaga a nodos de capas sucesivas de la red. Los cálculos de una CNN incluyen aplicar la operación matemática de convolución a cada filtro para producir la salida de ese filtro. Convolución es un tipo especializado de operación matemática realizada por dos funciones para producir una tercera función que es una versión modificada de una de las dos funciones originales. En terminología de red convolucional, la primera función de la convolución puede denominarse entrada, mientras que la segunda función puede denominarse núcleo (*kernel*) de convolución. La salida puede denominarse mapa de características. Por ejemplo, los datos de entrada para una capa de convolución puede ser una matriz multidimensional de datos que define los diversos componentes de color de una imagen de entrada. El núcleo de convolución puede ser una matriz multidimensional de parámetros, donde los parámetros se adaptan mediante el proceso de entrenamiento para la red neuronal.

Las redes neuronales recurrentes (RNN) son una familia de redes neuronales de alimentación directa que incluyen conexiones de retroalimentación entre capas. Las RNN permiten el modelado de datos secuenciales al compartir datos de parámetros en diferentes partes de la red neuronal. La arquitectura para una RNN incluye ciclos. Los ciclos representan la influencia de un valor actual de una variable sobre su propio valor en un momento futuro, ya que al menos una parte de los datos de salida de la RNN se utiliza como retroalimentación para procesar una entrada posterior en una secuencia. Esta característica hace que las RNN sean particularmente útiles para el procesamiento del lenguaje debido a la naturaleza variable en la que pueden estar compuestos los datos del lenguaje.

Las figuras descritas a continuación presentan redes ejemplares de alimentación directa, CNN y RNN, y también describen un proceso general para entrenar e implantar respectivamente cada uno de esos tipos de redes. Se entenderá que estas descripciones son ejemplares y no limitantes en cuanto a cualquier forma de realización específica descrita en el presente documento y que los conceptos ilustrados se pueden aplicar generalmente a redes neuronales profundas y técnicas de aprendizaje automático en general.

Las redes neuronales ejemplares descritas anteriormente se pueden utilizar para realizar un aprendizaje profundo. El aprendizaje profundo es aprendizaje automático mediante el uso de redes neuronales profundas. Las redes neuronales profundas utilizadas en el aprendizaje profundo son redes neuronales artificiales compuestas de múltiples capas ocultas, a diferencia de las redes neuronales superficiales que incluyen solo una capa oculta. Las redes neuronales más profundas suelen ser más intensivas desde el punto de vista computacional para su entrenamiento. Sin embargo, las capas ocultas adicionales de la red permiten el reconocimiento de patrones de varias etapas que da como resultado un error de salida reducido en relación con las técnicas de aprendizaje automático superficial.

Las redes neuronales profundas utilizadas en el aprendizaje profundo incluyen típicamente una red de sección de entrada (*front-end*) para realizar el reconocimiento de características acoplada a una red de sección de procesamiento (*back-end*) que representa un modelo matemático que puede realizar operaciones (por ejemplo, clasificación de objetos, reconocimiento de voz, etc.) en función de la representación de características proporcionada al modelo. El aprendizaje profundo permite llevar a cabo el aprendizaje automático sin necesidad de realizar una ingeniería de características manuales para el modelo. En cambio, las redes neuronales profundas pueden aprender características basadas en la estructura estadística o la correlación dentro de los datos de entrada. Las características aprendidas se pueden proporcionar a un modelo matemático que puede correlacionar las características detectadas con una salida. El modelo matemático utilizado por la red está generalmente especializado para la tarea específica a realizar, y se utilizarán diferentes modelos para realizar diferentes tareas.

Una vez estructurada la red neuronal, se puede aplicar un modelo de aprendizaje a la red para entrenar la red para que realice tareas específicas. El modelo de aprendizaje describe cómo ajustar los pesos dentro del modelo para reducir el error de salida de la red. La retropropagación de errores es un procedimiento común utilizado para entrenar redes neuronales. Un vector de entrada se presenta a la red para su procesamiento. La salida de la red se compara con la salida deseada utilizando una función de pérdida y se calcula un valor de error para cada una de las neuronas en la capa de salida. Los valores de error se propagan entonces hacia atrás hasta que cada neurona tenga un valor de error asociado que represente aproximadamente su contribución a la salida original. La red puede entonces aprender de esos errores utilizando un algoritmo, tal como el algoritmo de descenso de gradiente estocástico, para actualizar los pesos de la de la red neuronal.

Las FIGS. 9A-9B ilustran una red neuronal convolucional ejemplar. La FIG. 9A ilustra varias capas dentro de una CNN. Como se muestra en la FIG. 9A, una CNN ejemplar utilizada para modelar el procesamiento de imágenes puede recibir una entrada 902 que describe las componentes rojo, verde y azul (RGB) de una imagen de entrada. La entrada 902 se puede procesar mediante múltiples capas convolucionales (por ejemplo, capa convolucional 904, capa convolucional 906). La salida de las múltiples capas convolucionales puede procesarse opcionalmente mediante un conjunto de capas completamente conectadas 908. Las neuronas de una capa completamente conectada tienen conexiones completas con todas las activaciones en la capa anterior, como se describió anteriormente para una red de alimentación directa. La salida de las capas completamente conectadas 908 se puede utilizar para generar un resultado de salida de la red. Las activaciones dentro de las capas completamente conectadas 908 pueden calcularse mediante el uso de multiplicación matricial en lugar de convolución. No todas las implementaciones de CNN utilizan capas completamente conectadas 908. Por ejemplo, en algunas implementaciones, la capa convolucional 906 puede generar una salida para la CNN.

Las capas convolucionales apenas están conectadas, lo que difiere de la configuración de red neuronal tradicional encontrada en las capas completamente conectadas 908. Las capas de red neuronal tradicionales están completamente conectadas, de modo que cada unidad de salida interactúa con cada unidad de entrada. Sin embargo, las capas convolucionales apenas están conectadas porque la salida de la convolución de un campo (en lugar del valor de estado respectivo de cada uno de los nodos del campo) se introduce en los nodos de la capa posterior, como se ilustra. Los núcleos asociados a las capas convolucionales realizan operaciones de convolución, cuya salida se envía a la siguiente capa. La reducción de dimensionalidad realizada dentro de las capas convolucionales es un aspecto que permite a la CNN escalar para procesar imágenes grandes.

La FIG. 9B ilustra fases de cálculo ejemplares dentro de una capa convolucional de una CNN. Los datos de entrada para una capa convolucional 912 de una CNN se puede procesar en tres fases de una capa convolucional 914. Las tres fases pueden incluir una fase de convolución 916, una fase de detección 918 y una fase de agrupamiento 920. La capa de convolución 914 puede entonces proporcionar datos a una capa convolucional sucesiva. La capa convolucional final de la red puede generar datos de mapa de características de salida o proporcionar datos de entrada a una capa completamente conectada, por ejemplo para generar un valor de clasificación para los datos de entrada para la CNN.

En la fase de convolución 916 se realizan varias convoluciones en paralelo para producir un conjunto de activaciones lineales. La fase de convolución 916 puede incluir una transformación afín, que es cualquier transformación que se puede especificar como una transformación lineal más una conversión. Las transformaciones afines incluyen rotaciones, traslaciones, escalado y combinaciones de estas transformaciones. La fase de convolución calcula la salida de funciones (por ejemplo, neuronas) que están conectadas a regiones específicas en la entrada, que se puede determinar como la región local asociada a la neurona. Las neuronas calculan un producto escalar entre los pesos de las neuronas y la región en la entrada local a la que están conectadas las neuronas. La salida de la fase de convolución 916 define un conjunto de activaciones lineales que se procesan mediante fases sucesivas de la capa convolucional 914.

Las activaciones lineales se pueden procesar mediante una fase de detección 918. En la fase de detección 918, cada activación lineal se procesa mediante una función de activación no lineal. La función de activación no lineal aumenta las propiedades no lineales de la red global sin afectar a los campos receptivos de la capa de convolución. Se pueden usar varios tipos de funciones de activación no lineal. Un tipo particular es la unidad lineal rectificada (ReLU), que utiliza una función de activación definida como $f(x) = \max(0, x)$, de modo que el umbral de activación es cero.

La fase de agrupamiento 920 utiliza una función de agrupamiento que reemplaza la salida de la capa convolucional 906 por una estadística de resumen de las salidas cercanas. La función de agrupamiento se puede utilizar para introducir una invariancia de conversión en la red neuronal, de modo que pequeñas conversiones en la entrada no cambien las salidas agrupadas. La invariancia a una conversión local puede ser útil en escenarios en los que la presencia de una característica en los datos de entrada sea más importante que la ubicación precisa de la característica. Se pueden utilizar varios tipos de funciones de agrupamiento durante la fase de agrupamiento 920, que incluyen agrupamiento máximo, agrupamiento promedio y agrupamiento de 12 normas. Además, algunas implementaciones de CNN no incluyen una fase de agrupamiento. En cambio, dichas implementaciones sustituyen a una fase de convolución adicional que tiene un mayor espaciamiento con respecto a las fases de convolución anteriores.

La salida de la capa convolucional 914 puede procesarse entonces mediante la siguiente capa 922. La siguiente capa 922 puede ser una capa convolucional adicional o una de las capas completamente conectadas 908. Por ejemplo, la primera capa convolucional 904 de la FIG. 9A puede proporcionar su salida a la segunda capa convolucional 906, mientras que la segunda capa convolucional puede proporcionar su salida a una primera capa de las capas completamente conectadas 908.

La FIG. 10 ilustra una red neuronal recurrente 1000 ejemplar. En una red neuronal recurrente (RNN), el estado anterior de la red influye en la salida del estado actual de la red. Las RNN se pueden construir de varias maneras usando una variedad de funciones. En general, el uso de RNN gira en torno al uso de modelos matemáticos para predecir el futuro en función de una secuencia previa de entradas. Por ejemplo, una RNN se puede utilizar para realizar un modelado estadístico del lenguaje para predecir una palabra próxima dada una secuencia previa de palabras. La RNN 1000 ilustrada puede describirse presentando una capa de entrada 1002 que recibe un vector de entrada, capas ocultas 1004 para implementar una función recurrente, un mecanismo de retroalimentación 1005 para habilitar una 'memoria' de estados anteriores y una capa de salida 1006 para proporcionar un resultado. La RNN 1000 funciona en base a instantes de tiempo. El estado de la RNN en un instante de tiempo dado se ve afectado en función del instante de tiempo anterior a través del mecanismo de retroalimentación 1005. En un instante de tiempo dado, el estado de las capas ocultas 1004 se define mediante el estado anterior y la entrada en el instante de tiempo actual. Una entrada inicial (x_1) en un primer instante de tiempo se puede procesar mediante la capa oculta 1004. La capa oculta 1004 puede procesar una segunda entrada (x_2) utilizando información de estado que se determina durante el procesamiento de la entrada inicial (x_1). Un estado dado puede calcularse como $s_t = f(Ux_t + Ws_{t-1})$, donde U y W son matrices de parámetros. La función f es generalmente no lineal, tal como la función tangencial hiperbólica (Tanh) o una variante

de la función rectificadora $f(x) = \max(0, x)$. Sin embargo, la función matemática específica utilizada en las capas ocultas 1004 puede variar dependiendo de los detalles de implementación específicos de la RNN 1000.

Además de las redes CNN y RNN básicas descritas, se pueden habilitar variaciones en esas redes. Un ejemplo de variante de RNN es la RNN de memoria a corto y largo plazo (LSTM). Las RNN de LSTM son capaces de aprender dependencias a largo plazo que pueden ser necesarias para procesar secuencias más largas de lenguaje. Una variante en la CNN es una red convolucional de creencia profunda, que tiene una estructura similar a una CNN y está entrenada de una manera similar a una red de creencia profunda. Una red de creencia profunda (DBN) es una red neuronal generativa que está compuesta por múltiples capas de variables estocásticas (aleatorias). Las DBN se pueden entrenar capa por capa usando aprendizaje codicioso no supervisado. Los pesos aprendidos de la DBN se pueden utilizar para proporcionar redes neuronales entrenadas previamente mediante la determinación de un conjunto inicial óptimo de pesos para la red neuronal.

La FIG. 11 ilustra el entrenamiento y la implantación de una red neuronal profunda. Una vez que una red dada se ha estructurado para una tarea, la red neuronal se entrena utilizando un conjunto de datos de entrenamiento 1102. Se han desarrollado diversos marcos estructurales de entrenamiento 1104 para permitir la aceleración de hardware del proceso de entrenamiento. Por ejemplo, el marco estructural de aprendizaje automático 604 de la FIG. 6 puede configurarse como un marco estructural de entrenamiento 604. El marco estructural de entrenamiento 604 puede conectarse a una red neuronal no entrenada 1106 y permitir el entrenamiento de la red neuronal no entrenada utilizando los recursos de procesamiento paralelo descritos en el presente documento para generar una red neuronal entrenada 1108.

Para comenzar el proceso de entrenamiento, los pesos iniciales pueden elegirse al azar o mediante entrenamiento previo utilizando una red de creencia profunda. El ciclo de entrenamiento se realizará de manera supervisada o no supervisada.

El aprendizaje supervisado es un procedimiento de aprendizaje en el que el entrenamiento se realiza como una operación mediada, tal como cuando el conjunto de datos de entrenamiento 1102 incluye una entrada emparejada con la salida deseada para la entrada, o donde el conjunto de datos de entrenamiento incluye una entrada que tiene una salida conocida y la salida de la red neuronal se califica manualmente. La red procesa las entradas y compara las salidas resultantes con un conjunto de salidas esperadas o deseadas. Los errores se propagan a través del sistema. El marco estructural de entrenamiento 1104 puede ajustarse para ajustar los pesos que controlan la red neuronal no entrenada 1106. El marco estructural de entrenamiento 1104 puede proporcionar herramientas para supervisar lo bien que la red neuronal no entrenada 1106 converge hacia un modelo adecuado para generar respuestas correctas en base a datos de entrada conocidos. El proceso de entrenamiento se produce repetidamente a medida que se ajustan los pesos de la red para refinar la salida generada por la red neuronal. El proceso de entrenamiento puede continuar hasta que la red neuronal alcance una precisión estadísticamente deseada asociada a una red neuronal entrenada 1108. La red neuronal entrenada 1108 se puede implantar para implementar cualquier número de operaciones de aprendizaje automático.

El aprendizaje no supervisado es un procedimiento de aprendizaje en el que la red intenta entrenarse utilizando datos no etiquetados. Por lo tanto, para el aprendizaje no supervisado, el conjunto de datos de entrenamiento 1102 incluirá datos de entrada sin ningún dato de salida asociado. La red neuronal no entrenada 1106 puede aprender agrupaciones dentro de la entrada no etiquetada y puede determinar cómo se relacionan las entradas individuales con el conjunto de datos global. El entrenamiento no supervisado se puede utilizar para generar un mapa autoorganizado, que es un tipo de red neuronal entrenada 1107 capaz de realizar operaciones útiles para reducir la dimensionalidad de los datos. El entrenamiento no supervisado también se puede utilizar para realizar la detección de anomalías, lo que permite la identificación de puntos de datos de un conjunto de datos de entrada que se desvían de los patrones normales de los datos.

También se pueden utilizar variaciones en el entrenamiento supervisado y no supervisado. El aprendizaje semisupervisado es una técnica en la que el conjunto de datos de entrenamiento 1102 incluye una mezcla de datos etiquetados y no etiquetados de la misma distribución. El aprendizaje incremental es una variante del aprendizaje supervisado en el que los datos de entrada se utilizan continuamente para entrenar aún más el modelo. El aprendizaje incremental permite que la red neuronal entrenada 1108 se adapte a los nuevos datos 1112 sin olvidar el conocimiento inculcado dentro de la red durante el entrenamiento inicial.

Ya sea supervisado o no supervisado, el proceso de entrenamiento para redes neuronales particularmente profundas puede ser demasiado intensivo desde el punto de vista computacional para un solo nodo de cálculo. En lugar de utilizar un solo nodo de cálculo, se puede utilizar una red distribuida de nodos de cálculo para acelerar el proceso de entrenamiento.

La FIG. 12 es un diagrama de bloques que ilustra un aprendizaje distribuido. El aprendizaje distribuido es un modelo de entrenamiento que utiliza múltiples nodos de cálculo distribuidos para realizar un entrenamiento supervisado o no supervisado de una red neuronal. Cada nodo de cálculo distribuido puede incluir uno o más procesadores de ordenador principal y uno o más de los nodos de procesamiento de propósito general, tales como la unidad de procesamiento de

gráficos de propósito general altamente paralela 700 de la FIG. 700. Como se ilustra, el aprendizaje distribuido se puede realizar mediante paralelismo de modelos 1202, paralelismo de datos 1204 o una combinación de paralelismo de modelos y de datos 1204.

En el paralelismo de modelos 1202, diferentes nodos de cálculo en un sistema distribuido pueden realizar cálculos de entrenamiento para diferentes partes de una única red. Por ejemplo, cada capa de una red neuronal puede ser entrenada por un nodo de procesamiento diferente del sistema distribuido. Los beneficios del paralelismo de modelos incluyen la capacidad de escalar a modelos particularmente grandes. La división de los cálculos asociados a diferentes capas de la red neuronal permite el entrenamiento de redes neuronales muy grandes en las que los pesos de todas las capas no cabrían en la memoria de un solo nodo computacional. En algunos casos, el paralelismo de modelos puede ser particularmente útil para realizar un entrenamiento no supervisado de grandes redes neuronales.

En el paralelismo de datos 1204, los diferentes nodos de la red distribuida tienen una instancia completa del modelo y cada nodo recibe una parte diferente de los datos. A continuación se combinan los resultados de los diferentes nodos. Aunque puede haber diferentes enfoques del paralelismo de datos, los enfoques de entrenamiento paralelo de datos requieren una técnica de combinación de resultados y de sincronización de los parámetros de modelo entre cada nodo. Enfoques ejemplares para combinar datos incluyen la promediación de parámetros y el paralelismo de datos basado en actualizaciones. La promediación de parámetros entrena cada nodo en un subconjunto de los datos de entrenamiento y establece los parámetros globales (por ejemplo, pesos, sesgos) con respecto al promedio de los parámetros de cada nodo. La promediación de parámetros utiliza un servidor de parámetros central que mantiene los datos de parámetros. El paralelismo de datos basado en actualizaciones es similar a la promediación de parámetros, excepto que en lugar de transferir parámetros desde los nodos al servidor de parámetros, se transfieren las actualizaciones del modelo. Además, el paralelismo de datos basado en actualizaciones se puede realizar de manera descentralizada, donde las actualizaciones se comprimen y transfieren entre nodos.

El modelo combinado y el paralelismo de datos 1206 se pueden implementar, por ejemplo, en un sistema distribuido en el que cada nodo de cálculo incluye múltiples GPU. Cada nodo puede tener una instancia completa del modelo con distintas GPU dentro de cada nodo que se utilizan para entrenar diferentes partes del modelo.

El entrenamiento distribuido tiene una mayor sobrecarga en comparación con el entrenamiento en una sola máquina. Sin embargo, cada uno de los procesadores paralelos y GPGPU descritos en el presente documento puede implementar diversas técnicas para reducir la sobrecarga del entrenamiento distribuido, que incluyen técnicas para permitir la transferencia de datos de GPU a GPU de alto ancho de banda y la sincronización remota acelerada de datos.

Aplicaciones ejemplares de aprendizaje automático

El aprendizaje automático se puede aplicar para resolver una variedad de problemas tecnológicos, que incluyen, pero no se limitan a, la visión por ordenador, la conducción y navegación autónomas, el reconocimiento de voz y el procesamiento del lenguaje. La visión por ordenador ha sido tradicionalmente una de las áreas de investigación más activas para las aplicaciones de aprendizaje automático. Las aplicaciones de la visión por ordenador van desde la reproducción de las capacidades visuales humanas, como el reconocimiento de rostros, hasta la creación de nuevas categorías de capacidades visuales. Por ejemplo, las aplicaciones de visión por ordenador pueden configurarse para reconocer ondas de sonido a partir de las vibraciones inducidas en objetos visibles en un vídeo. El aprendizaje automático acelerado por procesador paralelo permite entrenar las aplicaciones de visión por ordenador utilizando un conjunto de datos de entrenamiento significativamente mayor de lo que era posible anteriormente y permite implantar sistemas de inferencia utilizando procesadores paralelos de baja potencia.

El aprendizaje automático acelerado por procesador paralelo tiene aplicaciones de conducción autónomas que incluyen el reconocimiento de carriles y señales de tráfico, la evitación de obstáculos, la navegación y el control de la conducción. Las técnicas de aprendizaje automático acelerado se pueden utilizar para entrenar modelos de conducción basados en conjuntos de datos que definen las respuestas apropiadas a una entrada de entrenamiento específica. Los procesadores paralelos descritos en el presente documento pueden permitir un entrenamiento rápido de las redes neuronales, cada vez más complejas, utilizadas para soluciones de conducción autónoma y permiten la implantación de procesadores de inferencia de baja potencia en una plataforma móvil adecuada para la integración en vehículos autónomos.

Las redes neuronales profundas aceleradas por procesador paralelo han permitido enfoques de aprendizaje automático para el reconocimiento automático de voz (ASR). El ASR incluye la creación de una función que calcula la secuencia lingüística más probable dada una secuencia acústica de entrada. El aprendizaje automático acelerado mediante el uso de redes neuronales profundas ha permitido la sustitución de los modelos ocultos de Markov (HMM) y los modelos de mezcla gaussiana (GMM) utilizados anteriormente para el ASR.

El aprendizaje automático acelerado por procesador paralelo también se puede utilizar para acelerar el procesamiento del lenguaje natural. Los procedimientos de aprendizaje automático pueden utilizar algoritmos de inferencia estadística

para producir modelos que sean robustos frente a entradas erróneas o desconocidas. Aplicaciones ejemplares de procesador de lenguaje natural incluyen la traducción automática de idiomas.

Las plataformas de procesamiento paralelo utilizadas para el aprendizaje automático se pueden dividir en plataformas de entrenamiento y plataformas de implantación. Las plataformas de entrenamiento son, en general, altamente paralelas e incluyen optimizaciones para acelerar el entrenamiento de un solo nodo y múltiples GPU y el entrenamiento de múltiples nodos y múltiples GPU. Procesadores paralelos ejemplares adecuados para el entrenamiento incluyen la unidad de procesamiento de gráficos de propósito general altamente paralela 700 de la FIG. 700 y el sistema informático de múltiples GPU 800 de la FIG. 800. Por el contrario, las plataformas de aprendizaje automático implantadas incluyen generalmente procesadores paralelos de menor potencia adecuados para su uso en productos tales como cámaras, robots autónomos y vehículos autónomos.

La FIG. 13 ilustra un sistema en chip (SOC) de inferencia 1300 ejemplar, adecuado para generar inferencias utilizando un modelo entrenado. El SOC 1300 puede integrar componentes de procesamiento, que incluyen un procesador de medios 1302, un procesador de visión 1304, una GPGPU 1306 y un procesador de múltiples núcleos 1308. El SOC 1300 puede incluir adicionalmente memoria en chip 1305 que puede habilitar un conjunto de datos en chip compartido al que puede acceder cada uno de los componentes de procesamiento. Los componentes de procesamiento pueden optimizarse para un funcionamiento de baja potencia para permitir la implantación en una variedad de plataformas de aprendizaje automático, incluidos vehículos autónomos y robots autónomos. Por ejemplo, una implementación del SOC 1300 se puede utilizar como una parte del sistema de control principal para un vehículo autónomo. Cuando el SOC 1300 está configurado para su uso en vehículos autónomos, el SOC está diseñado y configurado para cumplir con las normas de seguridad funcional pertinentes de la jurisdicción de implantación.

Durante el funcionamiento, el procesador de medios 1302 y el procesador de visión 1304 pueden trabajar en conjunto para acelerar las operaciones de visión por ordenador. El procesador de medios 1302 puede habilitar una decodificación de baja latencia de múltiples secuencias de vídeo de alta resolución (por ejemplo, 4K, 8K). Las secuencias de vídeo descodificadas se pueden escribir en un *búfer* en la memoria en chip 1305. El procesador de visión 1304 puede entonces analizar el vídeo descodificado y realizar operaciones de procesamiento preliminar en los fotogramas del vídeo descodificado como preparación al procesamiento de los fotogramas usando un modelo de reconocimiento de imágenes entrenado. Por ejemplo, el procesador de visión 1304 puede acelerar las operaciones de convolución para una CNN que se utiliza para realizar el reconocimiento de imágenes en los datos de vídeo de alta resolución, mientras que los cálculos del modelo de sección de procesamiento se realizan mediante la GPGPU 1306.

El procesador de múltiples núcleos 1308 puede incluir lógica de control para ayudar con la secuenciación y sincronización de las transferencias de datos y las operaciones de memoria compartida realizadas por el procesador de medios 1302 y el procesador de visión 1304. El procesador de múltiples núcleos 1308 también puede funcionar como un procesador de aplicaciones para ejecutar aplicaciones de software que pueden utilizar la capacidad de cálculo de inferencia de la GPGPU 1306. Por ejemplo, al menos una parte de la lógica de navegación y conducción puede implementarse en software que se ejecuta en el procesador de múltiples núcleos 1308. Dicho software puede proporcionar directamente cargas de trabajo computacionales a la GPGPU 1306 o las cargas de trabajo computacionales pueden proporcionarse al procesador de múltiples núcleos 1308, que puede descargar al menos una parte de esas operaciones a la GPGPU 1306.

La GPGPU 1306 puede incluir agrupaciones de cálculo tales como una configuración de baja potencia de las agrupaciones de cálculo 706A-706H dentro de la unidad de procesamiento de gráficos de propósito general altamente paralela 700. Las agrupaciones de cálculo dentro de la GPGPU 1306 pueden admitir instrucciones que están optimizadas específicamente para realizar cálculos de inferencia en una red neuronal entrenada. Por ejemplo, la GPGPU 1306 puede admitir instrucciones para realizar cálculos de baja precisión tales como operaciones de vectores de números enteros de 8 bits y 4 bits.

Inferencia mixta mediante baja y alta precisión

La computación acelerada por GPGPU permite la descarga de partes paralelas de una aplicación hacia la GPGPU, mientras que el resto del código de programa se ejecuta en un procesador de ordenador principal (por ejemplo, una CPU). Como se describe en el presente documento, las GPGPU incluyen unidades de cálculo con la capacidad de realizar operaciones de números enteros y de coma flotante. Por lo general, esas operaciones son exclusivas en el sentido de que una unidad de cálculo que se asigna para realizar una operación de números enteros desactivará o, de otro modo, inhabilitará los elementos de cálculo encargados de las operaciones de coma flotante. Lo contrario también puede ser cierto en el sentido de que la unidad de cálculo puede inhabilitar componentes de números enteros al realizar operaciones de coma flotante. Dicha configuración permite una reducción en el consumo de energía operativa de cada unidad de cálculo. De manera alternativa, algunas unidades de cálculo pueden configurarse para realizar cálculos de forma selectiva en una de múltiples precisiones. Por ejemplo, una unidad de cálculo puede configurarse para realizar una operación FP32 o una operación FP16 dual. Una unidad de cálculo configurada para realizar operaciones de números enteros de 32 bits puede realizar cuatro operaciones simultáneas de números enteros de 8 bits. Unidades de cálculo selectivas de múltiple precisión o de múltiples datos pueden habilitar unidades de cálculo que tienen capacidades robustas que también son energéticamente eficientes, en el sentido de que una sola unidad

de cálculo puede realizar una variedad de operaciones, al tiempo que unidades lógicas inactivas dentro de cada unidad de cálculo se inhabilitan para reducir el consumo de energía operativa.

Sin embargo, también es posible habilitar un funcionamiento energéticamente eficiente de una unidad de cálculo utilizando unidades lógicas que de otra manera estarían inactivas dentro de una unidad de cálculo. En las formas de realización descritas en el presente documento, los componentes lógicos de precisión variable y/o de datos variables dentro de una unidad de cálculo pueden hacerse funcionar simultáneamente, de modo que las unidades de cálculo que no se utilizan para atender una operación inicial pueden habilitarse para procesar una o más operaciones adicionales. Por ejemplo, y en una forma de realización, una unidad de cálculo que tiene unidades de lógica de números enteros y de coma flotante puede procesar operaciones de números enteros y de coma flotante simultáneamente. En una forma de realización, una unidad de cálculo configurada para realizar de manera selectiva una operación de 32 bits o una operación dual de 16 bits puede estar configurada para realizar una operación de 32 bits y una operación dual de 16 bits o una operación de 32 bits y múltiples operaciones independientes de 16 bits. En una forma de realización, tales operaciones se habilitan al permitir que múltiples instrucciones de diferentes tipos se proporcionen a una única unidad de cálculo. En una forma de realización, las instrucciones de tipo de datos mixtos están habilitadas para permitir que operaciones de tipo "una sola instrucción, múltiples subprocesos" acepten operandos de precisión mixta y/o de tipo de datos mixtos.

La FIG. 14 es un diagrama de bloques de una unidad de multiprocesador 1400, de acuerdo con una forma de realización. La unidad de multiprocesador 1400 puede ser una variante de un multiprocesador de gráficos 234 de la FIG. 2D. La unidad de multiprocesador 1400 incluye una unidad de obtención y descodificación 1402, una unidad de ramificación 1404, un archivo de registros 1406, un gestor de subprocesos 1406, una unidad de tipo "una sola instrucción, múltiples subprocesos" (unidad SIMT 1410) y un gestor de voltaje y frecuencia 1420. La unidad de obtención y descodificación 1402 puede obtener una instrucción para su ejecución mediante la unidad de multiprocesador 1400. La unidad de ramificación 1404 puede calcular ajustes de puntero de instrucción en base a una instrucción de salto ejecutada. El archivo de registro 1406 puede almacenar registros de propósito general y de arquitectura utilizados por la unidad SIMT 1410. El gestor de subprocesos 1406 puede distribuir y redistribuir subprocesos entre las unidades de cálculo de la unidad SIMT 1410. En una forma de realización, la unidad SIMT 1410 está configurada para ejecutar una única instrucción como múltiples subprocesos, donde cada subproceso de la instrucción es ejecutado por una unidad de cálculo distinta. En una forma de realización, las unidades de cálculo 1411 a 1418 incluyen una ALU de números enteros (por ejemplo, ALU 1411A-1418A) y una unidad de coma flotante (por ejemplo, FPU 1411B-1418B). El voltaje y la frecuencia de cada unidad de cálculo 1411-1418 dentro de la unidad SIMT 1410 pueden gestionarse dinámicamente mediante el gestor de voltaje y frecuencia 1420, que puede aumentar o disminuir el voltaje y la frecuencia de reloj suministrados a las diversas unidades de cálculo a medida que los componentes de las unidades de cálculo se habilitan e inhabilitan.

En algunas configuraciones habilitadas previamente, cada unidad de cálculo puede ejecutar un solo subproceso de una instrucción de números enteros o una instrucción de coma flotante. Si cualquiera de las ALU 1411A-1418A tiene la tarea de ejecutar un subproceso de una instrucción de números enteros, la FPU 1411B - FPU 1418B respectiva no está disponible para su uso para ejecutar un subproceso de una instrucción de coma flotante y puede desactivarse durante el funcionamiento de la ALU 1411A - ALU 1418A correspondiente. Por ejemplo, cuando la ALU 1411A puede ejecutar un subproceso de una instrucción de números enteros mientras que la FPU 1413B ejecuta un subproceso de una instrucción de coma flotante, la FPU 1411B se desactiva cuando la ALU 1411A está activa. Las formas de realización descritas en el presente documento superan dichas limitaciones al permitir, por ejemplo, que ALU 1411A ejecute un subproceso de una instrucción mientras que la FPU 1411B ejecuta un subproceso de una instrucción diferente. Además, una forma de realización proporciona soporte para operandos de precisión mixta o de tipo de datos mixtos, de modo que una única instrucción puede realizar simultáneamente operaciones para una instrucción que tiene operandos de coma flotante y de números enteros y/u operandos que tienen diferentes precisiones.

Las formas de realización descritas en el presente permiten un mayor rendimiento operativo para una agrupación de unidades de cálculo al hacer que todas las unidades lógicas dentro de cada unidad de cálculo estén disponibles para realizar cálculos. En tales formas de realización, las unidades lógicas dentro de una unidad de cálculo que están diseñadas para realizar cálculos de forma selectiva en una de múltiples precisiones o múltiples tipos de datos pueden configurarse para realizar múltiples operaciones simultáneas para cada precisión o tipo de datos admitidos por la unidad de cálculo. Para una unidad de cálculo dada 1411-1418, las ALU 1411A-1418A pueden realizar operaciones de números enteros mientras que las FPU 1411B-1418B realizan operaciones de coma flotante. Estas operaciones se pueden realizar para una única instrucción o para múltiples instrucciones. En una forma de realización se habilita una nueva clase de instrucción de precisión mixta en la que uno o más operandos son de un tipo de datos o precisión, mientras que uno o más operandos diferentes son de un tipo de datos o precisión diferentes. Por ejemplo, una instrucción puede aceptar dos o más operandos de elementos múltiples que incluyen tipos de datos de números enteros y de coma flotante y una única instrucción se realiza por tipo de datos o por precisión.

La FIG. 15 ilustra un sistema de procesamiento de precisión mixta 1500, de acuerdo con una forma de realización. El sistema de procesamiento de precisión mixta 1500 incluye una unidad de cálculo 1509 que incluye una FPU 1508A y una ALU 1508B. En las formas de realización descritas en el presente documento, la unidad de cálculo 1509 puede ejecutar una instrucción de precisión mixta/tipo de datos mixtos. Por ejemplo, y en una forma de realización, una

instrucción puede procesarse para realizar una sola operación o múltiples operaciones fusionadas en múltiples operandos que incluyen múltiples elementos de datos. En una forma de realización, los elementos de datos dentro de un operando pueden ser elementos de precisión mixta o de tipo de datos mixtos. Por ejemplo, un primer registro de entrada 1501 puede almacenar un primer operando que incluye un elemento de coma flotante 1502A y múltiples elementos de números enteros, que incluyen un elemento de números enteros 1504A y un elemento de números enteros 1506A. Un segundo registro de entrada 1503 puede almacenar un segundo operando que incluye un elemento de coma flotante 1502B y múltiples elementos de números enteros, que incluyen un elemento de números enteros 1504B y un elemento de números enteros 1506B. Un tercer registro de entrada 1505 puede almacenar un tercer operando que incluye un elemento de coma flotante 1502C y múltiples elementos de números enteros, que incluyen un elemento de números enteros 1504C y un elemento de números enteros 1506C.

Los elementos se pueden utilizar como entrada para realizar una sola operación definida por un único código de operación (*opcode*) (por ejemplo, código de operación 1510). Por ejemplo, el código de operación 1510 puede especificar una operación fusionada de multiplicación y suma de múltiples elementos en la que la FPU 1508A multiplica y suma elementos de coma flotante. La FPU 1508A puede multiplicar elementos de coma flotante (elemento FP 1502A y elemento FP 1502B) y sumar el producto de la multiplicación a un tercer elemento de coma flotante (elemento FP 1502C). En paralelo, la ALU 1508B puede realizar una operación fusionada de multiplicación y suma de números enteros duales en la que se multiplica un primer conjunto de elementos de números enteros (elemento INT 1504A y elemento INT 1504B) y un segundo conjunto de elementos de números enteros (elemento INT 1506A y elemento INT 1506B) y el producto de cada multiplicación se suma a un tercer elemento de números enteros (elemento INT 1504C y elemento INT 1506C). Distintas banderas de estado 1512A-1512C se pueden establecer en base a cada operación. Las distintas banderas de estado 1512A-1512C se pueden establecer para indicar una salida de estado conocida en la técnica, que incluye, pero sin limitarse a, acarreo, negativo, cero y desbordamiento. En una forma de realización, las distintas banderas de estado 1512A-1512C pueden proporcionarse como un vector de estado, donde cada elemento del vector de estado está asociado a cada operación. Se pueden generar múltiples resultados (por ejemplo, resultado 1522A, resultado 1522B, resultado 1522C), donde un primer resultado 1522A es un resultado en coma flotante y el segundo y tercer resultados 1522B-1522C son resultados en números enteros.

En una forma de realización, la precisión de los elementos también se puede mezclar. Por ejemplo, y en una forma de realización, el registro de entrada 1501, el registro de entrada 1503 y el registro de entrada 1505 son registros de 32 bits. Los elementos FP 1502A-1502C puede ser elementos de coma flotante de 16 bits (por ejemplo, FP16), mientras que los elementos INT 1504A-1504C y los elementos INT 1506A-1506C pueden ser cada uno elementos de números enteros de 8 bits (por ejemplo, INT8). El registro de salida 1520 también puede ser un registro de 32 bits, donde el resultado 1522A es de 16 bits, mientras que el resultado 1522B y el resultado 1522C son cada uno de 8 bits. En varias formas de realización se pueden utilizar diferentes tamaños de registro, que incluyen registros de 64 bits, 128 bits, 256 bits y 512 bits, lo que permite elementos de entrada que oscilan entre 8 bits y 64 bits.

La lógica operativa 1700 para el sistema de procesamiento de precisión mixta 1500 se muestra en la FIG. 17, y se puede implementar mediante la unidad de multiprocesador 1400 de la FIG. 14. Con referencia adicional a la FIG. 14, la unidad de obtención y descodificación 1402 de la unidad de multiprocesador 1400 puede obtener y descodificar una única instrucción que incluye múltiples operandos, donde los múltiples operandos hacen referencia a múltiples elementos de datos que tienen diferentes precisiones, tal como se muestra en el bloque 1702 de la FIG. 17. El gestor de subprocesos 1406 puede enviar múltiples subprocesos de la única instrucción para su ejecución dentro de una unidad de cálculo (por ejemplo, la unidad de cálculo 1509 de la FIG. 15) de una GPGPU, tal como se muestra en el bloque 1704. En paralelo, la unidad de cálculo puede realizar una operación de instrucción en un primer conjunto de operandos que tienen una primera precisión a través de una primera unidad lógica dentro de una unidad de cálculo, como se muestra en el bloque 1706, y realizar la operación de instrucción en un segundo conjunto de operandos que tienen una segunda precisión a través de una segunda unidad lógica dentro de la unidad de cálculo, como se muestra en el bloque 1708. Las unidades lógicas pueden proporcionar múltiples resultados de la operación, tal como se muestra en el bloque 1710. Los diferentes operandos de precisión pueden ser del mismo tipo de datos (por ejemplo, coma flotante, coma fija, número entero) con diferentes precisiones (por ejemplo, 8 bits, 16 bits, 32 bits, etc.), mismos operandos de precisión de diferentes tipos de datos (por ejemplo, FP-16 e INT-16), o diferentes tipos de datos de diferente precisión (por ejemplo, INT-8 dual y FP16). Dichas instrucciones pueden ser particularmente útiles cuando se realizan operaciones de procesamiento para redes neuronales de precisión mixta o de tipo de datos mixtos en las que los datos de entrada para una capa son de una precisión o tipo de datos diferentes a los pesos aplicados a los datos de entrada.

Aunque una ALU de números enteros 150B se describe en la FIG. 15, las formas de realización no se limitan específicamente al uso de unidades aritmético-lógicas de únicamente de números enteros. En una forma de realización, las ALU de números enteros descritas en el presente documento pueden ser unidades de coma flotante configuradas para realizar operaciones de números enteros.

La FIG. 16 ilustra un sistema de procesamiento de precisión mixta 1600 adicional, de acuerdo con una forma de realización. El sistema de procesamiento de precisión mixta 1600 ilustrado está configurado para permitir la ejecución paralela de cargas de trabajo de coma flotante (por ejemplo, carga de trabajo FP 1602) y cargas de trabajo de números enteros (por ejemplo, carga de trabajo INT 1604) en múltiples unidades de cálculo (por ejemplo, unidad de cálculo

1607 y unidad de cálculo 1609). Tanto la unidad de cálculo 1607 como la unidad de cálculo 1609 incluyen un conjunto de unidades de coma flotante (FPU 1606A, FPU 1608A) y un conjunto de unidades aritmético-lógicas de números enteros (por ejemplo, ALU 1606B, ALU 1608B). En implementaciones anteriores, por ejemplo, un subproceso de carga de trabajo FP 1602 se ejecutaría, por ejemplo, en las FPU 1606A de la unidad de cálculo 1607, mientras que un subproceso de la carga de trabajo INT 1604 se ejecutaría en paralelo en las ALU 1608B de la unidad de cálculo 1609. Las formas de realización descritas en el presente documento permiten que las FPU y las ALU de las unidades de cálculo funcionen en paralelo.

La lógica operativa 1800 para el sistema de procesamiento de precisión mixta 1600 se muestra en la FIG. **18.** En una forma de realización, la lógica operativa 1800 puede obtener y descodificar una instrucción de números enteros que se ejecutará a través de múltiples subprocesos dentro de una GPGPU, tal como se muestra en el bloque 1802. La lógica 1800 también puede obtener y descodificar una instrucción de coma flotante que se ejecutará a través de múltiples subprocesos dentro de la GPGPU, como se muestra en el bloque 1804. La lógica operativa 1800 puede permitir la ejecución paralela de las instrucciones de números enteros y de coma flotante y ejecutar la instrucción de números enteros a través de unidades de números enteros de una primera unidad de cálculo y una segunda unidad de cálculo en el bloque 1805 mientras se ejecuta la instrucción de coma flotante a través de unidades de coma flotante de la primera unidad de cálculo y la segunda unidad de cálculo, como se muestra en el bloque 1806. La lógica 1800 puede hacer que la instrucción de números enteros genere resultados en números enteros en el bloque 1807, mientras que hace que la operación de coma flotante genere resultados en coma flotante en el bloque 1808.

Sistema de inferencia especializada para una red neuronal

La realización de operaciones de inferencia se puede hacer más eficaz en sistemas de aprendizaje automático implantados si el procesador paralelo o GPGPU utilizados para realizar las operaciones de inferencia están especializados para el tipo de cálculos realizados durante la inferencia para una red neuronal. En una forma de realización, la lógica de inferencia especializada puede realizar cálculos utilizando datos de pesos comprimidos y/o codificados en memoria. La codificación adaptativa se puede habilitar en función de un perfil generado para una red neuronal. El perfil se puede generar en función de una determinación de un conjunto de valores de peso comúnmente utilizados o patrones comunes que aparecen dentro de los valores de peso. Codificaciones de menos octetos para valores de peso comunes pueden almacenarse en memoria, lo que reduce los requisitos de energía o permite que redes más grandes se almacenen en memoria.

La FIG. 19 ilustra un sistema de aprendizaje automático 1900, de acuerdo con una forma de realización. En una forma de realización, el sistema de aprendizaje automático 1900 es un sistema de inferencia especializada para una red neuronal que admite la codificación de datos de pesos para una red neuronal. Los datos de pesos codificados pueden permitir una representación de tamaño reducido de la información de peso para una red neuronal implantada. El tamaño reducido de los datos de peso puede permitir inferencias energéticamente eficientes o puede permitir el procesamiento de redes neuronales más grandes para un tamaño de memoria dado.

En una forma de realización, el sistema de aprendizaje automático 1900 comienza con una red neuronal no entrenada 1902 que se puede procesar mediante un sistema de entrenamiento 1903. El sistema de entrenamiento 1903 puede generar una red neuronal entrenada 1904. Se puede utilizar un sistema de perfilado de datos de pesos 1905 para perfilar los datos de pesos de la red neuronal entrenada 1904. El sistema de perfilado de datos de pesos 1905 puede generar datos de pesos codificados en frecuencia 1906, así como un perfil de codificación 1910 que se utilizó para generar los datos de pesos codificados en frecuencia 1906. Los datos de pesos codificados en frecuencia 1906 y el perfil de codificación 1910 pueden almacenarse en la memoria de GPGPU 1908.

Los datos de pesos codificados en frecuencia 1906 y el perfil de codificación 1910 almacenados en la memoria de GPGPU 1908 se pueden utilizar para realizar cálculos de capa de red neuronal en una unidad de cálculo de GPGPU 1914. En una forma de realización, el perfil de codificación 1910 puede leerse de la memoria de GPGPU 1908 y utilizarse para configurar un descodificador de pesos de GPGPU 1912. El descodificador de pesos de GPGPU 1912 puede descodificar los datos de pesos codificados en frecuencia 1906 para proporcionar datos de pesos descodificados 1913 a la unidad de cálculo de GPGPU 1914. Los datos de entrada 1911 para una capa de red neuronal también se pueden leer de la memoria de GPGPU 1908. Los datos de entrada 1911 y los datos de peso descodificados 1913 se pueden procesar mediante la unidad de cálculo de GPGPU 1914 para generar resultados de cálculo de GPGPU 1915.

En una forma de realización, la unidad de cálculo de GPGPU 1914 puede estar configurada para incluir el descodificador de pesos de GPGPU 1912, de modo que el perfil de codificación 1910 y los datos de pesos codificados en frecuencia 1906 pueden proporcionarse directamente a la unidad de cálculo 1914 junto con datos de entrada 1911 para generar resultados de cálculo de GPGPU 1915.

Las operaciones lógicas 2000 del sistema de aprendizaje automático 1900 se ilustran mediante el diagrama de flujo de la FIG. 20. En una forma de realización, las operaciones lógicas 2000 pueden configurar una GPGPU para perfilar pesos de una red neuronal entrenada para generalizar un perfil de pesos para los datos de pesos de la red neuronal, como se muestra en el bloque 2002. La lógica 2000 puede entonces hacer que la GPGPU codifique los pesos de la

red neuronal utilizando el perfil de peso, tal como se muestra en el bloque 2004. La lógica 2000 puede hacer entonces que la GPGPU almacene pesos codificados y el perfil de peso a la memoria de GPGPU, como se muestra en el bloque 2006. Mientras se realizan cálculos para una red neuronal, las operaciones lógicas 2000 pueden hacer que la GPGPU lea pesos codificados de la memoria de GPGPU durante el procesamiento de la red neuronal, como se muestra en el bloque 2008. En una forma de realización, las operaciones lógicas 2000 pueden hacer que la GPGPU descodifique los pesos codificados en función de un perfil de peso, tal como se muestra en el bloque 2010, antes de que la GPGPU realice cálculos para la red neuronal en el bloque 2012. Los pesos codificados se pueden descodificar usando el descodificador de pesos de GPGPU 1912 de la FIG. 19. El proceso de descodificación se puede omitir cuando la GPGPU está configurada para aceptar directamente pesos codificados de la memoria de GPGPU. En dicha forma de realización, las operaciones lógicas 2000 se pueden configurar de modo que la GPGPU realice cálculos para la red neuronal en el bloque 2012 sin descodificar previamente los datos de pesos.

Sistema de procesamiento de gráficos ejemplar adicional

Los detalles de las formas de realización descritas anteriormente se pueden incorporar en los sistemas y dispositivos de procesamiento de gráficos descritos a continuación. El sistema y los dispositivos de procesamiento de gráficos de las FIG. 21-34 ilustran sistemas alternativos y hardware de procesamiento de gráficos que pueden implementar todas y cada una de las técnicas descritas anteriormente.

Visión general de un sistema de procesamiento de gráficos ejemplar adicional

La FIG. 21 es un diagrama de bloques de un sistema de procesamiento 2100, de acuerdo con una forma de realización. En diversas formas de realización, el sistema 2100 incluye uno o más procesadores 2102 y uno o más procesadores de gráficos 2108, y puede ser un sistema de escritorio de un solo procesador, un sistema de estación de trabajo de múltiples procesadores o un sistema de servidor que tiene un gran número de procesadores 2102 o de núcleos de procesador 2107. En una forma de realización, el sistema 2100 es una plataforma de procesamiento incorporada dentro de un circuito integrado de sistema en chip (SoC) para su uso en dispositivos móviles, portátiles o integrados.

Una forma de realización del sistema 2100 puede incluir, o incorporarse en, una plataforma de juegos basada en servidor, una consola de juegos, que incluye una consola de juegos y medios, una consola de juegos móvil, una consola de juegos de mano o una consola de juegos en línea. En algunas formas de realización, el sistema 2100 es un teléfono móvil, teléfono inteligente, dispositivo de tableta electrónica o un dispositivo de Internet móvil. El sistema de procesamiento de datos 2100 también puede incluir, estar acoplado a o estar integrado dentro de un dispositivo ponible, tal como un dispositivo ponible de reloj inteligente, un dispositivo de gafas inteligentes, un dispositivo de realidad aumentada o un dispositivo de realidad virtual. En algunas formas de realización, el sistema de procesamiento de datos 2100 es un dispositivo de televisión o descodificador que tiene uno o más procesadores 2102 y una interfaz gráfica generada por uno o más procesadores de gráficos 2108.

En algunas formas de realización, cada procesador de los uno o más procesadores 2102 incluye uno o más núcleos de procesador 2107 para procesar instrucciones que, cuando se ejecutan, realizan operaciones para el sistema y el software de usuario. En algunas formas de realización, cada núcleo de los uno o más núcleos de procesador 2107 está configurado para procesar un conjunto de instrucciones específico 2109. En algunas formas de realización, el conjunto de instrucciones 2109 puede facilitar el cálculo de conjuntos de instrucciones complejas (CISC), el cálculo de conjuntos de instrucciones reducidas (RISC) o el cálculo a través de una palabra de instrucción muy larga (VLIW). Cada uno de los múltiples núcleos de procesador 2107 puede procesar un conjunto de instrucciones diferente 2109, que puede incluir instrucciones para facilitar la emulación de otros conjuntos de instrucciones. El núcleo de procesador 2107 también puede incluir otros dispositivos de procesamiento, tales como un procesador de señales digitales (DSP).

En algunas formas de realización, el procesador 2102 incluye memoria caché 2104. Dependiendo de la arquitectura, el procesador 2102 puede tener una única memoria caché interna o múltiples niveles de memoria caché interna. En algunas formas de realización, la memoria caché se comparte entre varios componentes del procesador 2102. En algunas formas de realización, el procesador 2102 también utiliza una memoria caché externa (por ejemplo, una memoria caché de nivel 3 (L3) o memoria caché de último nivel (LLC)) (no mostrada), que puede compartirse entre los núcleos de procesador 2107 utilizando técnicas de coherencia de memoria caché conocidas. Un archivo de registro 2106 se incluye adicionalmente en el procesador 2102, que puede incluir diferentes tipos de registros para almacenar diferentes tipos de datos (por ejemplo, registros de números enteros, registros de coma flotante, registros de estado y un registro de puntero de instrucción). Algunos registros pueden ser registros de propósito general, mientras que otros registros pueden ser específicos para el diseño del procesador 2102.

En algunas formas de realización, el procesador 2102 está acoplado a un bus de procesador 2110 para transmitir señales de comunicación tales como señales de dirección, datos o control entre el procesador 2102 y otros componentes del sistema 2100. En una forma de realización, el sistema 2100 utiliza una arquitectura de sistema de "concentrador" ejemplar, que incluye un concentrador de controladores de memoria 2116 y un concentrador de controladores de entrada/salida (E/S) 2130. Un concentrador de controladores de memoria 2116 facilita la comunicación entre un dispositivo de memoria y otros componentes del sistema 2100, mientras que un concentrador de controladores de E/S (ICH) 2130 proporciona conexiones a dispositivos de E/S a través de un bus de E/S local. En

una forma de realización, la lógica del concentrador de controladores de memoria 2116 está integrada dentro del procesador.

El dispositivo de memoria 2120 puede ser un dispositivo de memoria de acceso aleatorio dinámica (DRAM), un dispositivo de memoria de acceso aleatorio estática (SRAM), un dispositivo de memoria flash, un dispositivo de memoria de cambio de fase o algún otro dispositivo de memoria que tenga un funcionamiento adecuado para servir como memoria de proceso. En una forma de realización, el dispositivo de memoria 2120 puede funcionar como memoria de sistema para el sistema 2100, para almacenar datos 2122 e instrucciones 2121 para su uso cuando el uno o más procesadores 2102 ejecutan una aplicación o proceso. El concentrador de controladores de memoria 2116 también está acoplado a un procesador de gráficos externo opcional 2112 que puede comunicarse con el uno o más procesadores de gráficos 2108 en los procesadores 2102 para realizar operaciones de gráficos y medios.

En algunas formas de realización, el ICH 2130 permite que dispositivos periféricos se conecten al dispositivo de memoria 2120 y al procesador 2102 a través de un bus de E/S de alta velocidad. Los dispositivos periféricos de E/S incluyen, pero sin limitarse a, un controlador de audio 2146, una interfaz de firmware 2128, un transceptor inalámbrico 2126 (por ejemplo, Wi-Fi, Bluetooth), un dispositivo de almacenamiento de datos 2124 (por ejemplo, unidad de disco duro, memoria flash, etc.) y un controlador de E/S heredado 2140 para acoplar dispositivos heredados (por ejemplo, un sistema personal 2 (PS/2)) al sistema. Uno o más controladores de bus serie universal (USB) 2142 conectan dispositivos de entrada, tales como combinaciones de teclado y ratón 2144. Un controlador de red 2134 también puede acoplarse al ICH 2130. En algunas formas de realización, un controlador de red de alto rendimiento (no mostrado) está acoplado al bus de procesador 2110. Se apreciará que el sistema 2100 mostrado es ejemplar y no limitativo, ya que también se pueden utilizar otros tipos de sistemas de procesamiento de datos que están configurados de manera diferente. Por ejemplo, el concentrador de controladores de E/S 2130 puede estar integrado dentro del uno o más procesadores 2102, o el concentrador de controladores de memoria 2116 y el concentrador de controladores de E/S 2130 pueden estar integrados en un procesador de gráficos externo discreto, tal como el procesador de gráficos externo 2112.

La FIG. 22 es un diagrama de bloques de una forma de realización de un procesador 2200 que tiene uno o más núcleos de procesador 2202A-2202N, un controlador de memoria integrado 2214 y un procesador de gráficos integrado 2208. Los elementos de la FIG. 22 que tienen los mismos números (o nombres) de referencia que los elementos de cualquier otra figura del presente documento pueden actuar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a esto. El procesador 2200 puede incluir núcleos adicionales hasta e incluyendo el núcleo adicional 2202N representados mediante recuadros de línea discontinua. Cada uno de los núcleos de procesador 2202A-2202N incluye una o más unidades de memoria caché interna 2204A-2204N. En algunas formas de realización, cada núcleo de procesador también tiene acceso a una o más unidades de memoria caché compartida 2206.

Las unidades de memoria caché interna 2204A-2204N y las unidades de memoria caché compartida 2206 representan una jerarquía de memoria caché dentro del procesador 2200. La jerarquía de memoria caché puede incluir al menos un nivel de instrucción y memoria caché de datos dentro de cada núcleo de procesador y uno o más niveles de memoria caché de nivel medio compartida, tal como un nivel 2 (L2), nivel 3 (L3), nivel 4 (L4) u otros niveles de memoria caché, donde el nivel más alto de memoria caché antes de la memoria externa se clasifica como la LLC. En algunas formas de realización, la lógica de coherencia de memoria caché mantiene la coherencia entre las diversas unidades de memoria caché 2206 y 2204A-2204N.

En algunas formas de realización, el procesador 2200 también puede incluir un conjunto de una o más unidades de controlador de bus 2216 y un núcleo de agente de sistema 2210. La una o más unidades de controlador de bus 2216 gestionan un conjunto de buses periféricos, tales como uno o más buses de interconexión de componentes periféricos (por ejemplo, PCI, PCI Express). El núcleo de agente de sistema 2210 proporciona funcionalidad de gestión para los diversos componentes de procesador. En algunas formas de realización, el núcleo de agente de sistema 2210 incluye uno o más controladores de memoria integrados 2214 para gestionar el acceso a varios dispositivos de memoria externos (no mostrados).

En algunas formas de realización, uno o más de los núcleos de procesador 2202A-2202N incluyen soporte para múltiples subprocesos simultáneos. En dicha forma de realización, el núcleo de agente de sistema 2210 incluye componentes para coordinar y hacer funcionar núcleos 2202A-2202N durante un procesamiento de múltiples subprocesos. El núcleo de agente de sistema 2210 puede incluir adicionalmente una unidad de control de potencia (PCU), que incluye lógica y componentes para regular el estado de potencia de los núcleos de procesador 2202A-2202N y del procesador de gráficos 2208.

En algunas formas de realización, el procesador 2200 incluye adicionalmente el procesador de gráficos 2208 para ejecutar operaciones de procesamiento de gráficos. En algunas formas de realización, el procesador de gráficos 2208 está acoplado al conjunto de unidades de memoria caché compartida 2206 y al núcleo de agente de sistema 2210, que incluye el uno o más controladores de memoria integrados 2214. En algunas formas de realización, un controlador de visualización 2211 está acoplado al procesador de gráficos 2208 para dirigir la salida del procesador de gráficos a uno o más dispositivos de visualización acoplados. En algunas formas de realización, el controlador de visualización

2211 puede ser otro módulo acoplado al procesador de gráficos a través de al menos una interconexión, o puede estar integrado dentro del procesador de gráficos 2208 o del núcleo de agente de sistema 2210.

En algunas formas de realización, se utiliza una unidad de interconexión basada en anillo 2212 para acoplar los componentes internos del procesador 2200. Sin embargo, se puede utilizar una unidad de interconexión alternativa, tal como una interconexión de punto a punto, una interconexión conmutada u otras técnicas, incluidas técnicas ampliamente conocidas en la técnica. En algunas formas de realización, el procesador de gráficos 2208 está acoplado a la interconexión de anillo 2212 a través de un enlace de E/S 2213.

El enlace de E/S 2213 ejemplar representa al menos una de múltiples variedades de interconexiones de E/S, incluida una interconexión de E/S en encapsulado que facilita la comunicación entre diversos componentes de procesador y un módulo de memoria integrado de alto rendimiento 2218, tal como un módulo de eDRAM. En algunas formas de realización, cada uno de los núcleos de procesador 2202A-2202N y el procesador de gráficos 2208 utilizan módulos de memoria integrados 2218 como memoria caché de último nivel compartida.

En algunas formas de realización, los núcleos de procesador 2202A-2202N son núcleos homogéneos que ejecutan la misma arquitectura de conjuntos de instrucciones. En otra forma de realización, los núcleos de procesador 2202A-2202N son heterogéneos en cuanto a la arquitectura de conjuntos de instrucciones (ISA), donde uno o más de los núcleos de procesador 2202A-2202N ejecutan un primer conjunto de instrucciones, mientras que al menos uno de los otros núcleos ejecuta un subconjunto del primer conjunto de instrucciones o un conjunto de instrucciones diferente. En una forma de realización, los núcleos de procesador 2202A-2202N son heterogéneos en cuanto a la microarquitectura, donde uno o más núcleos que tienen un consumo de energía relativamente mayor están acoplados a uno o más núcleos de energía que tienen un consumo de energía menor. Además, el procesador 2200 se puede implementar en uno o más chips o como un circuito integrado de SoC que tiene los componentes ilustrados, además de otros componentes.

La FIG. 23 es un diagrama de bloques de un procesador de gráficos 2300, que puede ser una unidad de procesamiento de gráficos discreta o puede ser un procesador de gráficos integrado con una pluralidad de núcleos de procesamiento. En algunas formas de realización, el procesador de gráficos se comunica a través de una interfaz de E/S mapeada en memoria para registros del procesador de gráficos y con comandos ubicados en la memoria de procesador. En algunas formas de realización, el procesador de gráficos 2300 incluye una interfaz de memoria 2314 para acceder a memoria. La interfaz de memoria 2314 puede ser una interfaz para la memoria local, una o más memorias caché internas, una o más memorias caché externas compartidas y/o para la memoria del sistema.

En algunas formas de realización, el procesador de gráficos 2300 también incluye un controlador de visualización 2302 para dirigir los datos de salida de visualización a un dispositivo de visualización 2320. El controlador de visualización 2302 incluye hardware para uno o más planos de superposición para la visualización y composición de múltiples capas de vídeo o elementos de interfaz de usuario. En algunas formas de realización, el procesador de gráficos 2300 incluye un motor de códec de vídeo 2306 para codificar, descodificar o transcodificar medios hacia, desde o entre uno o más formatos de codificación de medios, que incluyen, pero sin limitarse a, formatos del Grupo de Expertos de Imágenes en Movimiento (MPEG) tales como MPEG-2, formatos de codificación avanzada de vídeo (AVC) tales como H.264/MPEG-4 AVC, así como formatos de la Sociedad de Ingenieros de Imágenes en Movimiento y de Televisión (SMPTE) 421M/VC-1 y del Grupo Conjunto de Expertos en Fotografía (JPEG) tales como los formatos JPEG y Motion JPEG (MJPEG).

En algunas formas de realización, el procesador de gráficos 2300 incluye un motor de transferencia de imágenes en bloques (BLIT) 2304 para realizar operaciones de rasterización bidimensionales (2D) que incluyen, por ejemplo, transferencias de bloques de límite de bits. Sin embargo, en una forma de realización, las operaciones de gráficos 2D se realizan utilizando uno o más componentes del motor de procesamiento de gráficos (GPE) 2310. En algunas formas de realización, el GPE 2310 es un motor de cálculo para realizar operaciones de gráficos, incluidas operaciones de gráficos tridimensionales (3D) y operaciones de medios.

En algunas formas de realización, el GPE 310 incluye un procesamiento en cadena 3D 2312 para realizar operaciones en 3D, tales como renderizar imágenes y escenas tridimensionales usando funciones de procesamiento que actúan sobre formas de primitivas en 3D (por ejemplo, rectángulo, triángulo, etc.). El procesamiento en cadena 3D 2312 incluye elementos de función fija y programables que realizan diversas tareas dentro del elemento y/o generan subprocesos de ejecución para un subsistema 3D/de medios 2315. Si bien el procesamiento en cadena 3D 2312 se puede utilizar para realizar operaciones de medios, una forma de realización de GPE 2310 también incluye un procesamiento en cadena de medios 2316 que se utiliza específicamente para realizar operaciones de medios, tal como posprocesamiento de vídeo y mejora de imagen.

En algunas formas de realización, el procesamiento en cadena de medios 2316 incluye unidades lógicas de función fija o programables para realizar una o más operaciones de medios especializados, tales como aceleración de descodificación de vídeo, desintercalación de vídeo y aceleración de codificación de vídeo en lugar de, o en nombre de, el motor de códec de vídeo 2306. En algunas formas de realización, la cadena de procesamiento de medios 2316 incluye adicionalmente una unidad de generación de subprocesos para generar subprocesos para su ejecución en el

subsistema 3D/de medios 2315. Los subprocesos generados realizan cálculos para las operaciones de medios en una o más unidades de ejecución de gráficos incluidas en el subsistema 3D/de medios 2315.

En algunas formas de realización, el subsistema de 3D/medios 2315 incluye lógica para ejecutar subprocesos generados por el procesamiento en cadena 3D 2312 y el procesamiento en cadena de medios 2316. En una forma de realización, los procesamientos en cadena envían solicitudes de ejecución de subprocesos al subsistema 3D/de medios 2315, que incluyen lógica de envío de subprocesos para arbitrar y enviar las diversas solicitudes a recursos de ejecución de subprocesos disponibles. Los recursos de ejecución incluyen una formación de unidades de ejecución de gráficos para procesar los subprocesos 3D y de medios. En algunas formas de realización, el subsistema de 3D/de medios 2315 incluye una o más memorias caché internas para instrucciones y datos de subprocesos. En algunas formas de realización, el subsistema también incluye memoria compartida, que incluye registros y memoria direccionable, para compartir datos entre subprocesos y almacenar datos de salida.

Motor de procesamiento de gráficos ejemplar adicional

La FIG. 24 es un diagrama de bloques de un motor de procesamiento de gráficos 2410 de un procesador de gráficos de acuerdo con algunas formas de realización. En una forma de realización, el motor de procesamiento de gráficos (GPE) 2410 es una versión del GPE 2310 mostrado en la FIG. **23.** Elementos de la FIG. **24** que tienen los mismos números (o nombres) de referencia que los elementos de cualquier otra figura del presente documento pueden actuar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a esto. Por ejemplo, se ilustra el procesamiento en cadena 3D 2312 y el procesamiento en cadena de medios 2316 de la FIG. 23. El procesamiento en cadena de medios 2316 es opcional en algunas formas de realización del GPE 2410 y puede no estar incluido explícitamente en el GPE 2410. Por ejemplo, y en al menos una forma de realización, otro procesador de medios y/o imágenes está acoplado al GPE 2410.

En algunas formas de realización, el GPE 2410 está acoplado a o incluye un transmisor de comandos 2403, que proporciona un flujo de comandos al procesamiento en cadena 3D 2312 y/o el procesamiento en cadena de medios 2316. En algunas formas de realización, el transmisor de comandos 2403 está acoplado a memoria, que puede ser memoria de sistema, o a una o más de memoria caché interna y memoria caché compartida. En algunas formas de realización, el transmisor de comandos 2403 recibe comandos desde la memoria y envía los comandos al procesamiento en cadena 3D 2312 y/o al procesamiento en cadena de medios 2316. Los comandos son directivas obtenidas de un *búfer* de anillo, que almacena comandos para el procesamiento en cadena 3D 2312 y el procesamiento en cadena de medios 2316. En una forma de realización, el *búfer* de anillo puede incluir adicionalmente *búferes* de comandos por lotes que almacenan lotes de múltiples comandos. Los comandos para el procesamiento en cadena 3D 2312 también pueden incluir referencias a datos almacenados en memoria, tal como, pero sin limitarse a, datos de vértice y geometría para el procesamiento en cadena 3D 2312 y/o datos de imagen y objetos de memoria para el procesamiento en cadena de medios 2316. El procesamiento en cadena 3D 2312 y el procesamiento en cadena de medios 2316 procesan los comandos y datos realizando operaciones mediante lógica en las cadenas de procesamiento respectivas o enviando uno o más subprocesos de ejecución a una formación de núcleos de gráficos 2414.

En diversas formas de realización, el procesamiento en cadena 3D 2312 puede ejecutar uno o más programas de sombreado, tales como sombreadores de vértices, sombreadores de geometría, sombreadores de píxeles, sombreadores de fragmentos, sombreadores de cálculo u otros programas de sombreado, mediante el procesamiento de las instrucciones y el envío de subprocesos de ejecución a la formación de núcleos de gráficos 2414. La formación de núcleos de gráficos 2414 proporciona un bloque unificado de recursos de ejecución. La lógica de ejecución de múltiples propósitos (por ejemplo, unidades de ejecución) de la formación de núcleos de gráficos 2414 incluye soporte para diversos lenguajes de sombreado de API 3D y puede ejecutar múltiples subprocesos de ejecución simultáneos asociados a múltiples sombreadores.

En algunas formas de realización, la formación de núcleos de gráficos 2414 también incluye lógica de ejecución para realizar funciones de medios, tales como procesamiento de imágenes y/o vídeo. En una forma de realización, las unidades de ejecución incluyen adicionalmente lógica de propósito general que puede programarse para realizar operaciones computacionales de propósito general paralelas, además de operaciones de procesamiento de gráficos. La lógica de propósito general puede realizar operaciones de procesamiento en paralelo o en conjunción con la lógica de propósito general dentro del/de los núcleo(s) de procesador 1607 de la FIG. 16 o el núcleo 2202A-2202N de la FIG. 22.

Los datos de salida generados por subprocesos que se ejecutan en la formación de núcleos de gráficos 2414 pueden proporcionar datos a la memoria en un *búfer* de retorno unificado (URB) 2418. El URB 2418 puede almacenar datos para múltiples subprocesos. En algunas formas de realización, el URB 2418 se puede utilizar para enviar datos entre diferentes subprocesos que se ejecutan en la formación de núcleos de gráficos 2414. En algunas formas de realización, el URB 2418 se puede utilizar adicionalmente para la sincronización entre subprocesos de la formación de núcleos de gráficos y la lógica de función fija dentro de la lógica de función compartida 2420.

En algunas formas de realización, la formación de núcleos de gráficos 2414 es escalable, de modo que la formación incluye un número variable de núcleos de gráficos, cada uno de los cuales tiene un número variable de unidades de ejecución en función de la potencia objetivo y el nivel de rendimiento del GPE 2410. En una forma de realización, los

recursos de ejecución son dinámicamente escalables, de modo que los recursos de ejecución se pueden habilitar o inhabilitar según sea necesario.

La formación de núcleos de gráficos 2414 está acoplada a la lógica de función compartida 2420 que incluye múltiples recursos que se comparten entre los núcleos de gráficos de la formación de núcleos de gráficos. Las funciones compartidas dentro de la lógica de función compartida 2420 son unidades lógicas de hardware que proporcionan funcionalidad complementaria especializada a la formación de núcleos de gráficos 2414. En varias formas de realización, la lógica de función compartida 2420 incluye, pero no se limita a, un muestreador 2421, lógica matemática 2422 y comunicación entre hilos (ITC) 2423. Además, algunas formas de realización implementan uno o más memorias caché 2425 dentro de la lógica de función compartida 2420. Se implementa una función compartida cuando la demanda de una función especializada dada es insuficiente para su inclusión dentro de la formación de núcleos de gráficos 2414. En cambio, una única instancia de esa función especializada se implementa como una entidad independiente en la lógica de función compartida 2420 y se comparte entre los recursos de ejecución dentro de la formación de núcleos de gráficos 2414. El conjunto preciso de funciones que se comparten entre la formación de núcleos de gráficos 2414 y que se incluyen dentro de la formación de núcleos de gráficos 2414 varía entre formas de realización.

La FIG. 25 es un diagrama de bloques de otra forma de realización de un procesador de gráficos 2500. Elementos de la FIG. 25 que tienen los mismos números (o nombres) de referencia que los elementos de cualquier otra figura del presente documento pueden actuar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a esto.

En algunas formas de realización, el procesador de gráficos 2500 incluye una interconexión de anillo 2502, una sección de entrada de procesamiento en cadena 2504, un motor de medios 2537 y núcleos de gráficos 2580A-2580N. En algunas formas de realización, la interconexión de anillo 2502 acopla el procesador de gráficos a otras unidades de procesamiento, incluidos otros procesadores de gráficos o uno o más núcleos de procesador de propósito general. En algunas formas de realización, el procesador de gráficos es uno de muchos procesadores integrados dentro de un sistema de procesamiento de múltiples núcleos.

En algunas formas de realización, el procesador de gráficos 2500 recibe lotes de comandos a través de la interconexión de anillo 2502. Los comandos entrantes son interpretados por un transmisor de comandos 2503 en la sección de entrada de procesamiento en cadena 2504. En algunas formas de realización, el procesador de gráficos 2500 incluye lógica de ejecución escalable para realizar procesamiento de geometría 3D y procesamiento de medios a través del/de los núcleo(s) de gráficos 2580A-2580N. Para los comandos de procesamiento de geometría 3D, el transmisor de comandos 2503 suministra comandos al procesamiento en cadena de geometría 2536. Para al menos algunos comandos de procesamiento de medios, el transmisor de comandos 2503 suministra los comandos a una sección de entrada de vídeo 2534, que está acoplada a un motor de medios 2537. En algunas formas de realización, el motor de medios 2537 incluye un motor de calidad de vídeo (VQE) 2530 para el posprocesamiento de imágenes y vídeo y un motor de codificación/descodificación de múltiples formatos (MFX) 2533 para proporcionar codificación y descodificación de datos de medios acelerados por hardware. En algunas formas de realización, tanto el procesamiento en cadena de geometría 2536 como el motor de medios 2537 generan subprocesos de ejecución para los recursos de ejecución de subprocesos proporcionados por al menos un núcleo de gráficos 2580A.

En algunas formas de realización, el procesador de gráficos 2500 incluye recursos de ejecución de subprocesos escalables que presentan núcleos modulares 2580A-2580N (a veces denominados sectores de núcleo), cada uno de los cuales tiene múltiples subnúcleos 2550A-2550N, 2560A-2560N (a veces denominados subsectores de núcleo). En algunas formas de realización, el procesador de gráficos 2500 puede tener cualquier número de núcleos de gráficos 2580A a 2580N. En algunas formas de realización, el procesador de gráficos 2500 incluye un núcleo de gráficos 2580A que tiene al menos un primer subnúcleo 2550A y un segundo subnúcleo 2560A. En otras formas de realización, el procesador de gráficos es un procesador de baja potencia con un solo subnúcleo (por ejemplo, 2550A). En algunas formas de realización, el procesador de gráficos 2500 incluye múltiples núcleos de gráficos 2580A-2580N, cada uno de los cuales incluye un conjunto de primeros subnúcleos 2550A-2550N y un conjunto de segundos subnúcleos 2560A-2560N. Cada subnúcleo del conjunto de primeros subnúcleos 2550A-2550N incluye al menos un primer conjunto de unidades de ejecución 2552A-2552N y muestreadores de medios/textura 2554A-2554N. Cada subnúcleo del conjunto de segundos subnúcleos 2560A-2560N incluye al menos un segundo conjunto de unidades de ejecución 2562A-2562N y muestreadores 2564A-2564N. En algunas formas de realización, cada subnúcleo 2550A-2550N, 2560A-2560N comparte un conjunto de recursos compartidos 2570A-2570N. En algunas formas de realización, los recursos compartidos incluyen memoria caché compartida y lógica de operaciones de píxeles. También se pueden incluir otros recursos compartidos en las diversas formas de realización del procesador de gráficos.

Unidades de ejecución ejemplares adicionales

La FIG. 26 ilustra una lógica de ejecución de subprocesos 2600 que incluye una formación de elementos de procesamiento empleados en algunas formas de realización de un GPE. Elementos de la FIG. 26 que tienen los mismos números (o nombres) de referencia que los elementos de cualquier otra figura del presente documento pueden

actuar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a esto.

En algunas formas de realización, la lógica de ejecución de subprocesos 2600 incluye un procesador de sombreado 2602, un distribuidor de subprocesos 2604, una memoria caché de instrucciones 2606, una formación de unidades de ejecución escalable que incluye una pluralidad de unidades de ejecución 2608A-2608N, un muestreador 2610, una memoria caché de datos 2612 y un puerto de datos 2614. En una forma de realización, la formación de unidades de ejecución escalable puede escalarse dinámicamente al habilitar o inhabilitar una o más unidades de ejecución (por ejemplo, cualquiera de las unidades de ejecución 2608A, 2608B, 2608C, 2608D hasta 2608N-1 y 2608N) en función de los requisitos de cálculo de una carga de trabajo. En una forma de realización, los componentes incluidos están interconectados a través de una estructura de interconexión que está vinculada a cada uno de los componentes. En algunas formas de realización, la lógica de ejecución de subprocesos 2600 incluye una o más conexiones a memoria, tal como memoria de sistema o memoria caché, a través de uno o más de entre memoria caché de instrucciones 2606, el puerto de datos 2614, el muestreador 2610 y las unidades de ejecución 2608A-2608N. En algunas formas de realización, cada unidad de ejecución (por ejemplo, 2608A) es una unidad de cálculo de propósito general programable independiente que es capaz de ejecutar múltiples subprocesos de hardware simultáneos mientras procesa múltiples elementos de datos en paralelo para cada subproceso. En varias formas de realización, la formación de unidades de ejecución 2608A-2608N puede escalarse para incluir cualquier número de unidades de ejecución individuales.

En algunas formas de realización, las unidades de ejecución 2608A-2608N se utilizan principalmente para ejecutar programas de sombreado. Un procesador de sombreado 2602 puede procesar los diversos programas de sombreado y enviar subprocesos de ejecución asociados a los programas de sombreado a través de un distribuidor de subprocesos 2604. En una forma de realización, el distribuidor de subprocesos incluye lógica para arbitrar solicitudes de inicio de subprocesos del procesamiento en cadena de gráficos y medios e instanciar los subprocesos solicitados en una o más unidades de ejecución de las unidades de ejecución 2608A-2608N. Por ejemplo, el procesamiento en cadena de geometría (por ejemplo, 2536 de la FIG. 25) puede enviar sombreadores de vértices, teselación o geometría a la lógica de ejecución de subprocesos 2600 (FIG. 26) para su procesamiento. En algunas formas de realización, el distribuidor de subprocesos 2604 también puede procesar solicitudes de generación de subprocesos en tiempo de ejecución de los programas de sombreado en ejecución.

En algunas formas de realización, las unidades de ejecución 2608A-2608N admiten un conjunto de instrucciones que incluye soporte nativo para muchas instrucciones de sombreado de gráficos 3D estándar, de modo que los programas de sombreado de bibliotecas de gráficos (por ejemplo, Direct 3D y OpenGL) se ejecutan con una conversión mínima. Las unidades de ejecución admiten procesamiento de vértices y geometría (por ejemplo, programas de vértices, programas de geometría, sombreadores de vértices), procesamiento de píxeles (por ejemplo, sombreadores de píxeles, sombreadores de fragmentos) y procesamiento de propósito general (por ejemplo, sombreadores de medios y cálculo). Cada una de las unidades de ejecución 2608A-2608N es capaz de una ejecución de tipo "una sola instrucción, múltiples datos" (SIMD) de emisión múltiple y el funcionamiento con múltiples subprocesos permite un entorno de ejecución eficaz frente a los accesos a memoria de mayor latencia. Cada subproceso de hardware dentro de cada unidad de ejecución tiene un archivo de registros de alto ancho de banda dedicado y un estado de subproceso independiente asociado. La ejecución es de emisión múltiple por reloj a procesamientos en cadena capaces de operaciones de números enteros y de coma flotante de precisión simple y doble, capacidad de ramificación SIMD, operaciones lógicas, operaciones trascendentales y otras operaciones diversas. Mientras espera datos de la memoria o de una de las funciones compartidas, la lógica de dependencia de las unidades de ejecución 2608A-2608N hace que un subproceso en espera quede en suspensión hasta que se devuelvan los datos solicitados. Mientras el subproceso en espera está en suspensión, los recursos de hardware se pueden dedicar a procesar otros subprocesos. Por ejemplo, durante un retardo asociado a una operación de sombreado de vértices, una unidad de ejecución puede realizar operaciones de un sombreador de píxeles, sombreador de fragmentos u otro tipo de programa de sombreado, incluido un sombreador de vértices diferente.

Cada unidad de ejecución de las unidades de ejecución 2608A-2608N realiza operaciones en formaciones de elementos de datos. El número de elementos de datos es el "tamaño de ejecución" o el número de canales para la instrucción. Un canal de ejecución es una unidad lógica de ejecución para el acceso a elementos de datos, el enmascaramiento y el control de flujo dentro de las instrucciones. El número de canales puede ser independiente del número de unidades aritmético-lógicas (ALU) o unidades de coma flotante (FPU) físicas de un procesador de gráficos en particular. En algunas formas de realización, las unidades de ejecución 2608A-2608N admiten tipos de datos enteros y de coma flotante.

El conjunto de instrucciones de unidad de ejecución incluye instrucciones SIMD. Los diversos elementos de datos se pueden almacenar como un tipo de datos encapsulado en un registro y la unidad de ejecución procesará los diversos elementos en función del tamaño de los datos de los elementos. Por ejemplo, cuando se realizan operaciones en un vector de 256 bits de ancho, los 256 bits del vector se almacenan en un registro y la unidad de ejecución realiza operaciones en el vector como cuatro elementos distintos de datos encapsulados de 64 bits (elementos de datos con un tamaño de cuatro palabras (QW)), ocho elementos distintos de datos encapsulados de 32 bits (elementos de datos con un tamaño de dos palabras (DW)), dieciséis elementos distintos de datos encapsulados de 16 bits (elementos de

datos con un tamaño de palabra (W)), o treinta y dos elementos distintos de datos de 8 bits (elementos de datos con un tamaño de octeto (B)). Sin embargo, diferentes anchos de vectores y tamaños de registro son posibles.

Una o más memorias caché de instrucciones internas (por ejemplo, 2606) se incluyen en la lógica de ejecución de subprocesos 2600 para almacenar en memoria caché instrucciones de subprocesos para las unidades de ejecución. En algunas formas de realización, una o más memorias caché de datos (por ejemplo, 2612) se incluyen para almacenar en memoria caché datos de subprocesos durante la ejecución de subprocesos. En algunas formas de realización, se incluye un muestreador 2610 para proporcionar muestreo de textura para operaciones 3D y muestreo de medios para operaciones de medios. En algunas formas de realización, el muestreador 2610 incluye funcionalidad de muestreo de textura o medios especializada para procesar datos de textura o medios durante el proceso de muestreo antes de proporcionar los datos muestreados a una unidad de ejecución.

Durante la ejecución, el procesamiento en cadena de gráficos y medios envían solicitudes de inicio de subprocesos a la lógica de ejecución de subprocesos 2600 a través de la lógica de generación y envío de subprocesos. Una vez que un grupo de objetos geométricos se ha procesado y rasterizado obteniéndose datos de píxeles, se invoca la lógica de procesador de píxeles (por ejemplo, lógica de sombreado de píxeles, lógica de sombreado de fragmentos, etc.) del procesador de sombreado 2602 para calcular adicionalmente información de salida y hacer que los resultados se escriban en superficies de salida (por ejemplo, *búferes* de color, *búferes* de profundidad, *búferes* de estarcido, etc.). En algunas formas de realización, un sombreador de píxeles o sombreador de fragmentos calcula los valores de los diversos atributos de vértice que deben interpolarse a través del objeto rasterizado. En algunas formas de realización, la lógica de procesador de píxeles dentro del procesador de sombreado 2602 se ejecuta entonces en un programa de sombreado de fragmentos o píxeles suministrado por la interfaz de programación de aplicaciones (API). Para ejecutar el programa de sombreado, el procesador de sombreado 2602 envía subprocesos a una unidad de ejecución (por ejemplo, 2608A) a través del distribuidor de subprocesos 2604. En algunas formas de realización, el sombreador de píxeles 2602 utiliza lógica de muestreo de textura en el muestreador 2610 para acceder a datos de textura en mapas de textura almacenados en memoria. Las operaciones aritméticas en los datos de textura y los datos de geometría de entrada calculan datos de color de píxeles para cada fragmento geométrico, o descartan uno o más píxeles de un procesamiento adicional.

En algunas formas de realización, el puerto de datos 2614 proporciona un mecanismo de acceso a memoria para que la lógica de ejecución de subprocesos 2600 proporcione datos procesados a memoria para su procesamiento en un procesamiento en cadena de salida de procesador de gráficos. En algunas formas de realización, el puerto de datos 2614 incluye o está acoplado a una o más memorias caché (por ejemplo, memoria caché de datos 2612) para almacenar en caché datos para acceder a memoria a través del puerto de datos.

La FIG. 27 es un diagrama de bloques que ilustra formatos de instrucciones de procesador de gráficos 2700 de acuerdo con algunas formas de realización. En una o más formas de realización, las unidades de ejecución de procesador de gráficos admiten un conjunto de instrucciones que tiene instrucciones en múltiples formatos. Los recuadros de líneas continuas ilustran los componentes que generalmente se incluyen en una instrucción de unidad de ejecución, mientras que las líneas discontinuas incluyen componentes que son opcionales o que solo se incluyen en un subconjunto de las instrucciones. En algunas formas de realización, el formato de instrucción 2700 descrito e ilustrado son macroinstrucciones, en el sentido de que son instrucciones suministradas a la unidad de ejecución, a diferencia de microoperaciones resultantes de la descodificación de instrucciones una vez que se procesa la instrucción.

En algunas formas de realización, las unidades de ejecución de procesador de gráficos admiten de forma nativa instrucciones en un formato de instrucción de 128 bits 2710. Un formato de instrucción compactado de 64 bits 2730 está disponible para algunas instrucciones en función de la instrucción seleccionada, las opciones de instrucción y el número de operandos. El formato de instrucción nativo de 128 bits 710 proporciona acceso a todas las opciones de instrucción, mientras que algunas opciones y operaciones están restringidas en el formato de 64 bits 2730. Las instrucciones nativas disponibles en el formato de 64 bits 2730 varían según la forma de realización. En algunas formas de realización, la instrucción se compacta, en parte, usando un conjunto de valores de índice en un campo de índice 2713. El hardware de la unidad de ejecución hace referencia a un conjunto de tablas de compactación en función de los valores de índice y utiliza las salidas de la tabla de compactación para reconstruir una instrucción nativa en el formato de instrucción de 128 bits 2710.

Para cada formato, el código de operación de instrucción 2712 define la operación que la unidad de ejecución debe realizar. Las unidades de ejecución ejecutan cada instrucción en paralelo a través de los múltiples elementos de datos de cada operando. Por ejemplo, en respuesta a una instrucción de adición, la unidad de ejecución realiza una operación de adición simultánea a través de cada canal de color que representa un elemento de textura o elemento de imagen. De forma predeterminada, la unidad de ejecución realiza cada instrucción en todos los canales de datos de los operandos. En algunas formas de realización, el campo de control de instrucciones 2714 permite el control sobre determinadas opciones de ejecución, tales como selección de canales (por ejemplo, predicación) y orden de canales de datos (por ejemplo, reorganización de vectores (*swizzle*)). Para las instrucciones en el formato de instrucción de 128 bits 2710, un campo de tamaño de ejecución 2716 limita el número de canales de datos que se

ejecutarán en paralelo. En algunas formas de realización, el campo de tamaño de ejecución 2716 no está disponible para su uso en el formato de instrucción compacto de 64 bits 2730.

Algunas instrucciones de la unidad de ejecución tienen hasta tres operandos, que incluyen dos operandos fuente, src0 2720, src1 2722, y un destino 2718. En algunas formas de realización, las unidades de ejecución admiten instrucciones de doble destino, donde uno de los destinos está implícito. Las instrucciones de manipulación de datos pueden tener un tercer operando fuente (por ejemplo, SRC2 2724), donde el código de operación de instrucción 2712 determina el número de operandos fuente. El último operando fuente de una instrucción puede ser un valor inmediato (por ejemplo, codificado de forma fija) que se pasa con la instrucción.

En algunas formas de realización, el formato de instrucción de 128 bits 2710 incluye un campo de modo de acceso/dirección 2726 que especifica, por ejemplo, si se utiliza el modo de direccionamiento de registro directo o el modo de direccionamiento de registro indirecto. Cuando se utiliza el modo de direccionamiento de registro directo, la dirección de registro de uno o más operandos se proporciona directamente mediante bits en la instrucción.

En algunas formas de realización, el formato de instrucción de 128 bits 2710 incluye un campo de modo de acceso/dirección 2726, que especifica un modo de dirección y/o un modo de acceso para la instrucción. En una forma de realización, el modo de acceso se utiliza para definir una alineación de acceso a datos para la instrucción. Algunas formas de realización admiten modos de acceso que incluyen un modo de acceso alineado de 16 octetos y un modo de acceso alineado de 1 octeto, donde la alineación de octetos del modo de acceso determina la alineación de acceso de los operandos de instrucción. Por ejemplo, en un primer modo, la instrucción puede usar direccionamiento alineado de un octeto para operandos fuente y destino, y en un segundo modo, la instrucción puede usar direccionamiento alineado de 16 octetos para todos los operandos fuente y destino.

En una forma de realización, la parte de modo de dirección del campo de modo de acceso/dirección 2726 determina si la instrucción debe utilizar direccionamiento directo o indirecto. Cuando se utiliza el modo de direccionamiento de registro directo, los bits de la instrucción proporcionan directamente la dirección de registro de uno o más operandos. Cuando se utiliza el modo de direccionamiento de registro indirecto, la dirección de registro de uno o más operandos se puede calcular en función de un valor de registro de dirección y un campo inmediato de dirección en la instrucción.

En algunas formas de realización, las instrucciones se agrupan en función de los campos de bits del código de operación 2712 para simplificar el decodificador de código de operación 2740. Para un código de operación de 8 bits, los bits 4, 5 y 6 permiten que la unidad de ejecución determine el tipo de código de operación. La agrupación de código de operación precisa mostrada es meramente un ejemplo. En algunas formas de realización, un grupo de códigos de operaciones lógicas y de movimiento 2742 incluye instrucciones lógicas y de movimiento de datos (por ejemplo, mover (mov), comparar (cmp)). En algunas formas de realización, el grupo de lógica y movimiento 2742 comparte los cinco bits más significativos (MSB), donde las instrucciones de movimiento (mov) están en forma de 0000xxxxb y las instrucciones lógicas están en forma de 0001xxxxb. Un grupo de instrucciones de control de flujo 2744 (por ejemplo, llamada, salto (jmp)) incluye instrucciones en forma de 0010xxxxb (por ejemplo, 0x20). Un grupo de instrucciones misceláneas 2746 incluye una mezcla de instrucciones, incluidas instrucciones de sincronización (por ejemplo, esperar, enviar) en forma de 0011xxxxb (por ejemplo, 0x30). Un grupo de instrucciones matemáticas paralelo 2748 incluye instrucciones aritméticas orientadas a componentes (por ejemplo, sumar, multiplicar (mul)) en forma de 0100xxxxb (por ejemplo, 0x40). El grupo matemático paralelo 2748 realiza las operaciones aritméticas en paralelo a través de canales de datos. El grupo matemático vectorial 2750 incluye instrucciones aritméticas (por ejemplo, dp4) en forma de 0101xxxxb (por ejemplo, 0x50). El grupo de matemáticas vectoriales realiza cálculos aritméticos tales como cálculos de producto escalar en operandos vectoriales.

Procesamiento en cadena de gráficos ejemplar adicional

La FIG. 28 es un diagrama de bloques de otra forma de realización de un procesador de gráficos 2800. Elementos de la FIG. 28 que tienen los mismos números (o nombres) de referencia que los elementos de cualquier otra figura del presente documento pueden actuar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a esto.

En algunas formas de realización, el procesador de gráficos 2800 incluye un procedimiento en cadena de gráficos 2820, un procesamiento en cadena de medios 2830, un motor de visualización 2840, lógica de ejecución de subprocesos 2850 y un procesamiento en cadena de salida de renderización 2870. En algunas formas de realización, el procesador de gráficos 2800 es un procesador de gráficos de un sistema de procesamiento de múltiples núcleos que incluye uno o más núcleos de procesamiento de propósito general. El procesador de gráficos se controla mediante escrituras de registro en uno o más registros de control (no mostrados) o a través de comandos emitidos al procesador de gráficos 2800 a través de una interconexión de anillo 2802. En algunas formas de realización, la interconexión de anillo 2802 acopla el procesador de gráficos 2800 a otros componentes de procesamiento, tales como otros procesadores de gráficos o procesadores de propósito general. Los comandos de la interconexión de anillo 2802 son interpretados por un transmisor de comandos 2803, que suministra instrucciones a componentes individuales del procesamiento en cadena de gráficos 2820 del procesamiento en cadena de medios 2830.

En algunas formas de realización, el transmisor de comandos 2803 dirige el funcionamiento de una unidad de obtención de vértices 2805 que lee datos de vértices de una memoria y ejecuta comandos de procesamiento de vértices proporcionados por el transmisor de comandos 2803. En algunas formas de realización, la unidad de obtención de vértices 2805 proporciona datos de vértices a un sombreador de vértices 2807, que realiza operaciones de iluminación y transformación de espacio de coordenadas en cada vértice. En algunas formas de realización, la unidad de obtención de vértices 2805 y el sombreador de vértices 2807 ejecutan instrucciones de procesamiento de vértices mediante el envío de subprocesos de ejecución a las unidades de ejecución 2852A-2852B a través de un distribuidor de subprocesos 2831.

En algunas formas de realización, las unidades de ejecución 2852A-2852B son una formación de procesadores vectoriales que tienen un conjunto de instrucciones para realizar operaciones de gráficos y medios. En algunas formas de realización, las unidades de ejecución 2852A-2852B tienen una memoria caché L1 conectada 2851 que es específica para cada formación o está compartida entre las formaciones. La memoria caché se puede configurar como una memoria caché de datos, una memoria caché de instrucciones o una única memoria caché que se particiona para contener datos e instrucciones en diferentes particiones.

En algunas formas de realización, el procesamiento en cadena de gráficos 2820 incluye componentes de teselación para realizar teselación acelerada por hardware de objetos 3D. En algunas formas de realización, un sombreador de casco programable 811 configura las operaciones de teselación. Un sombreador de dominio programable 817 proporciona una evaluación de sección de procesamiento de la salida de teselación. Un teselador 2813 funciona en la dirección del sombreador de casco 2811 y contiene lógica de propósito especial para generar un conjunto de objetos geométricos detallados en función de un modelo geométrico aproximado que se proporciona como datos de entrada al procesamiento en cadena de gráficos 2820. En algunas formas de realización, si no se utiliza teselación, los componentes de teselación (por ejemplo, sombreador de casco 2811, teselador 2813 y sombreador de dominio 2817) se pueden omitir.

En algunas formas de realización, los objetos geométricos completos pueden ser procesados por un sombreador de geometría 2819 a través de uno o más subprocesos enviados a las unidades de ejecución 2852A-2852B, o pueden llevarse directamente a un recortador 2829. En algunas formas de realización, el sombreador de geometría realiza operaciones en objetos geométricos completos, en lugar de vértices o parches de vértices como en fases anteriores del procesamiento en cadena de gráficos. Si la teselación está inhabilitada, el sombreador de geometría 2819 recibe información del sombreador de vértices 2807. En algunas formas de realización, el sombreador de geometría 2819 se puede programar mediante un programa de sombreado de geometría para realizar la teselación de geometría si las unidades de teselación están inhabilitadas.

Antes de la rasterización, un recortador 2829 procesa datos de vértices. El recortador 2829 puede ser un recortador de función fija o un recortador programable que tiene funciones de sombreador de geometría y recorte. En algunas formas de realización, un componente de pruebas de profundidad y rasterización 2873 del procesamiento en cadena de salida de renderización 2870 envía sombreadores de píxeles para convertir los objetos geométricos en sus representaciones por píxel. En algunas formas de realización, lógica de sombreado de píxeles se incluye en lógica de ejecución de subprocesos 2850. En algunas formas de realización, una aplicación puede omitir el componente de pruebas de profundidad y rasterización 2873 y acceder a datos de vértices no rasterizados a través de una unidad de salida de flujo 2823.

El procesador de gráficos 2800 tiene un bus de interconexión, una estructura de interconexión o algún otro mecanismo de interconexión que permite que los datos y los mensajes pasen entre los componentes principales del procesador. En algunas formas de realización, las unidades de ejecución 2852A-2852B y memoria(s) caché asociada(s) 2851, el muestreador de textura y medios 2854 y la memoria caché de textura/muestreador 2858 se interconectan a través de un puerto de datos 2856 para realizar acceso a memoria y comunicarse con componentes de procesamiento en cadena de salida de renderización del procesador. En algunas formas de realización, el muestreador 2854, las memorias caché 2851, 2858 y las unidades de ejecución 2852A-2852B tienen rutas de acceso a memoria distintas.

En algunas formas de realización, el procesamiento en cadena de salida de renderización 2870 contiene un componente de pruebas de profundidad y rasterización 2873 que convierte objetos basados en vértices en una representación basada en píxeles asociada. En algunas formas de realización, la lógica de rasterización incluye una unidad de división en ventanas/enmascaramiento para realizar una rasterización de triángulos y líneas de función fija. Una memoria caché de renderización asociada 2878 y una memoria caché de profundidad 2879 también están disponibles en algunas formas de realización. Un componente de operaciones de píxeles 2877 realiza operaciones basadas en píxeles en los datos, aunque en algunos casos, las operaciones de píxeles asociadas a operaciones 2D (por ejemplo, transferencias de imágenes en bloques de bits con mezcla) se realizan mediante el motor 2D 2841 o se sustituyen en tiempo de visualización por el controlador de visualización 2843 utilizando planos de visualización superpuestos. En algunas formas de realización, una memoria caché L3 compartida 2875 está disponible para todos los componentes de gráficos, lo que permite la compartición de datos sin el uso de memoria de sistema principal.

En algunas formas de realización, el procesamiento en cadena de medios del procesador de gráficos 2830 incluye un motor de medios 2837 y una sección de procesamiento de vídeo 2834. En algunas formas de realización, la sección

de entrada de vídeo 2834 recibe comandos de procesamiento en cadena desde el transmisor de comandos 2803. En algunas formas de realización, el procesamiento en cadena de medios 2830 incluye otro transmisor de comandos. En algunas formas de realización, la sección de entrada de vídeo 2834 procesa comandos de medios antes de enviar el comando al motor de medios 2837. En algunas formas de realización, el motor de medios 2837 incluye funcionalidad de generación de subprocesos para generar subprocesos para su envío a la lógica de ejecución de subprocesos 2850 a través del distribuidor de subprocesos 2831.

En algunas formas de realización, el procesador de gráficos 2800 incluye un motor de visualización 2840. En algunas formas de realización, el motor de visualización 2840 es externo al procesador 2800 y está acoplado al procesador de gráficos a través de la interconexión de anillo 2802, o algún otro bus o estructura de interconexión. En algunas formas de realización, el motor de visualización 2840 incluye un motor 2D 2841 y un controlador de visualización 2843. En algunas formas de realización, el motor de visualización 2840 contiene lógica de propósito especial capaz de funcionar independientemente del procesamiento en cadena 3D. En algunas formas de realización, el controlador de visualización 2843 está acoplado a un dispositivo de visualización (no mostrado), que puede ser un dispositivo de visualización integrado en sistema, como en un ordenador portátil, o un dispositivo de visualización externo conectado a través de un conector de dispositivo de visualización.

En algunas formas de realización, el procesamiento en cadena de gráficos 2820 y el procesamiento en cadena de medios 2830 pueden configurarse para realizar operaciones en función de múltiples interfaces de programación de medios y gráficos y no son específicos de ninguna interfaz de programación de aplicaciones (API). En algunas formas de realización, el software de controlador para el procesador de gráficos convierte llamadas a API que son específicas de un gráfico o biblioteca de medios particular en comandos que pueden ser procesados por el procesador de gráficos. En algunas formas de realización, se proporciona soporte para la biblioteca de gráficos abierta (OpenGL), el lenguaje de computación abierto (OpenCL) y/o la API de gráficos y computación Vulkan, todos del Grupo Khronos. En algunas formas de realización, también se puede proporcionar soporte para la biblioteca Direct3D de Microsoft Corporation. En algunas formas de realización, se puede admitir una combinación de estas bibliotecas. También se puede proporcionar soporte para la biblioteca de visión por ordenador de código abierto (OpenCV). También se admitiría una futura API con un procesamiento en cadena 3D compatible si se puede hacer una correlación del procesamiento en cadena de la futura API con el procesamiento en cadena del procesador de gráficos.

Programación de procesamiento en cadena de gráficos

La FIG. 29A es un diagrama de bloques que ilustra un formato de comando de procesador de gráficos 2900 de acuerdo con algunas formas de realización. La FIG. 29B es un diagrama de bloques que ilustra una secuencia de comandos de procesador de gráficos 2910 de acuerdo con una forma de realización. Los recuadros de línea continua de la FIG. 29A ilustran los componentes que generalmente se incluyen en un comando de gráficos, mientras que las líneas discontinuas incluyen componentes que son opcionales o que solo se incluyen en un subconjunto de los comandos de gráficos. El formato de comando de procesador de gráficos 2900 ejemplar de la FIG. 29A incluye campos de datos para identificar un cliente objetivo 2902 del comando, un código de operación (*opcode*) de comando 2904 y los datos pertinentes 2906 para el comando. Un subcódigo 2905 y un tamaño de comando 2908 también se incluyen en algunos comandos.

En algunas formas de realización, el cliente 2902 especifica la unidad de cliente del dispositivo de gráficos que procesa los datos de comando. En algunas formas de realización, un analizador de comandos de procesador de gráficos examina el campo de cliente de cada comando para determinar un procesamiento adicional del comando y encaminar los datos de comando a la unidad de cliente apropiada. En algunas formas de realización, las unidades cliente de procesador de gráficos incluyen una unidad de interfaz de memoria, una unidad de renderización, una unidad 2D, una unidad 3D y una unidad de medios. Cada unidad cliente tiene un procesamiento en cadena correspondiente que procesa los comandos. Una vez que la unidad cliente recibe el comando, la unidad cliente lee el código de operación 2904 y, si está presente, el subcódigo de operación 2905 para determinar la operación a realizar. La unidad cliente lleva a cabo el comando utilizando información en el campo de datos 2906. Para algunos comandos, se espera que un tamaño de comando explícito 2908 especifique el tamaño del comando. En algunas formas de realización, el analizador de comandos determina automáticamente el tamaño de al menos algunos de los comandos en función del código de operación de comando. En algunas formas de realización, los comandos se alinean a través de múltiplos de una palabra doble.

El diagrama de flujo de la FIG. 29B muestra una secuencia de comandos de procesador de gráficos 2910 ejemplar. En algunas formas de realización, el software o firmware de un sistema de procesamiento de datos que presenta una forma de realización de un procesador de gráficos utiliza una versión de la secuencia de comandos mostrada para configurar, ejecutar y finalizar un conjunto de operaciones de gráficos. Se muestra y describe una secuencia de comandos de muestra solo a modo de ejemplo ya que las formas de realización no están limitadas a estos comandos específicos o a esta secuencia de comandos. Además, los comandos se pueden emitir como un lote de comandos en una secuencia de comandos, de modo que el procesador de gráficos procesará la secuencia de comandos en al menos una concurrencia parcial.

En algunas formas de realización, la secuencia de comandos de procesador de gráficos 2910 puede comenzar con un comando de vaciado de procesamiento en cadena 2912 para hacer que cualquier procesamiento en cadena de gráficos activa complete los comandos actualmente pendientes del procesamiento en cadena. En algunas formas de realización, el procesamiento en cadena 3D 2922 y el procesamiento en cadena de medios 2924 no se producen simultáneamente. El vaciado del procesamiento en cadena se realiza para hacer que el procesamiento en cadena de gráficos activo complete los comandos pendientes. En respuesta a un vaciado de procesamiento en cadena, el analizador de comandos para el procesador de gráficos detendrá el procesamiento de comandos hasta que los motores de dibujo activos completen las operaciones pendientes y se invaliden las memorias caché de lectura pertinentes. Opcionalmente, cualquier dato en la memoria caché de renderización que esté marcado como "sucio" se puede llevar a la memoria. En algunas formas de realización, el comando de vaciado de procesamiento en cadena 2912 se puede utilizar para la sincronización de procesamientos en cadena o antes de hacer que el procesador de gráficos pase un estado de baja potencia.

En algunas formas de realización, se utiliza un comando de selección de procesamiento en cadena 2913 cuando una secuencia de comandos requiere que el procesador de gráficos conmute explícitamente entre procesamientos en cadena. En algunas formas de realización, un comando de selección de procesamiento en cadena 2913 solo se requiere una vez dentro de un contexto de ejecución antes de emitir comandos de procesamiento en cadena a menos que el contexto sea emitir comandos para ambos procesamientos en cadena. En algunas formas de realización, un comando de vaciado de procesamiento en cadena 2912 se requiere inmediatamente antes de un conmutador de procesamiento en cadena a través del comando de selección de procesamiento en cadena 2913.

En algunas formas de realización, un comando de control de procesamiento en cadena 2914 configura un procesamiento en cadena de gráficos para su funcionamiento y se utiliza para programar el procesamiento en cadena 3D 2922 y el procesamiento en cadena de medios 2924. En algunas formas de realización, el comando de control de procesamiento en cadena 2914 configura el estado de procesamiento en cadena para el procesamiento en cadena activo. En una forma de realización, el comando de control de procesamiento en cadena 2914 se utiliza para la sincronización de procesamientos en cadena y para borrar datos de una o más memorias caché del procesamiento en cadena activo antes de procesar un lote de comandos.

En algunas formas de realización, comandos de estado de *búfer* de retorno 2916 se utilizan para configurar un conjunto de *búferes* de retorno para que los respectivos procesamientos en cadena escriban datos. Algunas operaciones de procesamiento en cadena requieren la asignación, selección o configuración de uno o más *búferes* de retorno en los que las operaciones escriben datos intermedios durante el procesamiento. En algunas formas de realización, el procesador de gráficos también utiliza uno o más *búferes* de retorno para almacenar datos de salida y para realizar una comunicación entre subprocesos cruzados. En algunas formas de realización, el estado de *búfer* de retorno 2916 incluye seleccionar el tamaño y el número de *búferes* de retorno a utilizar para un conjunto de operaciones de procesamiento en cadena.

Los comandos restantes en la secuencia de comandos difieren en función del procesamiento en cadena activo para las operaciones. En función de una determinación de procesamiento en cadena 2920, la secuencia de comandos se adapta al procesamiento en cadena 3D 2922 comenzando con el estado de procesamiento en cadena 3D 2930 o al procesamiento en cadena de medios 2924 comenzando en el estado de procesamiento en cadena de medios 2940.

Los comandos para configurar el estado de procesamiento en cadena 3D 2930 incluyen comandos de configuración de estado 3D para el estado de *búfer* de vértices, el estado de elemento de vértice, el estado de color constante, el estado de *búfer* de profundidad y otras variables de estado que deben configurarse antes de que se procesen comandos de primitivas 3D. Los valores de estos comandos se determinan, al menos en parte, en función de la API 3D particular en uso. En algunas formas de realización, los comandos de estado de procesamiento en cadena 3D 2930 también son capaces de inhabilitar u omitir de manera selectiva determinados elementos de procesamiento en cadena si esos elementos no van a utilizarse.

En algunas formas de realización, el comando de primitivas 3D 2932 se utiliza para enviar primitivas 3D que serán procesadas mediante el procesamiento en cadena 3D. Los comandos y parámetros asociados que se pasan al procesador de gráficos a través del comando de primitivas 3D 2932 se reenvían a la función de obtención de vértices en el procesamiento en cadena de gráficos. La función de obtención de vértices utiliza los datos de comando de primitivas 3D 2932 para generar estructuras de datos de vértices. Las estructuras de datos de vértices se almacenan en uno o más *búferes* de retorno. En algunas formas de realización, el comando de primitivas 3D 2932 se utiliza para realizar operaciones de vértices en primitivas 3D mediante sombreadores de vértices. Para procesar sombreadores de vértices, el procesamiento en cadena 3D 2922 envía subprocesos de ejecución de sombreado a unidades de ejecución de procesador de gráficos.

En algunas formas de realización, el procesamiento en cadena 3D 2922 se activa a través de un comando o evento de ejecución 2934. En algunas formas de realización, una escritura de registro activa la ejecución de comandos. En algunas formas de realización, la ejecución se activa mediante un comando 'go' o 'kick' en la secuencia de comandos. En una forma de realización, la ejecución de comandos se activa utilizando un comando de sincronización de procesamiento en cadena para vaciar la secuencia de comandos a través del procesamiento en cadena de gráficos.

El procesamiento en cadena 3D realizará el procesamiento de geometría para las primitivas 3D. Una vez que se completan las operaciones, los objetos geométricos resultantes se rasterizan y el motor de píxeles colorea los píxeles resultantes. También se pueden incluir comandos adicionales para controlar el sombreado de píxeles y las operaciones de sección de procesamiento de píxeles para esas operaciones.

En algunas formas de realización, la secuencia de comandos de procesador de gráficos 2910 sigue la ruta del procesamiento en cadena de medios 2924 cuando se realizan operaciones de medios. En general, el uso específico y la forma de programación del procesamiento en cadena de medios 2924 dependen de las operaciones de medios o de cálculo a realizar. Las operaciones específicas de descodificación de medios pueden descargarse hacia el procesamiento en cadena de medios durante la descodificación de medios. En algunas formas de realización, el procesamiento en cadena de medios también se puede omitir y la descodificación de medios se puede realizar en su totalidad o en parte utilizando recursos proporcionados por uno o más núcleos de procesamiento de propósito general. En una forma de realización, el procesamiento en cadena de medios también incluye elementos para operaciones de unidad de procesador de gráficos de propósito general (GPGPU), donde el procesador de gráficos se utiliza para realizar operaciones vectoriales SIMD utilizando programas de sombreado computacional que no están explícitamente relacionados con la renderización de primitivas de gráficos.

En algunas formas de realización, el procesamiento en cadena de medios 2924 está configurado de manera similar al procesamiento en cadena 3D 2922. Un conjunto de comandos para configurar el estado de procesamiento en cadena de medios 2940 se envían a o se ponen en una cola de comandos antes de comandos de objetos de medios 2942. En algunas formas de realización, comandos de estado de procesamiento en cadena de medios 2940 incluyen datos para configurar los elementos de procesamiento en cadena de medios que se utilizarán para procesar los objetos de medios. Esto incluye datos para configurar la lógica de descodificación de vídeo y de codificación de vídeo del procesamiento en cadena de medios, tal como el formato de codificación o descodificación. En algunas formas de realización, los comandos de estado de procesamiento en cadena de medios 2940 también admiten el uso de uno o más punteros a elementos de estado "indirecto" que contienen un lote de configuraciones de estado.

En algunas formas de realización, los comandos de objetos de medios 2942 suministran punteros a objetos de medios para su procesamiento mediante el procesamiento en cadena de medios. Los objetos de medios incluyen *búferes* de memoria que contienen datos de vídeo que deben procesarse. En algunas formas de realización, todos los estados de procesamiento en cadena de medios deben ser válidos antes de emitir un comando de objeto de medios 2942. Una vez que se configura el estado del procesamiento en cadena y se ponen en cola comandos de objetos de medios 2942, el procesamiento en cadena de medios 2924 se activa a través de un comando de ejecución 2944 o un evento de ejecución equivalente (por ejemplo, escritura de registro). La salida del procesamiento en cadena de medios 2924 se puede procesar posteriormente mediante operaciones proporcionadas por el procesamiento en cadena 3D 2922 o el procesamiento en cadena de medios 2924. En algunas formas de realización, las operaciones de GPGPU se configuran y ejecutan de manera similar a las operaciones de medios.

Arquitectura de software de gráficos

La FIG. 30 ilustra una arquitectura de software de gráficos ejemplar de un sistema de procesamiento de datos 3000 de acuerdo con algunas formas de realización. En algunas formas de realización, la arquitectura de software incluye una aplicación de gráficos 3D 3010, un sistema operativo 3020 y al menos un procesador 3030. En algunas formas de realización, el procesador 3030 incluye un procesador de gráficos 3032 y uno o más núcleos de procesador de propósito general 3034. Tanto la aplicación de gráficos 3010 como el sistema operativo 3020 se ejecutan en la memoria de sistema 3050 del sistema de procesamiento de datos.

En algunas formas de realización, la aplicación de gráficos 3D 3010 contiene uno o más programas de sombreado que incluyen instrucciones de sombreado 3012. Las instrucciones de lenguaje de sombreado pueden estar en un lenguaje de sombreado de alto nivel, tal como el lenguaje de sombreado de alto nivel (HLSL) o el lenguaje de sombreado OpenGL (GLSL). La solicitud también incluye instrucciones ejecutables 3014 en un lenguaje máquina adecuado para su ejecución mediante el núcleo de procesador de propósito general 3034. La solicitud también incluye objetos gráficos 3016 definidos por datos de vértices.

En algunas formas de realización, el sistema operativo 3020 es un sistema operativo Microsoft® Windows® de Microsoft Corporation, un sistema operativo tipo UNIX propietario o un sistema operativo tipo UNIX de código abierto que utiliza una variante del núcleo de Linux. El sistema operativo 3020 puede admitir una API de gráficos 3022, tal como la API Direct3D, la API OpenGL o la API Vulkan. Cuando la API Direct3D está en uso, el sistema operativo 3020 utiliza un compilador de sombreado de sección de entrada 3024 para compilar cualquier instrucción de sombreado 3012 de HLSL en un lenguaje de sombreado de nivel inferior. La compilación puede ser una compilación "justo a tiempo" (JIT) o la aplicación puede realizar una compilación previa de sombreador. En algunas formas de realización, los sombreadores de alto nivel se compilan en sombreadores de bajo nivel durante la compilación de la aplicación de gráficos 3D 3010. En algunas formas de realización, las instrucciones de sombreado 3012 se proporcionan en una forma intermedia, tal como una versión de la representación intermedia portátil estándar (SPIR) utilizada por el API Vulkan.

En algunas formas de realización, el controlador de gráficos de modo de usuario 3026 contiene un compilador de sombreado de sección de procesamiento 3027 para convertir las instrucciones de sombreado 3012 en una representación específica de hardware. Cuando se usa la API OpenGL, las instrucciones de sombreado 3012 en lenguaje de alto nivel de GLSL se pasan a un controlador de gráficos de modo de usuario 3026 para su compilación. En algunas formas de realización, el controlador de gráficos de modo de usuario 3026 utiliza funciones de modo de núcleo de sistema operativo 3028 para comunicarse con un controlador de gráficos de modo de núcleo 3029. En algunas formas de realización, el controlador de gráficos de modo de núcleo 3029 se comunica con el procesador de gráficos 3032 para enviar comandos e instrucciones.

Implementaciones de núcleos IP

Uno o más aspectos de al menos una forma de realización se pueden implementar mediante código representativo almacenado en un medio legible por máquina que representa y/o define lógica dentro de un circuito integrado, tal como un procesador. Por ejemplo, el medio legible por máquina puede incluir instrucciones que representan lógica diversa dentro del procesador. Cuando son leídas por una máquina, las instrucciones pueden hacer que la máquina genere la lógica para realizar las técnicas descritas en el presente documento. Dichas representaciones, conocidas como "núcleos IP", son unidades lógicas reutilizables para un circuito integrado que puede almacenarse en un medio tangible y legible por máquina como un modelo de hardware que describe la estructura del circuito integrado. El modelo de hardware puede suministrarse a varios clientes o instalaciones de fabricación, que cargan el modelo de hardware en máquinas de fabricación que fabrican el circuito integrado. El circuito integrado se puede fabricar de manera que el circuito realice operaciones descritas en asociación con cualquiera de las formas de realización descritas en el presente documento.

La FIG. 31 es un diagrama de bloques que ilustra un sistema de desarrollo de núcleos IP 3100 que se puede utilizar para fabricar un circuito integrado para realizar operaciones de acuerdo con una forma de realización. El sistema de desarrollo de núcleos IP 3100 se puede utilizar para generar diseños modulares y reutilizables que se pueden incorporar en un diseño más grande o se pueden utilizar para construir un circuito integrado completo (por ejemplo, un circuito integrado SOC). Una instalación de diseño 3130 puede generar una simulación de software 3110 de un diseño de núcleo IP en un lenguaje de programación de alto nivel (por ejemplo, C/C++). La simulación de software 3110 se puede utilizar para diseñar, probar y verificar el comportamiento del núcleo IP utilizando un modelo de simulación 3112. El modelo de simulación 3112 puede incluir simulaciones funcionales, de comportamiento y/o de temporización. Un diseño de nivel de transferencia de registro (RTL) 3115 se puede crear o sintetizar a partir del modelo de simulación 3112. El diseño RTL 3115 es una abstracción del comportamiento del circuito integrado que modela el flujo de señales digitales entre los registros de hardware, incluida la lógica asociada realizada utilizando las señales digitales modeladas. Además de un diseño RTL 3115, también se pueden crear, diseñar o sintetizar diseños de nivel inferior a nivel lógico o de transistor. Por lo tanto, los detalles particulares del diseño inicial y la simulación pueden variar.

El diseño RTL 3115, o equivalente, puede sintetizarse adicionalmente por la instalación de diseño en un modelo de hardware 3120, que puede estar en un lenguaje de descripción de hardware (HDL), o alguna otra representación de datos de diseño físico. El HDL se puede simular o probar adicionalmente para verificar el diseño de núcleos IP. El diseño de núcleos IP se puede almacenar para proporcionarse a una instalación de fabricación de terceros 3165 utilizando memoria no volátil 3140 (por ejemplo, disco duro, memoria flash o cualquier medio de almacenamiento no volátil). De manera alternativa, el diseño de núcleos IP puede transmitirse (por ejemplo, a través de Internet) por medio de una conexión por cable 3150 o una conexión inalámbrica 3160. La instalación de fabricación 3165 puede entonces fabricar un circuito integrado que está basado, al menos en parte, en el diseño de núcleos IP. El circuito integrado fabricado puede configurarse para realizar operaciones de acuerdo con al menos una forma de realización descrita en el presente documento.

Circuito integrado de sistema en chip ejemplar

Las FIGS. 32-34 ilustran circuitos integrados y procesadores de gráficos asociados ejemplares que pueden fabricarse utilizando uno o más núcleos IP, de acuerdo con diversas formas de realización descritas en el presente documento. Además de lo que se ilustra, se pueden incluir otras lógicas y circuitos, que incluyen procesadores/núcleos de gráficos adicionales, controladores de interfaz periférica o núcleos de procesador de propósito general.

La FIG. 32 es un diagrama de bloques que ilustra un circuito integrado de sistema en chip 3200 ejemplar que se puede fabricar utilizando uno o más núcleos IP, de acuerdo con una forma de realización. El circuito integrado 3200 ejemplar incluye uno o más procesadores de aplicación 3205 (por ejemplo, CPU), al menos un procesador de gráficos 3210 y puede incluir adicionalmente un procesador de imágenes 3215 y/o un procesador de vídeo 3220, cualquiera de los cuales puede ser un núcleo IP modular de los mismos o múltiples instalaciones de diseño diferentes. El circuito integrado 3200 incluye lógica periférica o de bus que incluye un controlador USB 3225, un controlador UART 3230, un controlador SPI/SDIO 3235 y un controlador I^2S/I2C 3^240. Además, el circuito integrado puede incluir un dispositivo de visualización 3245 acoplado a uno o más de un controlador de interfaz multimedia de alta definición (HDMI) 3250 y una interfaz de visualización de interfaz de procesador de la industria móvil (MIPI) 3255. El almacenamiento puede proporcionarse mediante un subsistema de memoria flash 3260 que incluye memoria flash y un controlador de

memoria flash. La interfaz de memoria puede proporcionarse a través de un controlador de memoria 3265 para acceder a dispositivos de memoria SDRAM o SRAM. Algunos circuitos integrados incluyen adicionalmente un motor de seguridad integrado 3270.

La FIG. 33 es un diagrama de bloques que ilustra un procesador de gráficos 3310 ejemplar de un sistema en un circuito integrado en chip que se puede fabricar utilizando uno o más núcleos IP, de acuerdo con una forma de realización. El procesador de gráficos 3310 puede ser una variante del procesador de gráficos 3210 de la FIG. 32. El procesador de gráficos 3310 incluye un procesador de vértices 3305 y uno o más procesadores de fragmentos 3315A-3315N (por ejemplo, 3315A, 3315B, 3315C, 3315D hasta 3315N-1 y 3315N). El procesador de gráficos 3310 puede ejecutar diferentes programas de sombreado a través de una lógica distinta, de modo que el procesador de vértices 3305 esté optimizado para ejecutar operaciones para programas de sombreado de vértices, mientras que el uno o más procesadores de fragmentos 3315A-3315N ejecutan operaciones de sombreado de fragmentos (por ejemplo, píxeles) para programas de sombreado de fragmentos o píxeles. El procesador de vértices 3305 realiza la fase de procesamiento de vértices del procesamiento en cadena de gráficos 3D y genera datos de primitivas y vértices. El/los procesador(es) de fragmentos 3315A-3315N utilizan los datos de primitivas y vértices generados por el procesador de vértices 3305 para producir un *búfer* de fotogramas que se muestra en un dispositivo de visualización. En una forma de realización, el/los procesador(es) de fragmentos 3315A-3315N están optimizados para ejecutar programas de sombreado de fragmentos según lo dispuesto en la API OpenGL, que puede utilizarse para realizar operaciones similares a un programa de sombreado de píxeles según lo dispuesto en la API Direct 3D.

El procesador de gráficos 3310 incluye adicionalmente una o más unidades de gestión de memoria (MMU) 3320A-3320B, una o más memorias caché 3325A-3325B y una o más interconexiones de circuito 3330A-3330B. La una o más MMU 3320A-3320B proporcionan una correlación entre direcciones virtuales y físicas para el circuito integrado 3310, así como para el procesador de vértices 3305 y/o el/los procesador(es) de fragmentos 3315A-3315N, que pueden hacer referencia a datos de vértices o de imágenes/texturas almacenados en memoria, además de a datos de vértices o imágenes/texturas almacenados en la una o más memorias caché 3325A-3325B. En una forma de realización, la una o más MMU 3325A-3325B se pueden sincronizar con otras MMU dentro del sistema, incluidas una o más MMU asociadas al uno o más procesadores de aplicación 3205, al procesador de imágenes 3215 y/o al procesador de vídeo 3220 de la FIG. 32, de modo que cada procesador 3205-3220 puede participar en un sistema de memoria virtual compartida o unificada. La una o más interconexiones de circuito 3330A-3330B permiten que el procesador de gráficos 3310 interactúe con otros núcleos IP dentro del SoC, ya sea a través de un bus interno del SoC o a través de una conexión directa, de acuerdo con las formas de realización.

La FIG. 34 es un diagrama de bloques que ilustra un procesador de gráficos 3410 ejemplar adicional de un sistema en un circuito integrado en chip que se puede fabricar utilizando uno o más núcleos IP, de acuerdo con una forma de realización. El procesador de gráficos 3410 puede ser una variante del procesador de gráficos 3210 de la FIG. **32.** El procesador de gráficos 3410 incluye la una o más MMU 3320A-3320B, memorias caché 3325A-3325B e interconexiones de circuito 3330A-3330B del circuito integrado 3300 de la FIG. 33.

El procesador de gráficos 3410 incluye uno o más núcleos de sombreado 3415A-3415N (por ejemplo, 3415A, 3415B, 3415C, 3415D, 3415E, 3415F hasta 3415N-1 y 3415N), que proporcionan una arquitectura de núcleo de sombreado unificada en la que un solo núcleo o tipo o núcleo puede ejecutar todos los tipos de código de sombreado programable, incluido el código de programa de sombreado, para implementar sombreadores de vértices, sombreadores de fragmentos y/o sombreadores de cálculo. El número exacto de núcleos de sombreado presentes puede variar entre formas de realización e implementaciones. Además, el procesador de gráficos 3410 incluye un gestor de tareas entre núcleos 3405, que actúa como un distribuidor de subprocesos para enviar subprocesos de ejecución a uno o más núcleos de sombreado 3415A-3415N y una unidad de mosaico 3418 para acelerar las operaciones de mosaico para la renderización basada en mosaicos, donde las operaciones de renderización para una escena se subdividen en espacio de imagen, por ejemplo, para aprovechar la coherencia espacial local dentro de una escena o para optimizar el uso de memorias caché internas.

Las formas de realización descritas en el presente documento hacen referencia a configuraciones específicas de hardware, tales como circuitos integrados específicos de la aplicación (ASIC), configuradas para realizar determinadas operaciones o que tienen una funcionalidad predeterminada. Dichos dispositivos electrónicos incluyen típicamente un conjunto de uno o más procesadores acoplados a otro u otros componentes, tales como uno o más dispositivos de almacenamiento (medios de almacenamiento legibles por máquina no transitorios), dispositivos de entrada/salida de usuario (por ejemplo, un teclado, una pantalla táctil y/o un dispositivo de visualización) y conexiones de red. El acoplamiento del conjunto de procesadores y otros componentes es típicamente a través de uno o más buses y puentes (también denominados controladores de bus). El dispositivo de almacenamiento y las señales que transportan el tráfico de red representan, respectivamente, uno o más medios de almacenamiento legibles por máquina y medios de comunicación legibles por máquina. Por lo tanto, los dispositivos de almacenamiento de un dispositivo electrónico dado almacenan típicamente código y/o datos para su ejecución en el conjunto de uno o más procesadores de ese dispositivo electrónico.

Evidentemente, una o más partes de una forma de realización se pueden implementar utilizando diferentes combinaciones de software, firmware y/o hardware. A lo largo de esta descripción detallada, a efectos de explicación,

se han establecido numerosos detalles específicos para proporcionar un entendimiento exhaustivo de la presente invención. Sin embargo, para un experto en la técnica resultará evidente que las formas de realización se pueden llevar a la práctica sin algunos de estos detalles específicos. En determinados casos, estructuras y funciones ampliamente conocidas no se han descrito en detalle para no complicar la materia objeto inventiva de las formas de realización. Por consiguiente, el alcance de la invención debe determinarse en los términos de las siguientes reivindicaciones.

REIVINDICACIONES

1. Una unidad de procesamiento de gráficos de propósito general (214), que incluye:

un multiprocesador de transmisión continua (234, 1400) que tiene una arquitectura de tipo "una sola instrucción, múltiples subprocesos", SIMT, que incluye múltiples subprocesos de hardware, donde el multiprocesador de transmisión continua (234, 1400) comprende:
múltiples conjuntos de unidades de cálculo (1411-1418), presentando cada unidad de cálculo (1411-1418) una unidad lógica de coma flotante (1411B - 1418B) configurada para realizar operaciones de coma flotante y una unidad lógica de números enteros (1411A - 1418A) configurada para realizar operaciones de números enteros; y
una memoria (270, 272) acoplada a los múltiples conjuntos de unidades de cálculo,
caracterizada por que en una unidad de cálculo, la unidad lógica de números enteros está habilitada para ejecutar un subproceso de una primera instrucción, mientras que la unidad lógica de coma flotante está habilitada para ejecutar un subproceso de una segunda instrucción, siendo la segunda instrucción diferente de la primera instrucción y ejecutándose el subproceso de la primera instrucción simultáneamente con el subproceso de la segunda instrucción.

2. La unidad de procesamiento de gráficos de propósito general (214) de la reivindicación 1, en la que la memoria (270, 272) está compartida entre los múltiples conjuntos de unidades de cálculo.

3. La unidad de procesamiento de gráficos de propósito general (214) de la reivindicación 2, en la que la memoria (270, 272) incluye una memoria caché de datos accesible por al menos los múltiples conjuntos de unidades de cálculo.

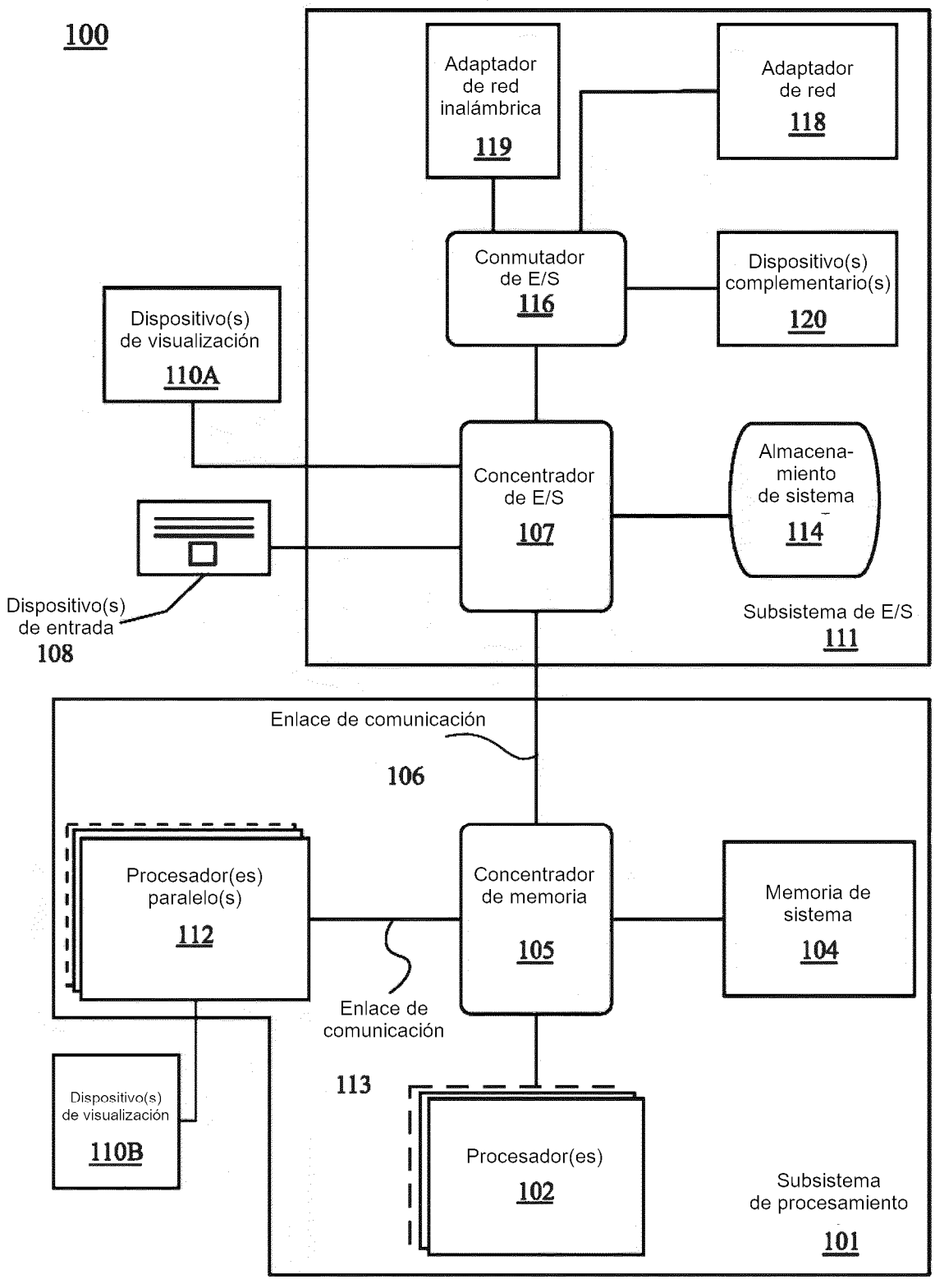

100
Adaptador de red inalámbrica 119
Adaptador de red 118
Conmutador de E/S 116
Dispositivo(s) complementario(s) 120
Dispositivo(s) de visualización 110A
Concentrador de E/S 107
Almacena-miento de sistema 114
Dispositivo(s) de entrada 108
Subsistema de E/S 111
Enlace de comunicación 106
Procesador(es) paralelo(s) 112
Concentrador de memoria 105
Memoria de sistema 104
Enlace de comunicación 113
Dispositivo(s) de visualización 110B
Procesador(es) 102
Subsistema de procesamiento 101

FIG. 1

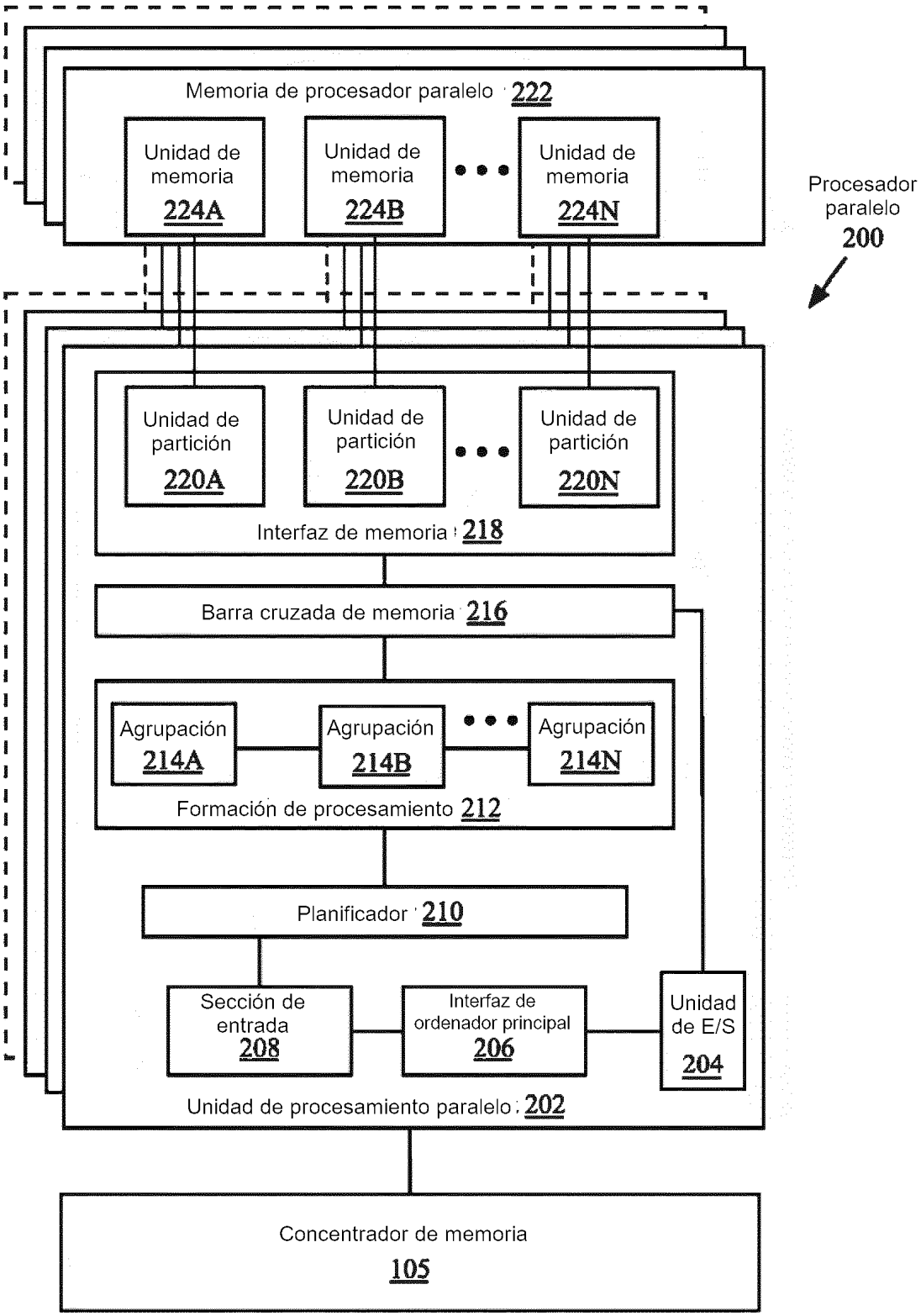

Memoria de procesador paralelo 222
Unidad de memoria 224A
Unidad de memoria 224B
Unidad de memoria 224N
Procesador paralelo 200
Unidad de partición 220A
Unidad de partición 220B
Unidad de partición 220N
Interfaz de memoria 218
Barra cruzada de memoria 216
Agrupación 214A
Agrupación 214B
Agrupación 214N
Formación de procesamiento 212
Planificador 210
Sección de entrada 208
Interfaz de ordenador principal 206
Unidad de E/S 204
Unidad de procesamiento paralelo 202
Concentrador de memoria 105

FIG. 2A

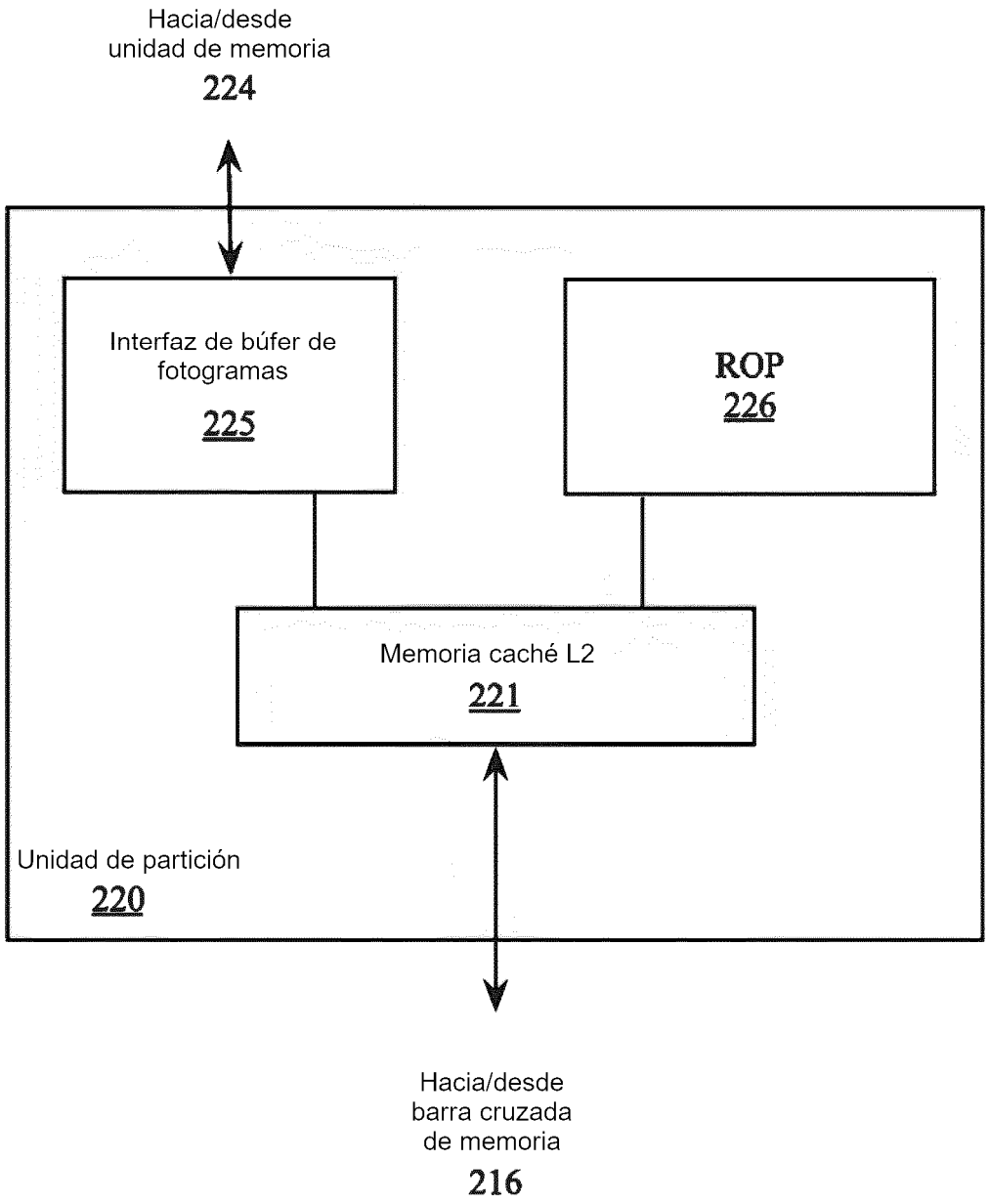

Hacia/desde
unidad de memoria
224
Interfaz de búfer de
fotogramas
225
ROP
226
Memoria caché L2
221
Unidad de partición
220
Hacia/desde
barra cruzada
de memoria
216

FIG. 2B

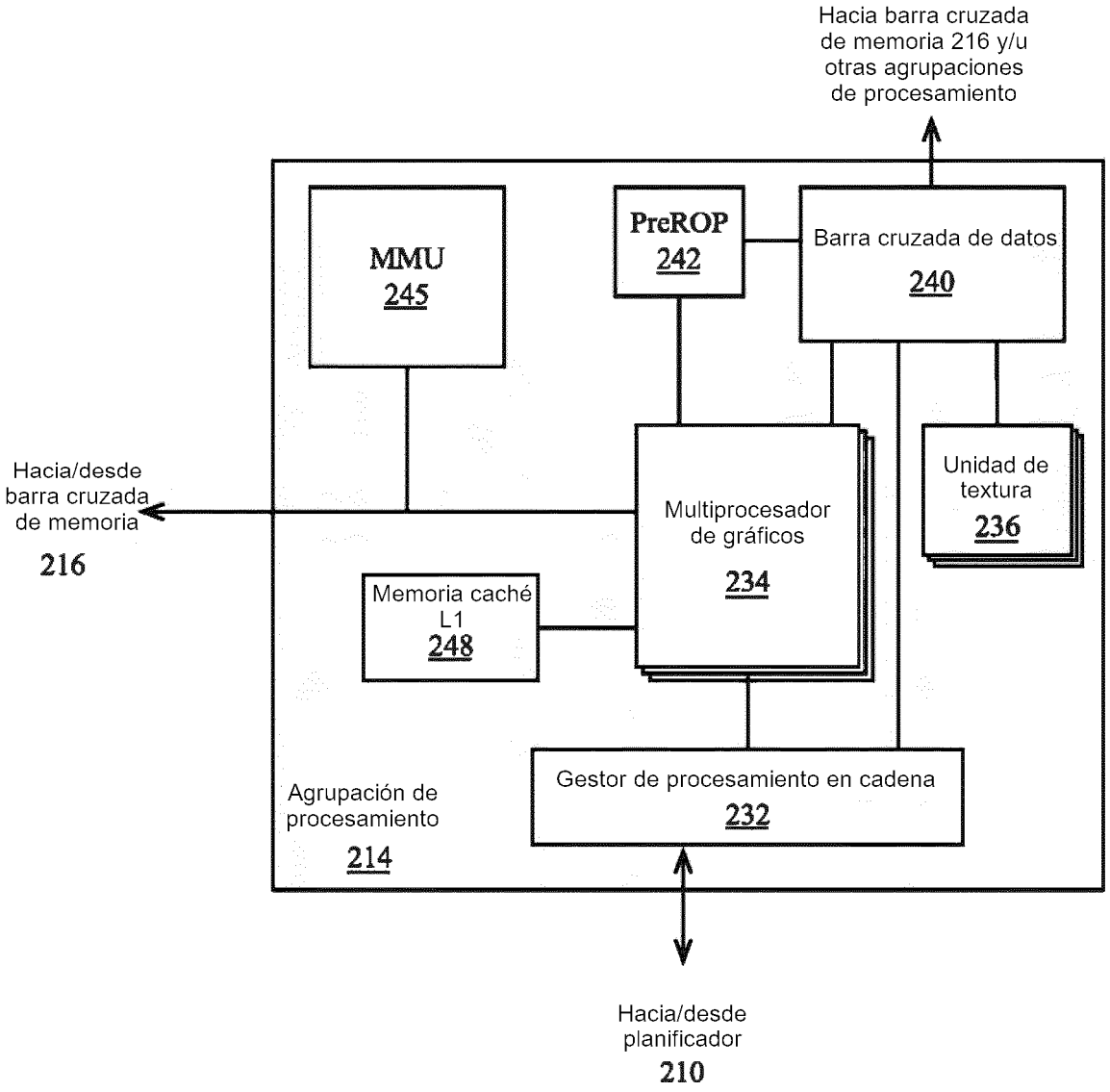

Hacia barra cruzada de memoria 216 y/u otras agrupaciones de procesamiento
MMU 245
PreROP 242
Barra cruzada de datos 240
Hacia/desde barra cruzada de memoria 216
Multiprocesador de gráficos 234
Unidad de textura 236
Memoria caché L1 248
Gestor de procesamiento en cadena 232
Agrupación de procesamiento 214
Hacia/desde planificador 210

FIG. 2C

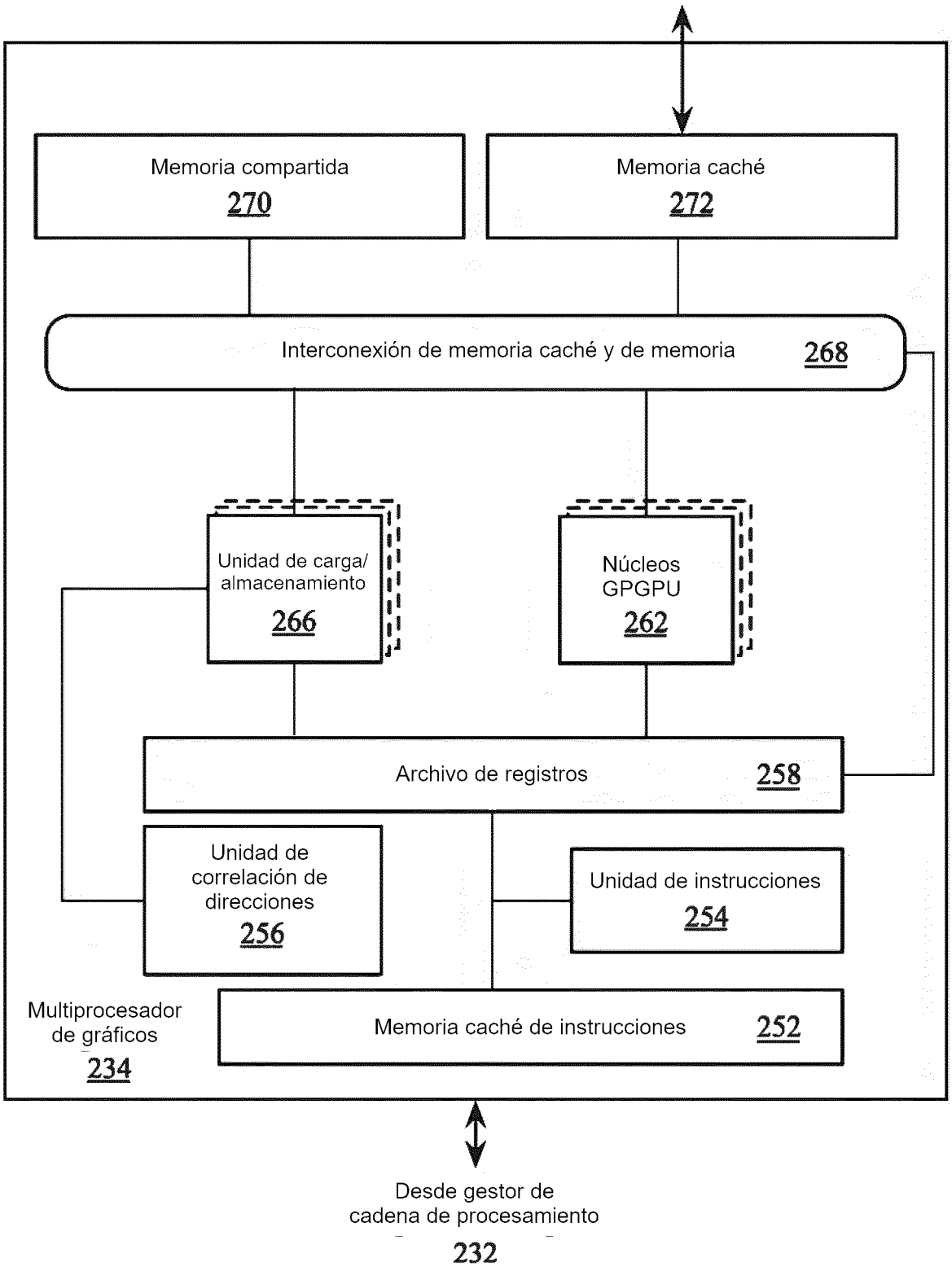

Memoria compartida
270
Memoria caché
272
Interconexión de memoria caché y de memoria
268
Unidad de carga/
almacenamiento
266
Núcleos
GPGPU
262
Archivo de registros
258
Unidad de
correlación de
direcciones
256
Unidad de instrucciones
254
Multiprocesador
de gráficos
234
Memoria caché de instrucciones
252
Desde gestor de
cadena de procesamiento
232

FIG. 2D

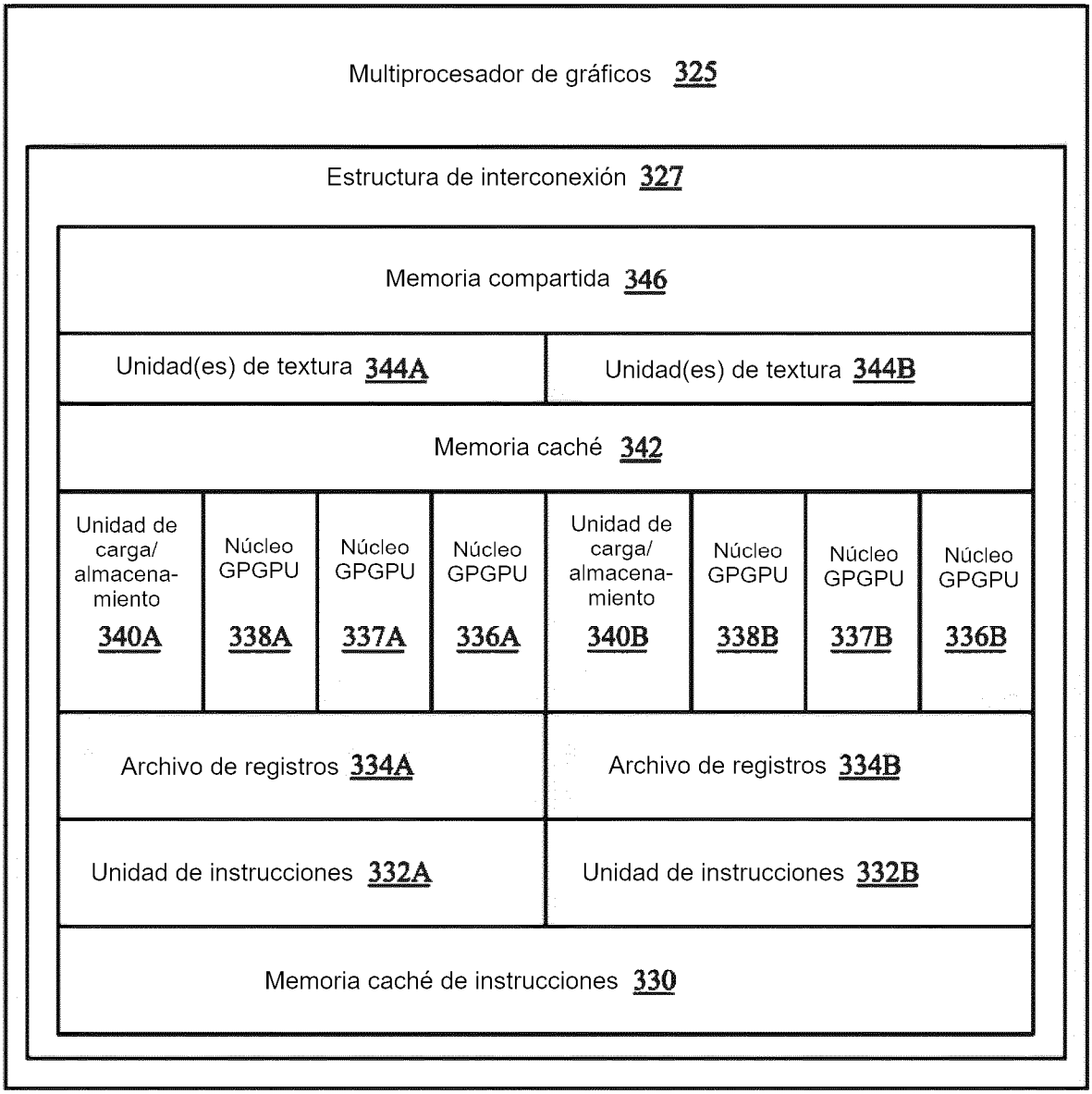

Multiprocesador de gráficos 325
Estructura de interconexión 327
Memoria compartida 346
Unidad(es) de textura 344A
Unidad(es) de textura 344B
Memoria caché 342
Unidad de carga/ almacena-miento 340A
Núcleo GPGPU 338A
Núcleo GPGPU 337A
Núcleo GPGPU 336A
Unidad de carga/ almacena-miento 340B
Núcleo GPGPU 338B
Núcleo GPGPU 337B
Núcleo GPGPU 336B
Archivo de registros 334A
Archivo de registros 334B
Unidad de instrucciones 332A
Unidad de instrucciones 332B
Memoria caché de instrucciones 330

FIG. 3A

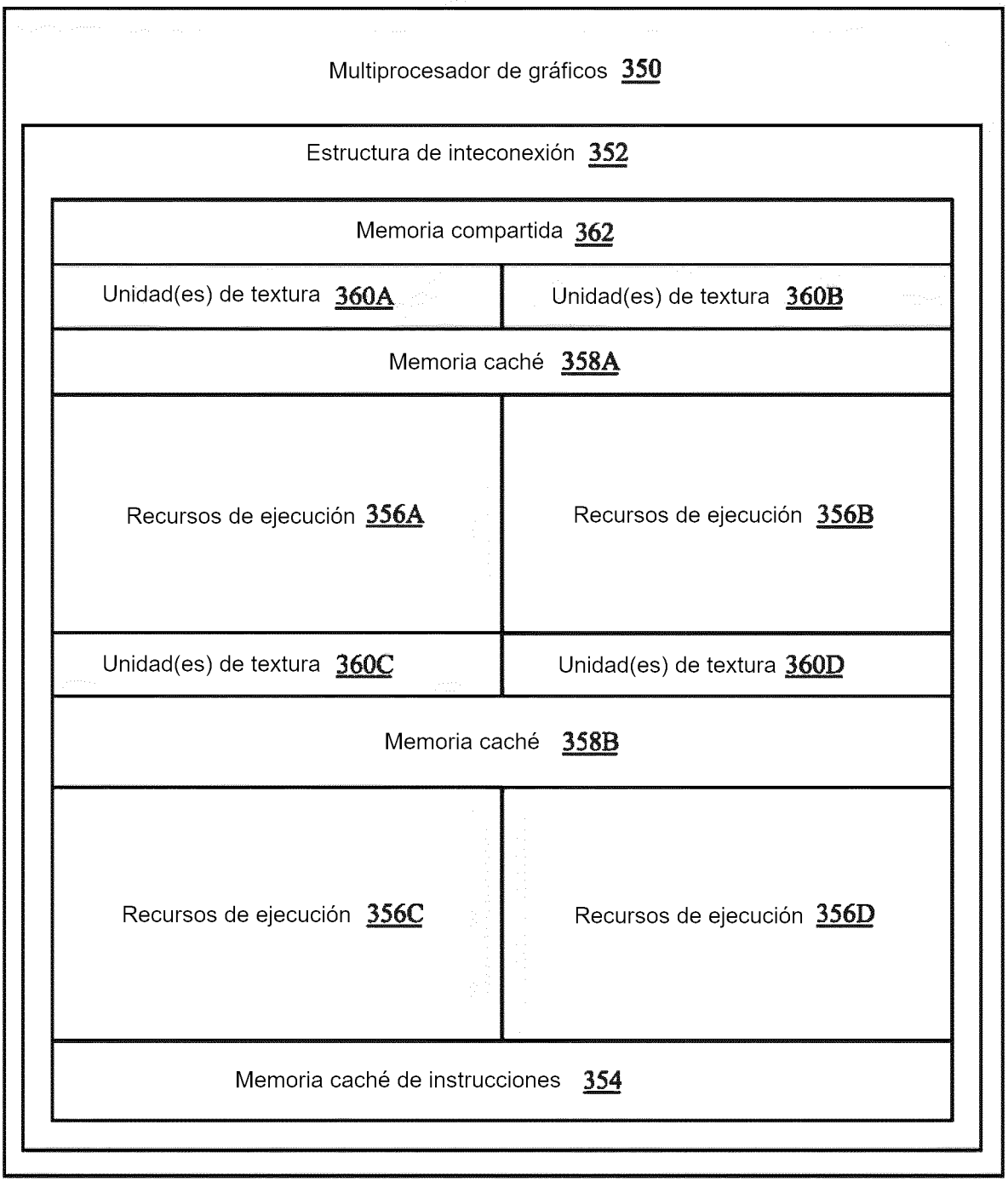

Multiprocesador de gráficos 350
Estructura de inteconexión 352
Memoria compartida 362
Unidad(es) de textura 360A
Unidad(es) de textura 360B
Memoria caché 358A
Recursos de ejecución 356A
Recursos de ejecución 356B
Unidad(es) de textura 360C
Unidad(es) de textura 360D
Memoria caché 358B
Recursos de ejecución 356C
Recursos de ejecución 356D
Memoria caché de instrucciones 354

FIG. 3B

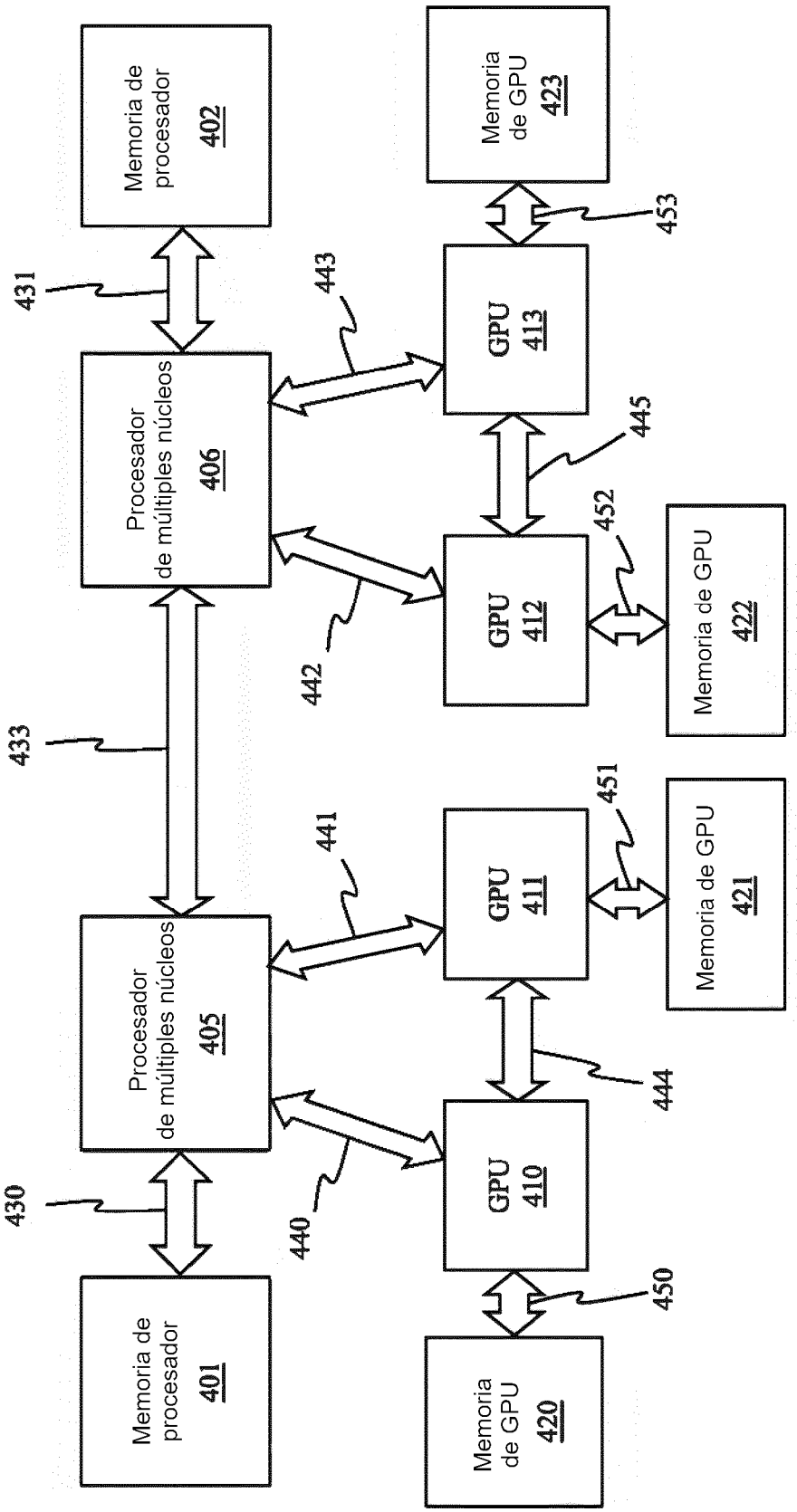

Memoria de procesador 402
Memoria de GPU 423
431
443
453
GPU 413
Procesador de múltiples núcleos 406
445
452
GPU 412
Memoria de GPU 422
442
433
441
451
GPU 411
Memoria de GPU 421
Procesador de múltiples núcleos 405
444
430
440
GPU 410
450
Memoria de procesador 401
Memoria de GPU 420

FIG. 4A

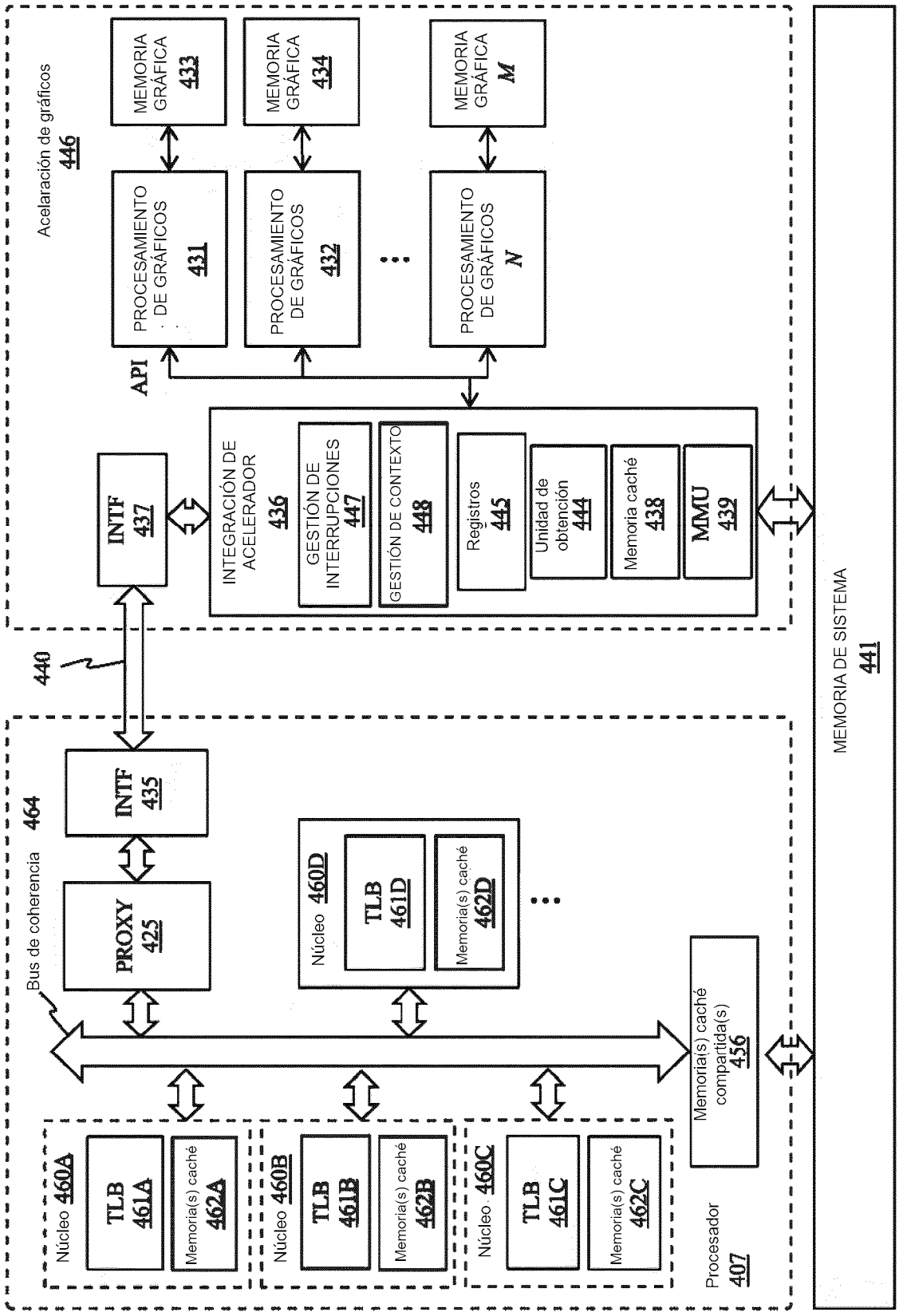

Acelaración de gráficos 446
MEMORIA GRÁFICA 433
MEMORIA GRÁFICA 434
MEMORIA GRÁFICA M
PROCESAMIENTO DE GRÁFICOS 431
PROCESAMIENTO DE GRÁFICOS 432
PROCESAMIENTO DE GRÁFICOS N
API
INTF 437
INTEGRACIÓN DE ACELERADOR 436
GESTIÓN DE INTERRUPCIONES 447
GESTIÓN DE CONTEXTO 448
Registros 445
Unidad de obtención 444
Memoria caché 438
MMU 439
440
Bus de coherencia 464
INTF 435
PROXY 425
Núcleo 460D
TLB 461D
Memoria(s) caché 462D
Memoria(s) caché compartida(s) 456
Núcleo 460A
TLB 461A
Memoria(s) caché 462A
Núcleo 460B
TLB 461B
Memoria(s) caché 462B
Núcleo 460C
TLB 461C
Memoria(s) caché 462C
Procesador 407
MEMORIA DE SISTEMA 441

FIG. 4B

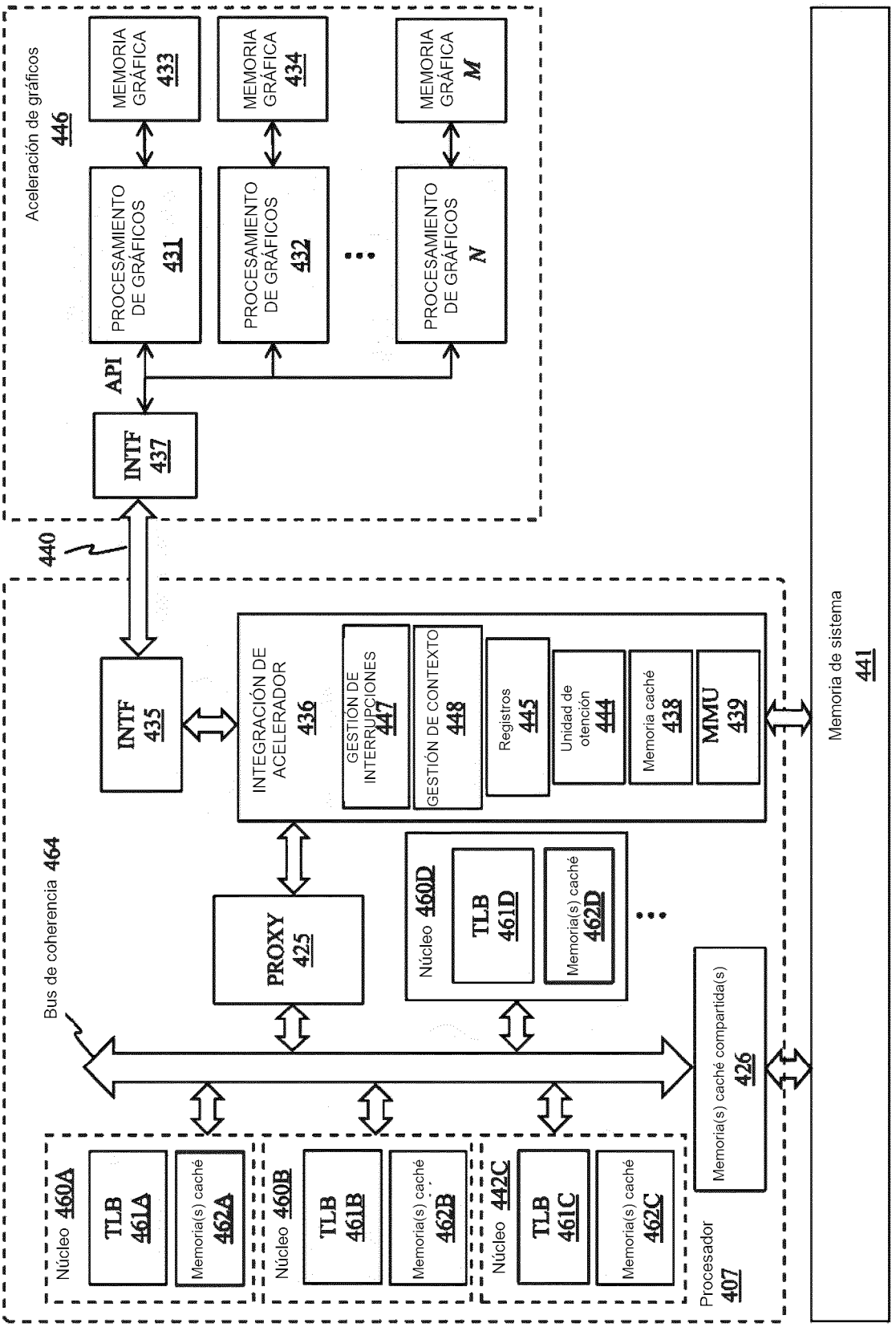

Aceleración de gráficos 446
MEMORIA GRÁFICA 433
MEMORIA GRÁFICA 434
MEMORIA GRÁFICA M
PROCESAMIENTO DE GRÁFICOS 431
PROCESAMIENTO DE GRÁFICOS 432
PROCESAMIENTO DE GRÁFICOS N
API
INTF 437
440
INTF 435
INTEGRACIÓN DE ACELERADOR 436
GESTIÓN DE INTERRUPCIONES 447
GESTIÓN DE CONTEXTO 448
Registros 445
Unidad de otención 444
Memoria caché 438
MMU 439
Memoria de sistema 441
Bus de coherencia 464
PROXY 425
Núcleo 460D
TLB 461D
Memoria(s) caché 462D
Memoria(s) caché compartida(s) 426
Núcleo 460A
TLB 461A
Memoria(s) caché 462A
Núcleo 460B
TLB 461B
Memoria(s) caché 462B
Núcleo 442C
TLB 461C
Memoria(s) caché 462C
Procesador 407

FIG. 4C

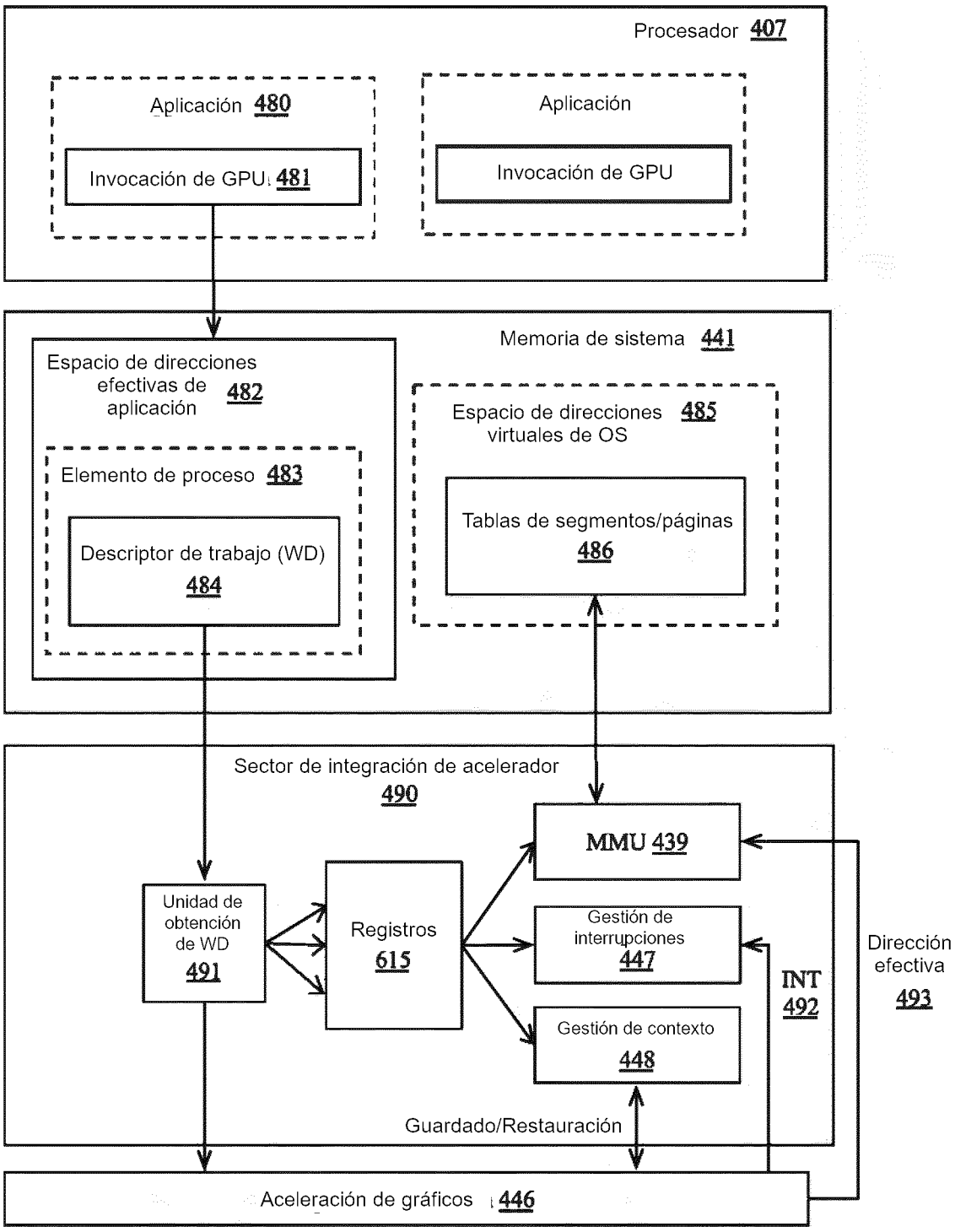

Procesador 407
Aplicación 480
Invocación de GPU 481
Aplicación
Invocación de GPU
Memoria de sistema 441
Espacio de direcciones efectivas de aplicación 482
Elemento de proceso 483
Descriptor de trabajo (WD) 484
Espacio de direcciones virtuales de OS 485
Tablas de segmentos/páginas 486
Sector de integración de acelerador 490
MMU 439
Unidad de obtención de WD 491
Registros 615
Gestión de interrupciones 447
Gestión de contexto 448
INT 492
Dirección efectiva 493
Guardado/Restauración
Aceleración de gráficos 446

FIG. 4D

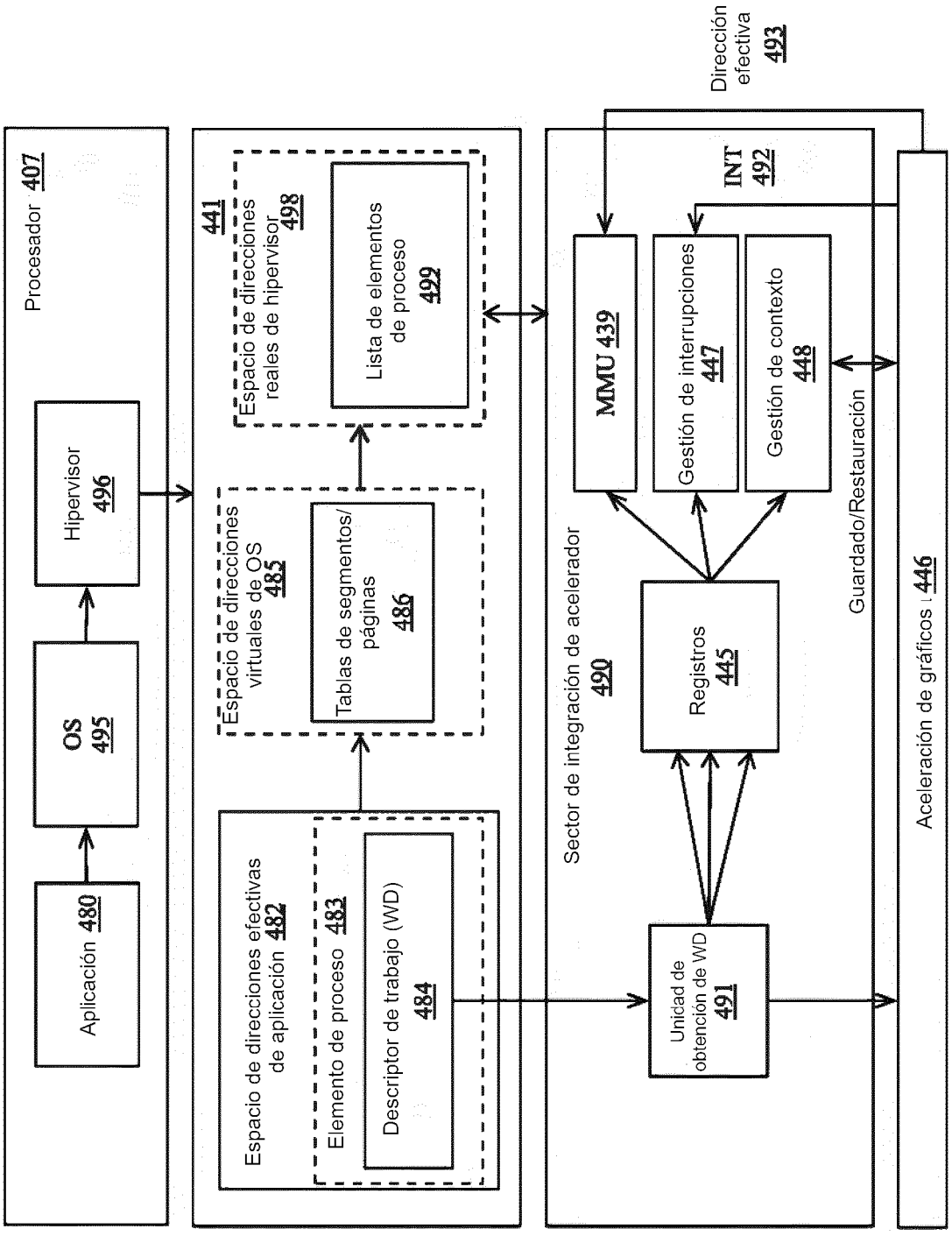

Procesador 407
Aplicación 480
OS 495
Hipervisor 496
441
Espacio de direcciones efectivas de aplicación 482
Elemento de proceso 483
Descriptor de trabajo (WD) 484
Espacio de direcciones virtuales de OS 485
Tablas de segmentos/ páginas 486
Espacio de direcciones reales de hipervisor 498
Lista de elementos de proceso 499
Sector de integración de acelerador 490
Unidad de obtención de WD 491
Registros 445
MMU 439
INT 492
Gestión de interrupciones 447
Gestión de contexto 448
Guardado/Restauración
Dirección efectiva 493
Aceleración de gráficos 446

FIG. 4E

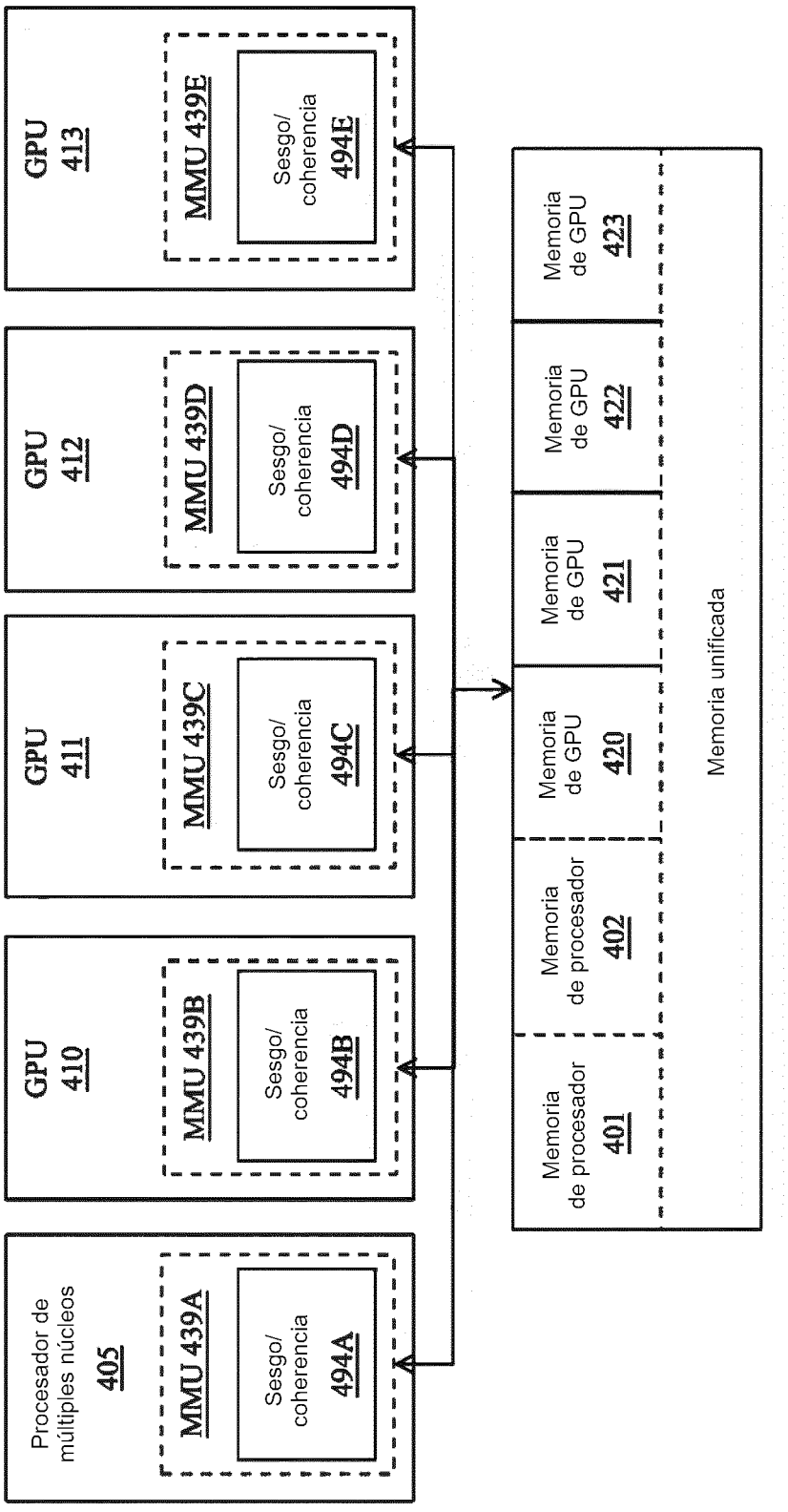

Procesador de múltiples núcleos
405
MMU 439A
Sesgo/ coherencia
494A
GPU
410
MMU 439B
Sesgo/ coherencia
494B
GPU
411
MMU 439C
Sesgo/ coherencia
494C
GPU
412
MMU 439D
Sesgo/ coherencia
494D
GPU
413
MMU 439E
Sesgo/ coherencia
494E
Memoria de procesador
401
Memoria de procesador
402
Memoria de GPU
420
Memoria de GPU
421
Memoria de GPU
422
Memoria de GPU
423
Memoria unificada

FIG. 4F

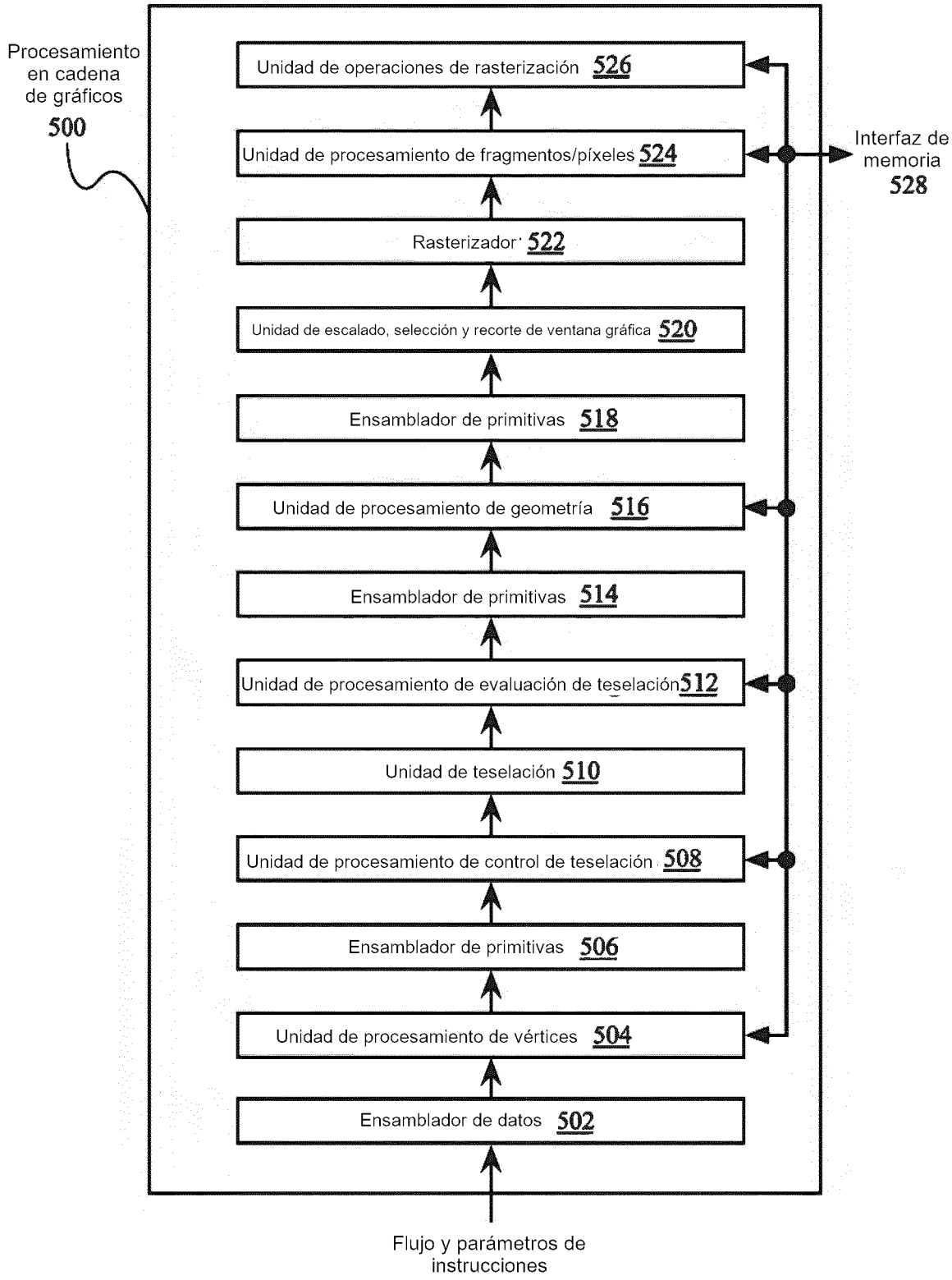

Procesamiento en cadena de gráficos
500
Unidad de operaciones de rasterización 526
Unidad de procesamiento de fragmentos/píxeles 524
Rasterizador 522
Unidad de escalado, selección y recorte de ventana gráfica 520
Ensamblador de primitivas 518
Unidad de procesamiento de geometría 516
Ensamblador de primitivas 514
Unidad de procesamiento de evaluación de teselación 512
Unidad de teselación 510
Unidad de procesamiento de control de teselación 508
Ensamblador de primitivas 506
Unidad de procesamiento de vértices 504
Ensamblador de datos 502
Interfaz de memoria 528
Flujo y parámetros de instrucciones

FIG. 5

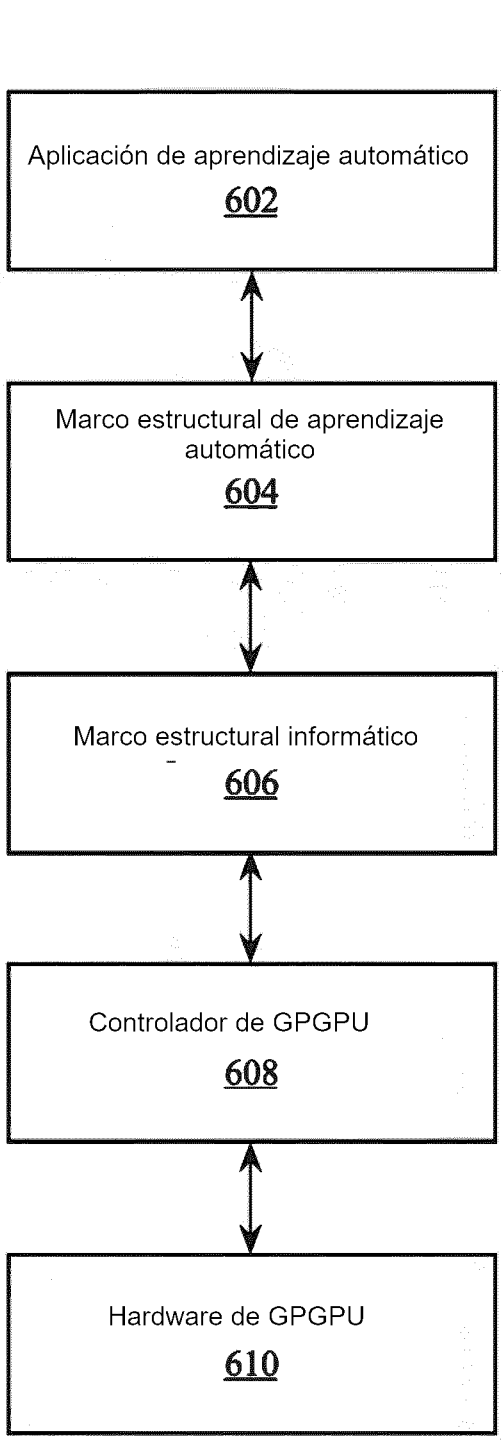

600
Aplicación de aprendizaje automático
602
Marco estructural de aprendizaje automático
604
Marco estructural informático
606
Controlador de GPGPU
608
Hardware de GPGPU
610

FIG. 6

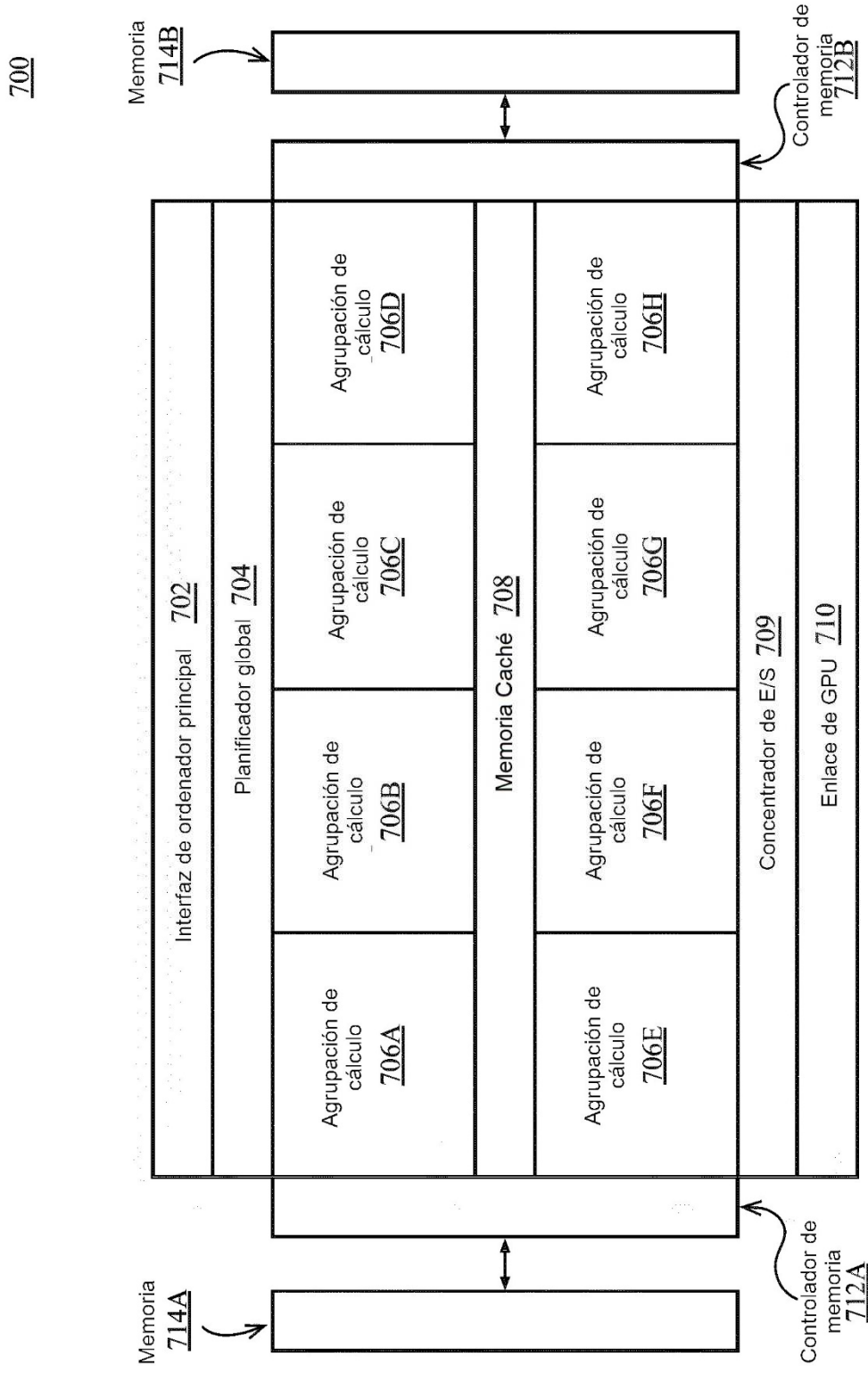

700
Memoria 714B
Controlador de memoria 712B
Interfaz de ordenador principal 702
Planificador global 704
Agrupación de cálculo 706A
Agrupación de cálculo 706B
Agrupación de cálculo 706C
Agrupación de cálculo 706D
Memoria Caché 708
Agrupación de cálculo 706E
Agrupación de cálculo 706F
Agrupación de cálculo 706G
Agrupación de cálculo 706H
Concentrador de E/S 709
Enlace de GPU 710
Memoria 714A
Controlador de memoria 712A

FIG. 7

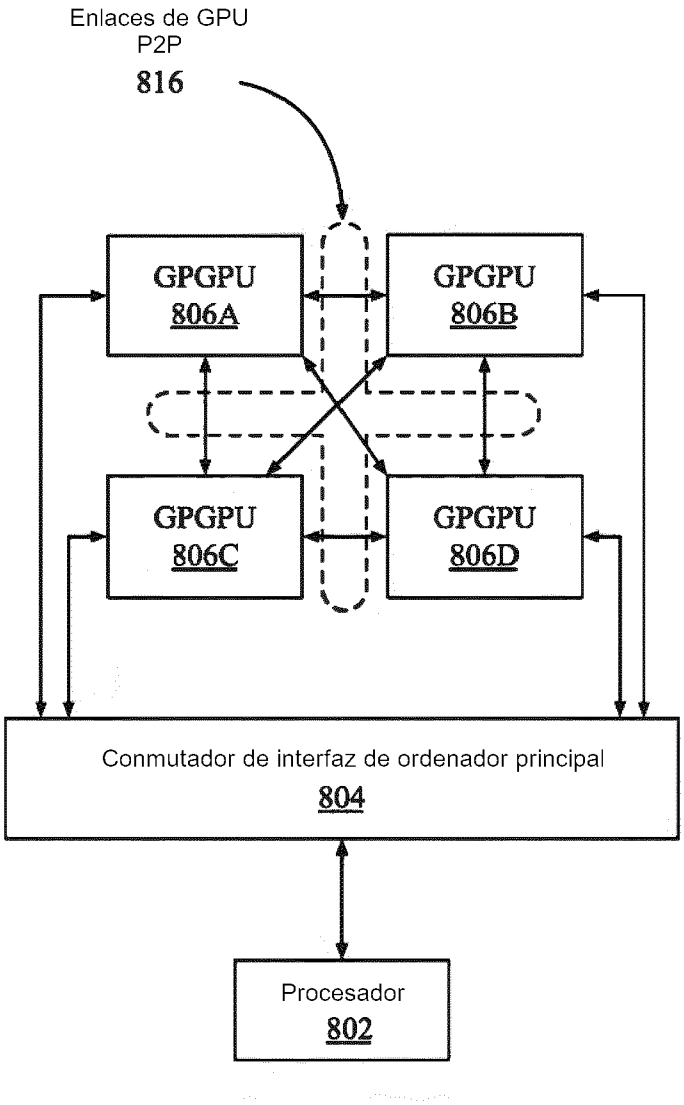

Enlaces de GPU
P2P
816
GPGPU
806A
GPGPU
806B
GPGPU
806C
GPGPU
806D
Conmutador de interfaz de ordenador principal
804
Procesador
802

FIG. 8

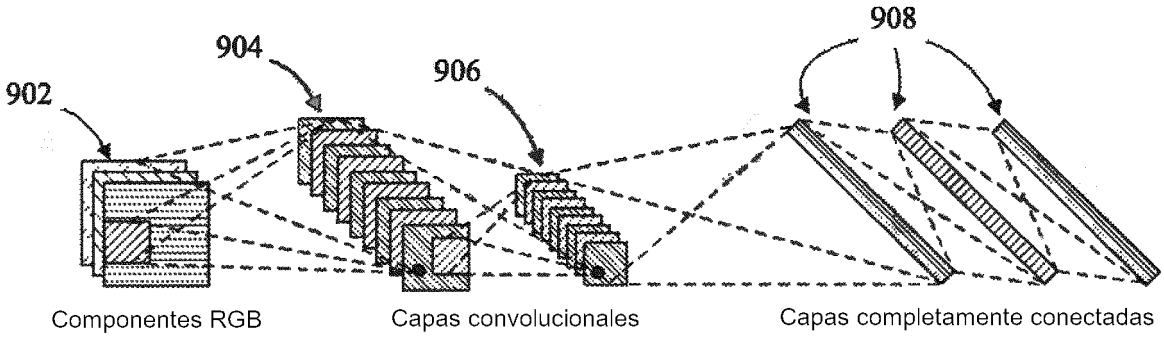

902
904
906
908
Componentes RGB
Capas convolucionales
Capas completamente conectadas

FIG. 9A

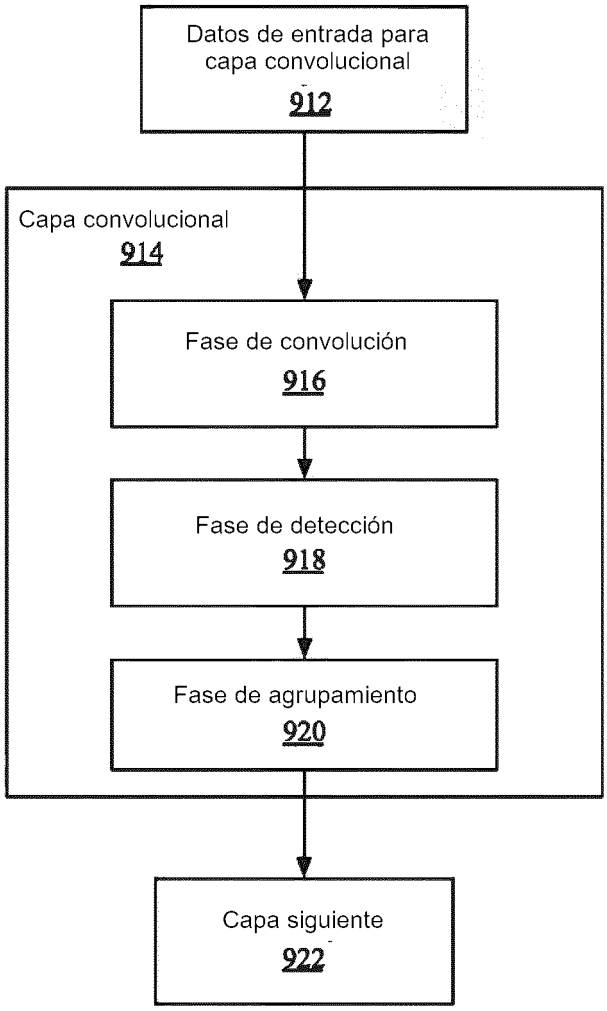

Datos de entrada para capa convolucional
912
Capa convolucional
914
Fase de convolución
916
Fase de detección
918
Fase de agrupamiento
920
Capa siguiente
922

FIG. 9B

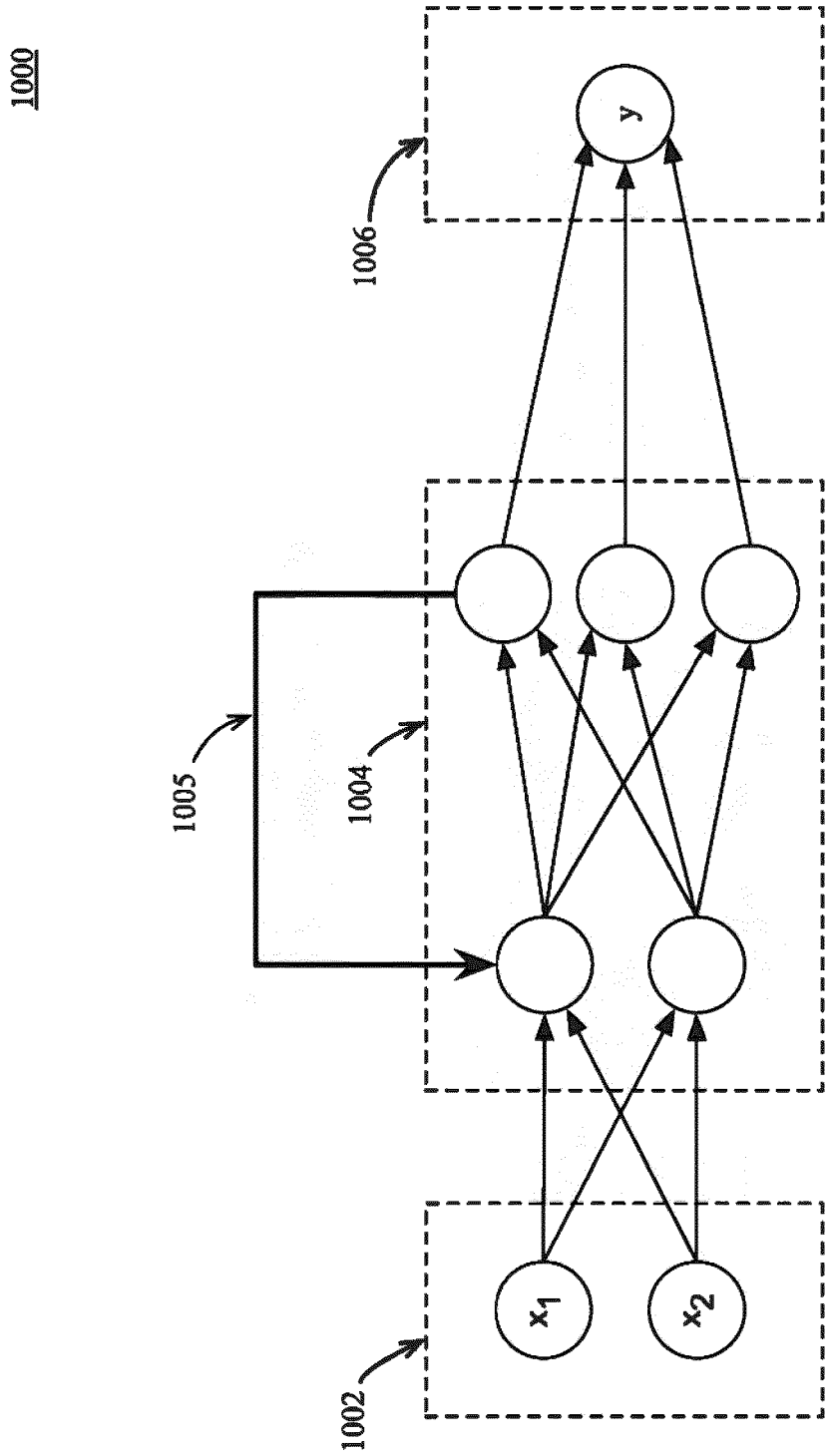


FIG. 10

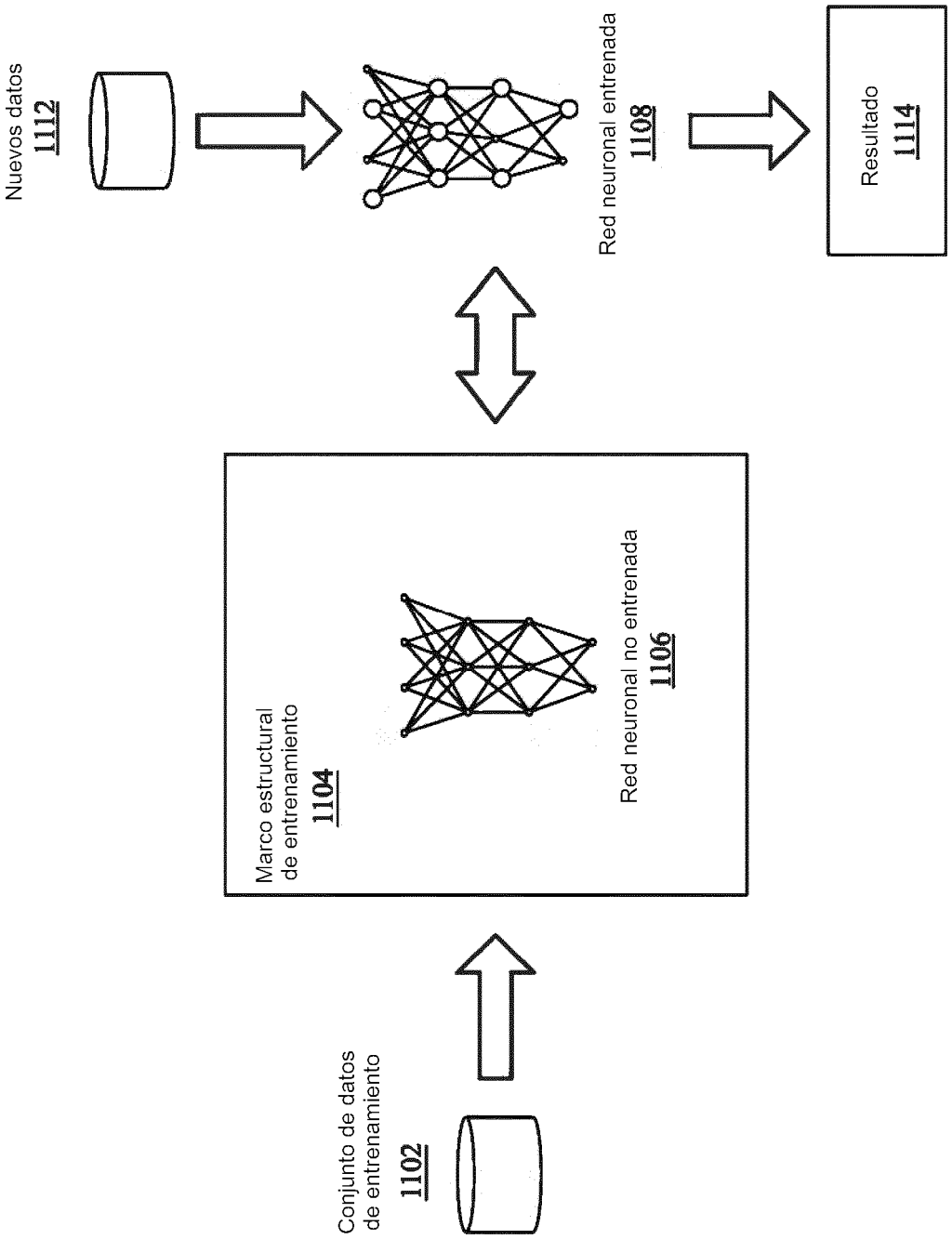

Nuevos datos
1112
Red neuronal entrenada
1108
Resultado
1114
Marco estructural
de entrenamiento
1104
Red neuronal no entrenada
1106
Conjunto de datos
de entrenamiento
1102

FIG. 11

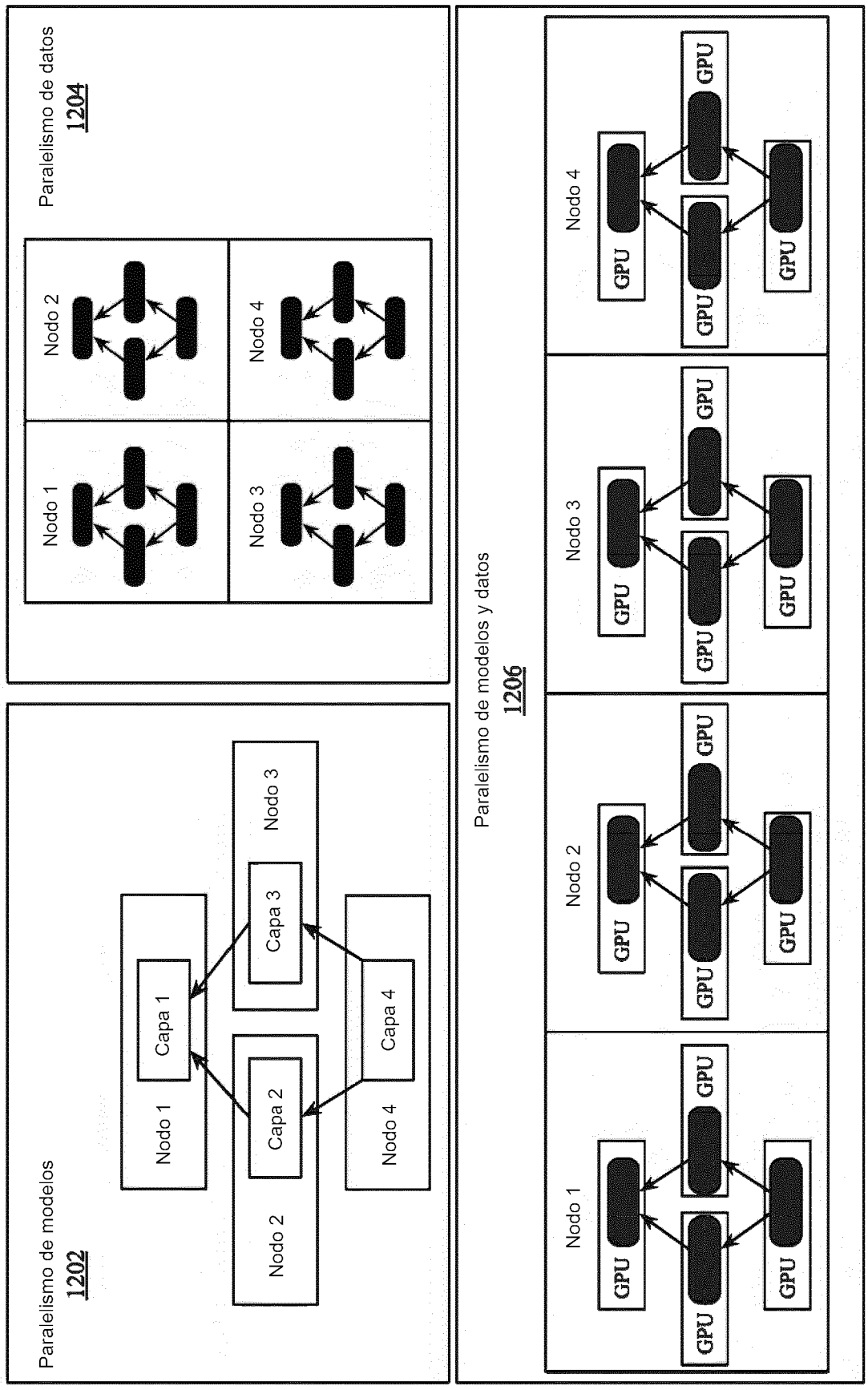

Paralelismo de modelos
1202
Nodo 1
Capa 1
Nodo 2
Capa 2
Nodo 3
Capa 3
Nodo 4
Capa 4
Paralelismo de datos
1204
Nodo 1
Nodo 2
Nodo 3
Nodo 4
Paralelismo de modelos y datos
1206
Nodo 1
Nodo 2
Nodo 3
Nodo 4
GPU

FIG. 12

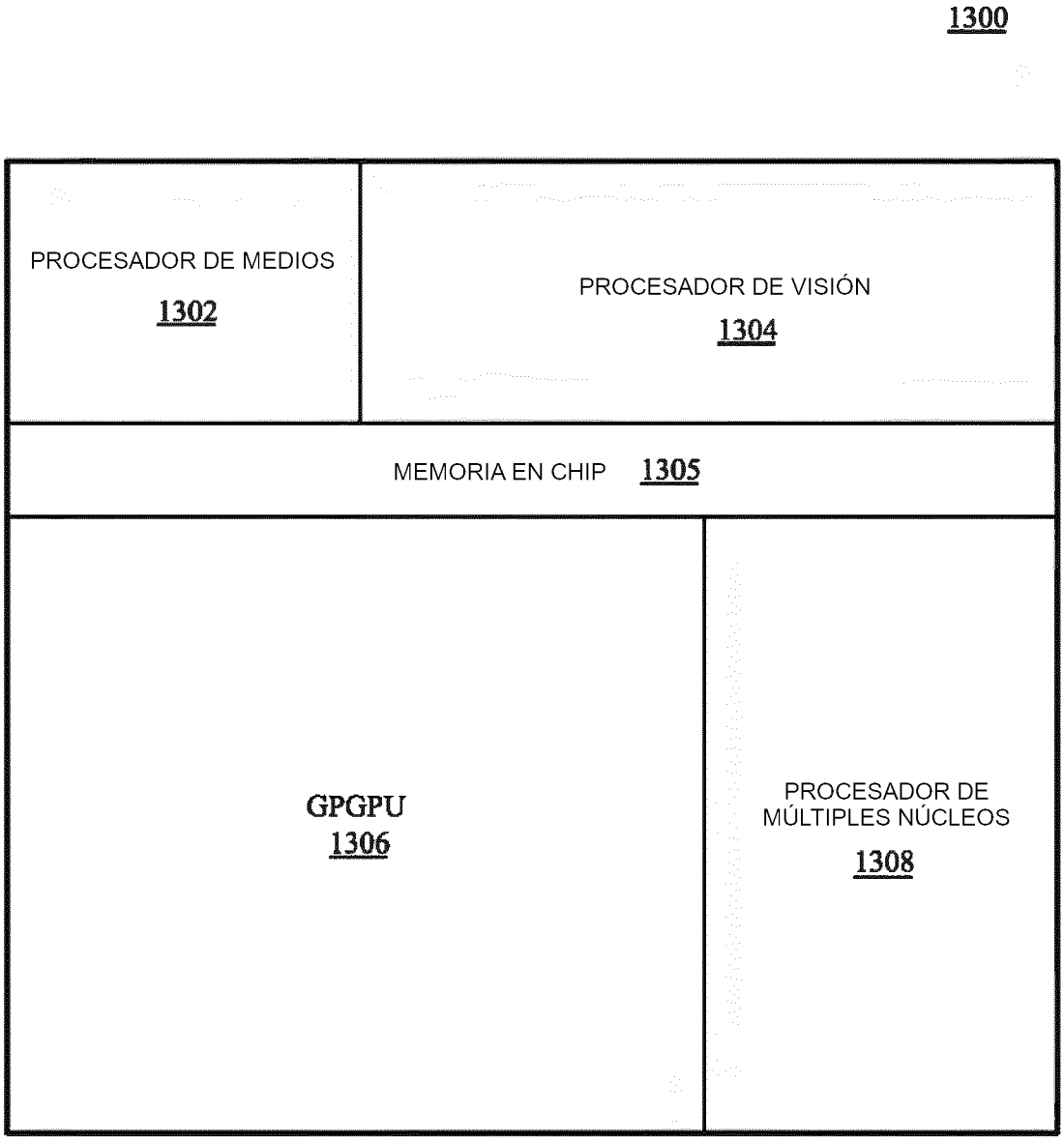

1300
PROCESADOR DE MEDIOS
1302
PROCESADOR DE VISIÓN
1304
MEMORIA EN CHIP 1305
GPGPU
1306
PROCESADOR DE MÚLTIPLES NÚCLEOS
1308

FIG. 13

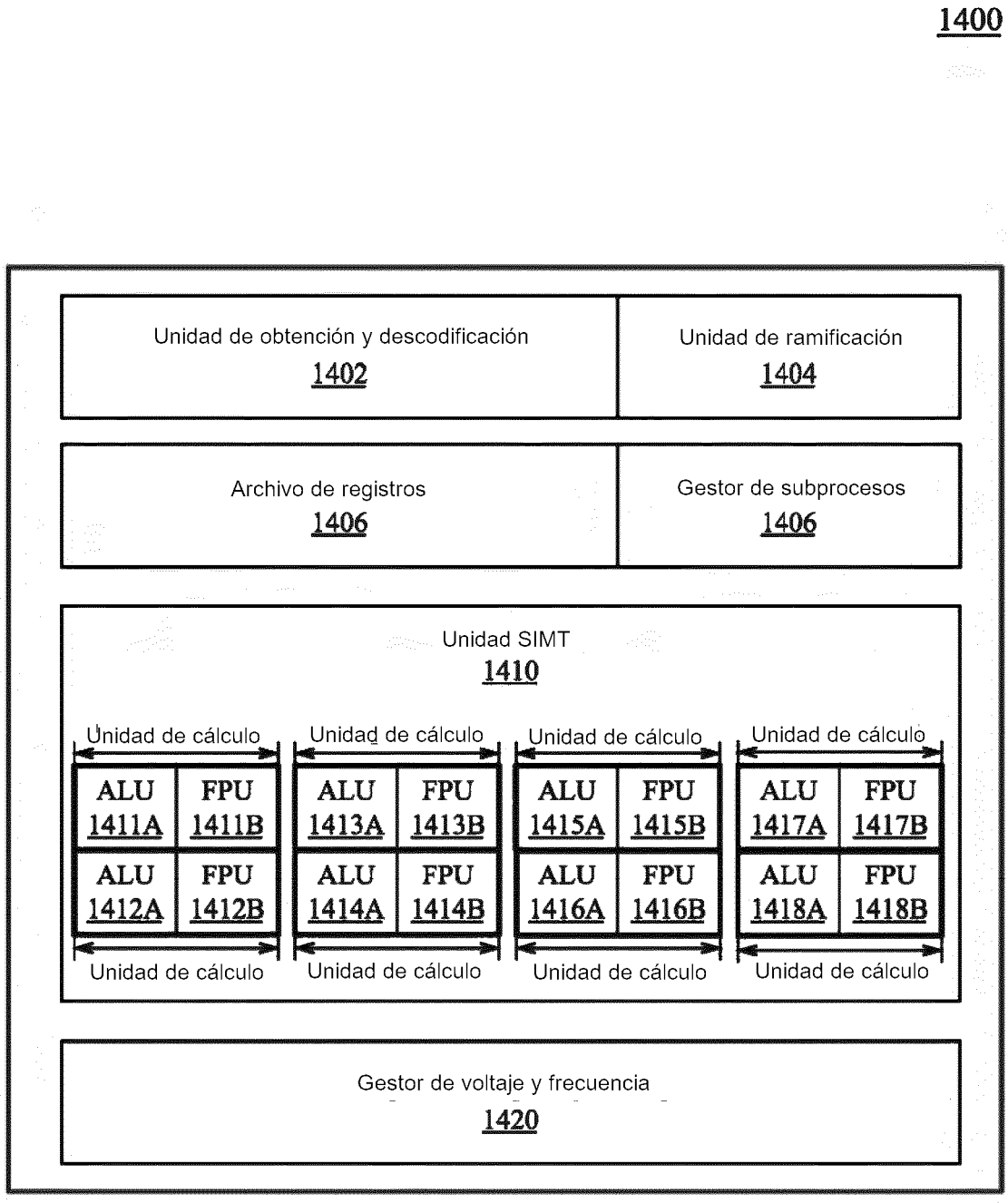

1400
Unidad de obtención y descodificación
1402
Unidad de ramificación
1404
Archivo de registros
1406
Gestor de subprocesos
1406
Unidad SIMT
1410
Unidad de cálculo
Unidad de cálculo
Unidad de cálculo
Unidad de cálculo
ALU 1411A
FPU 1411B
ALU 1412A
FPU 1412B
ALU 1413A
FPU 1413B
ALU 1414A
FPU 1414B
ALU 1415A
FPU 1415B
ALU 1416A
FPU 1416B
ALU 1417A
FPU 1417B
ALU 1418A
FPU 1418B
Unidad de cálculo
Unidad de cálculo
Unidad de cálculo
Unidad de cálculo
Gestor de voltaje y frecuencia
1420

FIG. 14

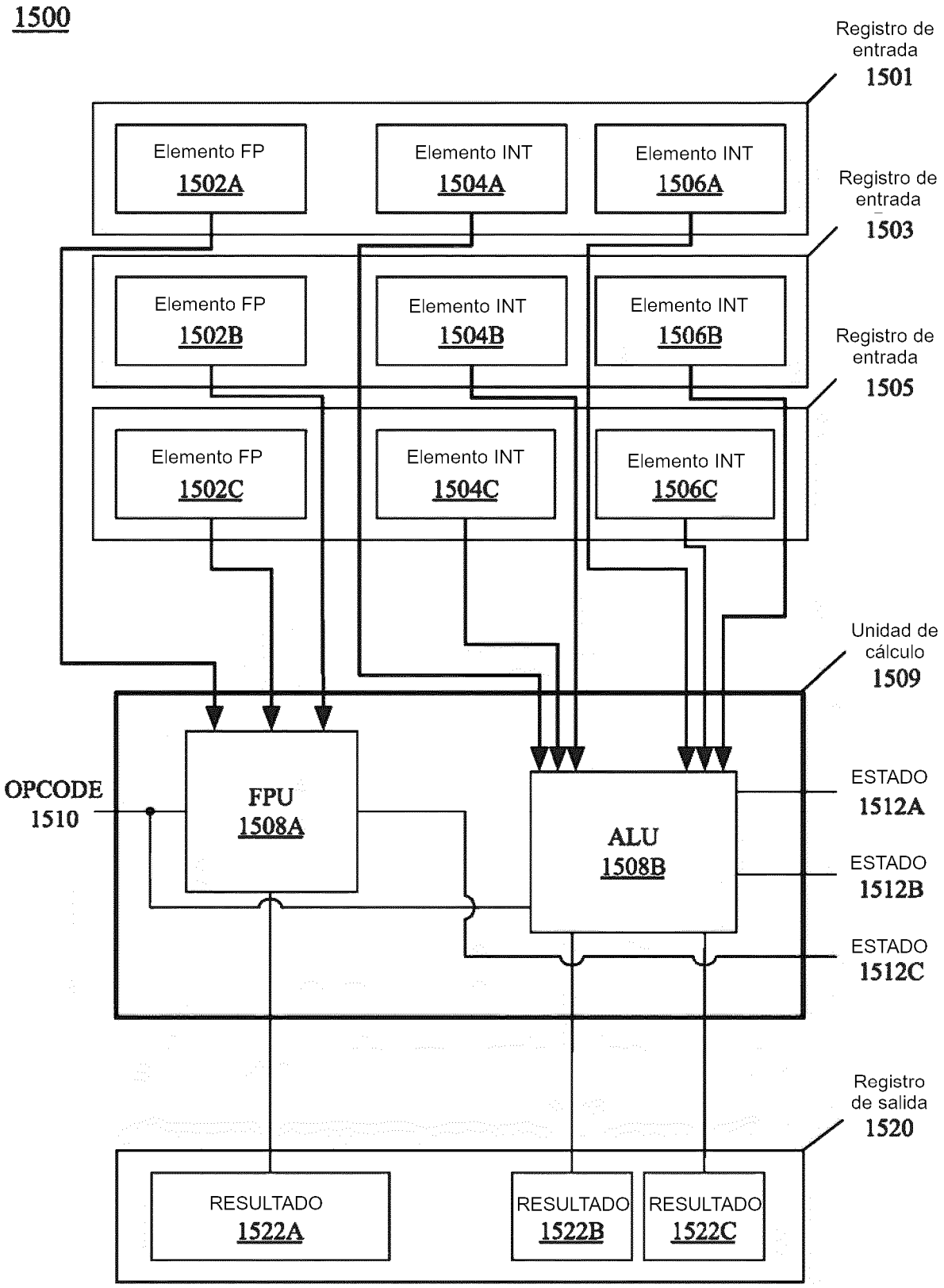

1500
Registro de entrada 1501
Elemento FP 1502A
Elemento INT 1504A
Elemento INT 1506A
Registro de entrada 1503
Elemento FP 1502B
Elemento INT 1504B
Elemento INT 1506B
Registro de entrada 1505
Elemento FP 1502C
Elemento INT 1504C
Elemento INT 1506C
Unidad de cálculo 1509
OPCODE 1510
FPU 1508A
ALU 1508B
ESTADO 1512A
ESTADO 1512B
ESTADO 1512C
Registro de salida 1520
RESULTADO 1522A
RESULTADO 1522B
RESULTADO 1522C

FIG. 15

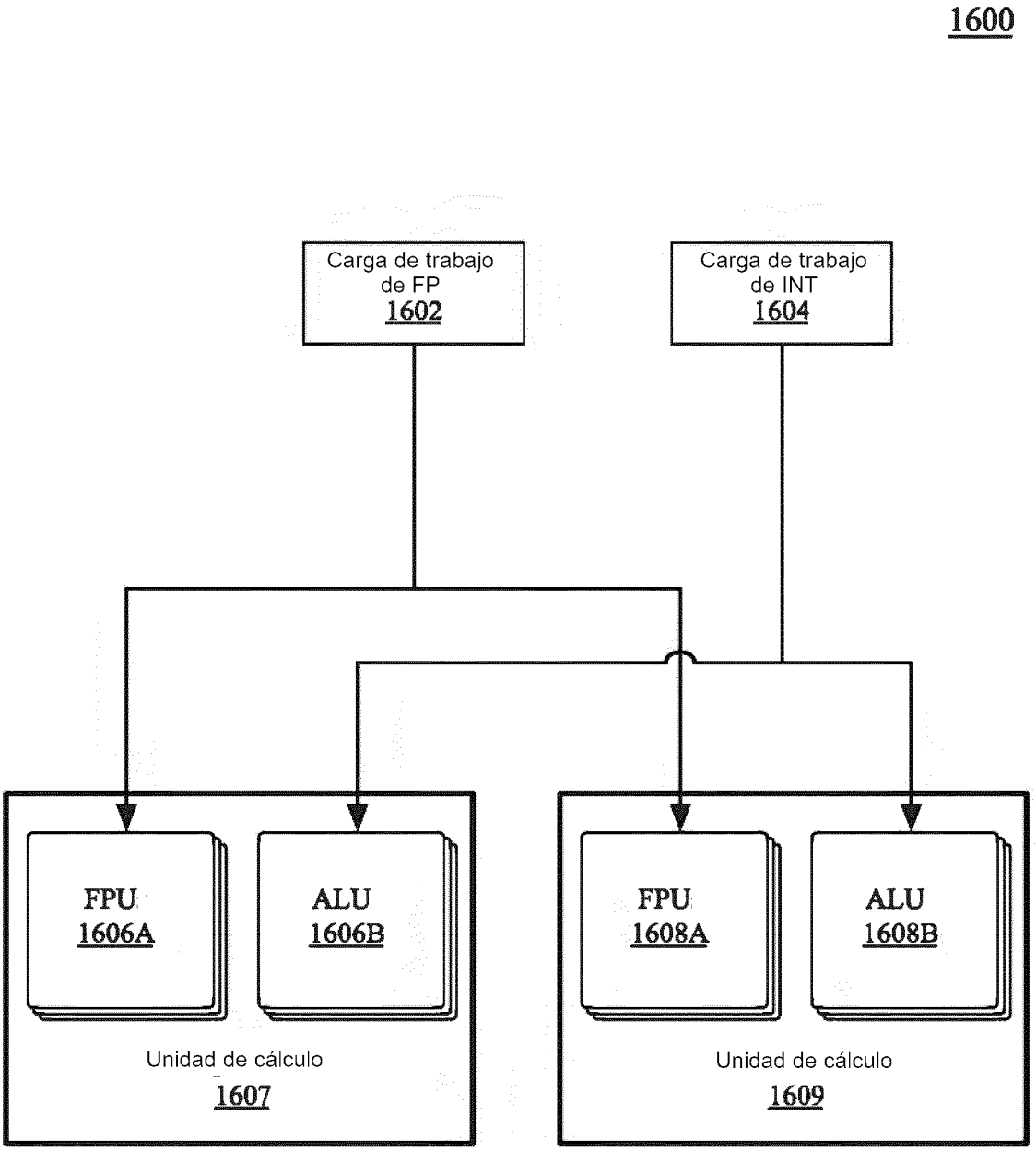

1600
Carga de trabajo de FP 1602
Carga de trabajo de INT 1604
FPU 1606A
ALU 1606B
Unidad de cálculo 1607
FPU 1608A
ALU 1608B
Unidad de cálculo 1609

FIG. 16

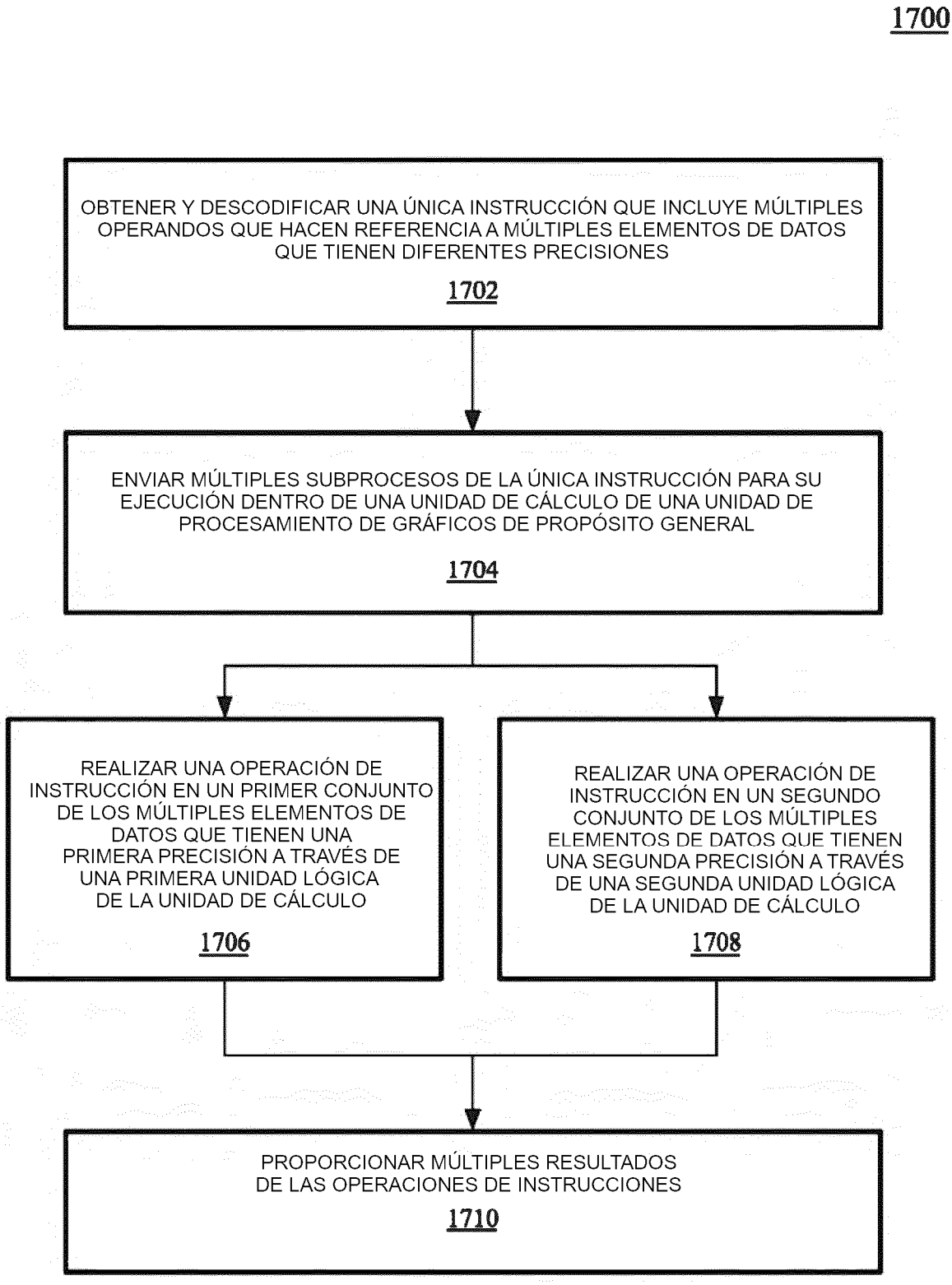

1700
OBTENER Y DESCODIFICAR UNA ÚNICA INSTRUCCIÓN QUE INCLUYE MÚLTIPLES OPERANDOS QUE HACEN REFERENCIA A MÚLTIPLES ELEMENTOS DE DATOS QUE TIENEN DIFERENTES PRECISIONES
1702
ENVIAR MÚLTIPLES SUBPROCESOS DE LA ÚNICA INSTRUCCIÓN PARA SU EJECUCIÓN DENTRO DE UNA UNIDAD DE CÁLCULO DE UNA UNIDAD DE PROCESAMIENTO DE GRÁFICOS DE PROPÓSITO GENERAL
1704
REALIZAR UNA OPERACIÓN DE INSTRUCCIÓN EN UN PRIMER CONJUNTO DE LOS MÚLTIPLES ELEMENTOS DE DATOS QUE TIENEN UNA PRIMERA PRECISIÓN A TRAVÉS DE UNA PRIMERA UNIDAD LÓGICA DE LA UNIDAD DE CÁLCULO
1706
REALIZAR UNA OPERACIÓN DE INSTRUCCIÓN EN UN SEGUNDO CONJUNTO DE LOS MÚLTIPLES ELEMENTOS DE DATOS QUE TIENEN UNA SEGUNDA PRECISIÓN A TRAVÉS DE UNA SEGUNDA UNIDAD LÓGICA DE LA UNIDAD DE CÁLCULO
1708
PROPORCIONAR MÚLTIPLES RESULTADOS DE LAS OPERACIONES DE INSTRUCCIONES
1710

FIG. 17

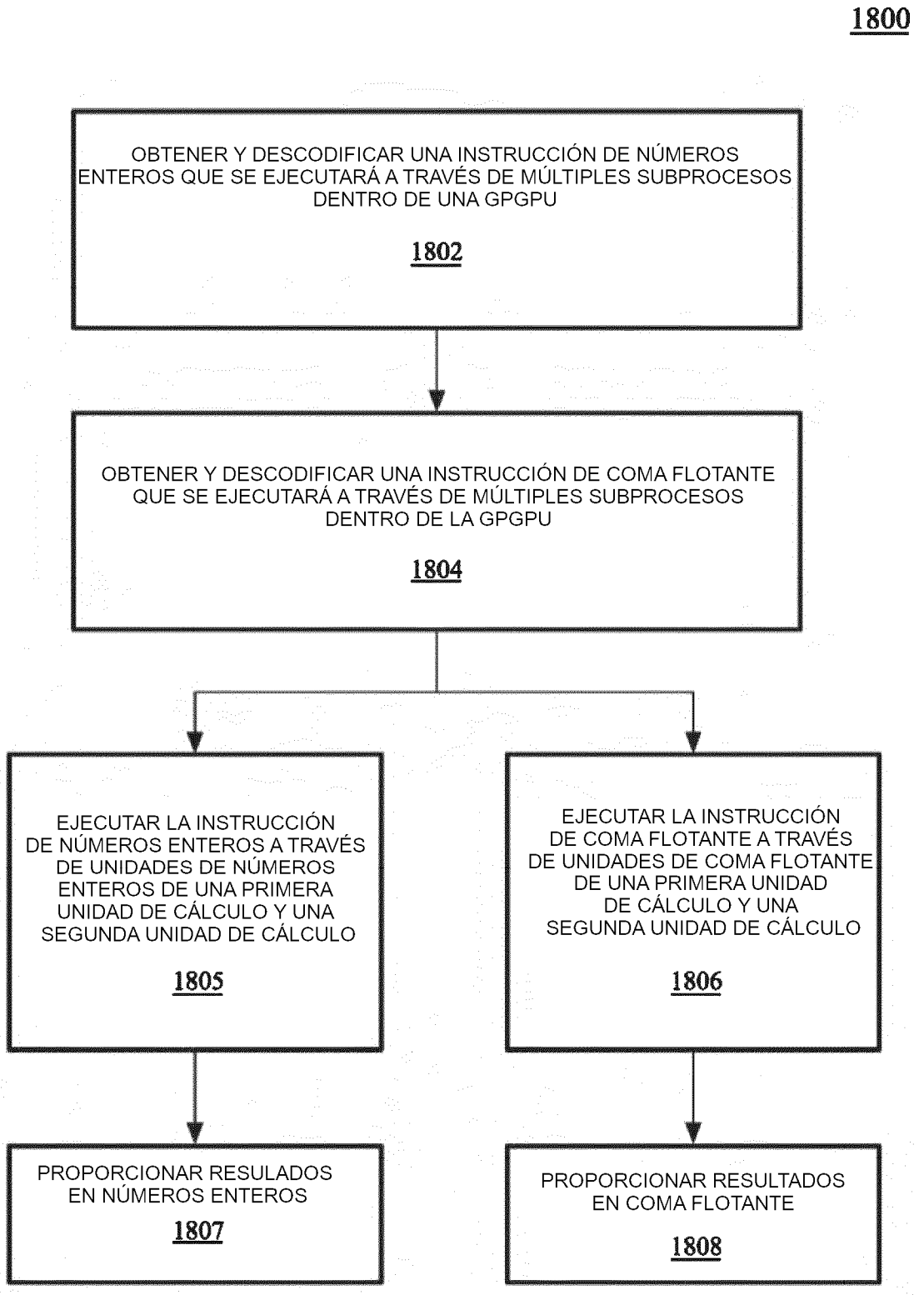

1800
OBTENER Y DESCODIFICAR UNA INSTRUCCIÓN DE NÚMEROS ENTEROS QUE SE EJECUTARÁ A TRAVÉS DE MÚLTIPLES SUBPROCESOS DENTRO DE UNA GPGPU
1802
OBTENER Y DESCODIFICAR UNA INSTRUCCIÓN DE COMA FLOTANTE QUE SE EJECUTARÁ A TRAVÉS DE MÚLTIPLES SUBPROCESOS DENTRO DE LA GPGPU
1804
EJECUTAR LA INSTRUCCIÓN DE NÚMEROS ENTEROS A TRAVÉS DE UNIDADES DE NÚMEROS ENTEROS DE UNA PRIMERA UNIDAD DE CÁLCULO Y UNA SEGUNDA UNIDAD DE CÁLCULO
1805
EJECUTAR LA INSTRUCCIÓN DE COMA FLOTANTE A TRAVÉS DE UNIDADES DE COMA FLOTANTE DE UNA PRIMERA UNIDAD DE CÁLCULO Y UNA SEGUNDA UNIDAD DE CÁLCULO
1806
PROPORCIONAR RESULADOS EN NÚMEROS ENTEROS
1807
PROPORCIONAR RESULTADOS EN COMA FLOTANTE
1808

FIG. 18

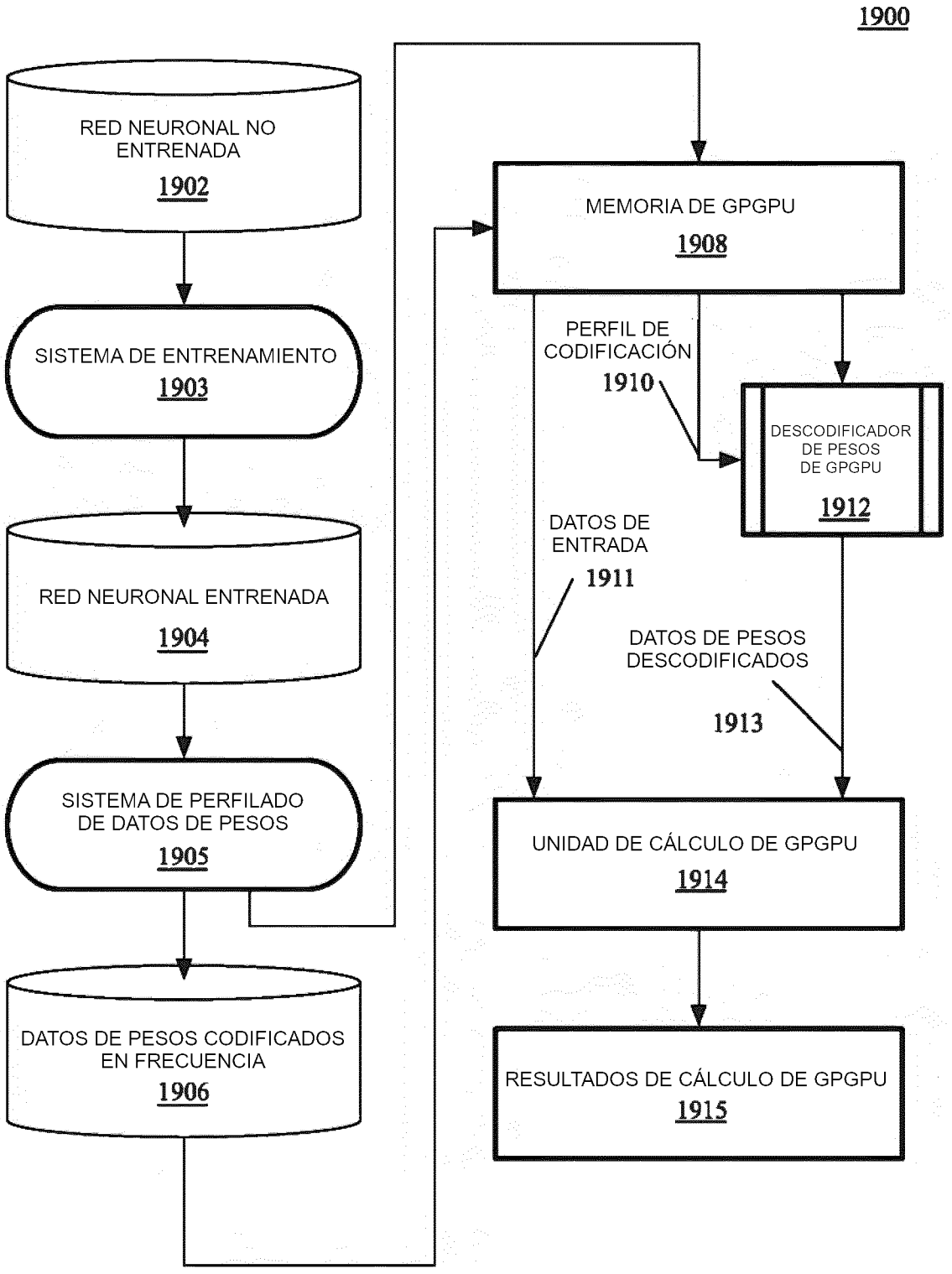

1900
RED NEURONAL NO ENTRENADA
1902
SISTEMA DE ENTRENAMIENTO
1903
RED NEURONAL ENTRENADA
1904
SISTEMA DE PERFILADO DE DATOS DE PESOS
1905
DATOS DE PESOS CODIFICADOS EN FRECUENCIA
1906
MEMORIA DE GPGPU
1908
PERFIL DE CODIFICACIÓN
1910
DESCODIFICADOR DE PESOS DE GPGPU
1912
DATOS DE ENTRADA
1911
DATOS DE PESOS DESCODIFICADOS
1913
UNIDAD DE CÁLCULO DE GPGPU
1914
RESULTADOS DE CÁLCULO DE GPGPU
1915

FIG. 19

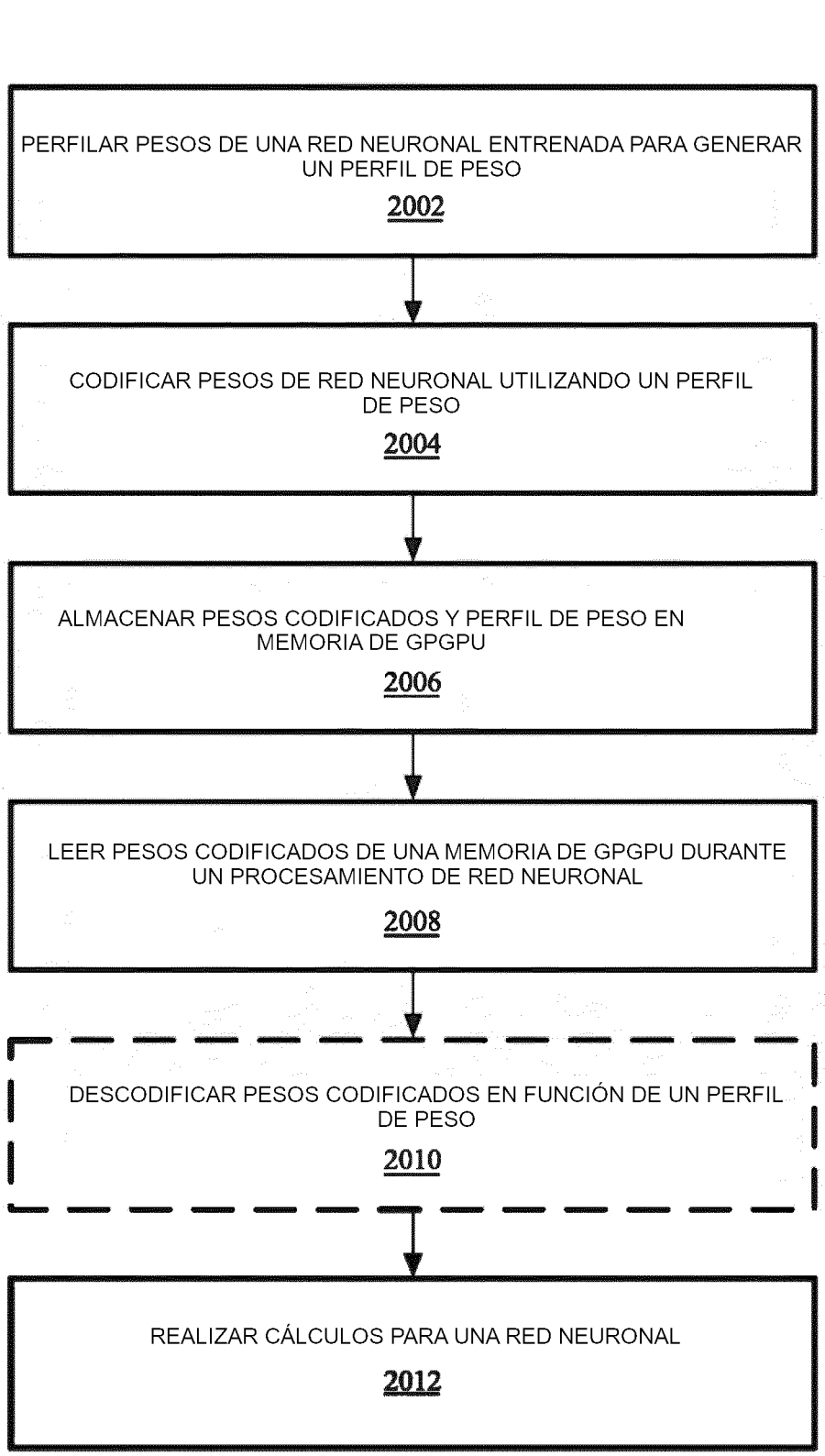

2000
PERFILAR PESOS DE UNA RED NEURONAL ENTRENADA PARA GENERAR UN PERFIL DE PESO
2002
CODIFICAR PESOS DE RED NEURONAL UTILIZANDO UN PERFIL DE PESO
2004
ALMACENAR PESOS CODIFICADOS Y PERFIL DE PESO EN MEMORIA DE GPGPU
2006
LEER PESOS CODIFICADOS DE UNA MEMORIA DE GPGPU DURANTE UN PROCESAMIENTO DE RED NEURONAL
2008
DESCODIFICAR PESOS CODIFICADOS EN FUNCIÓN DE UN PERFIL DE PESO
2010
REALIZAR CÁLCULOS PARA UNA RED NEURONAL
2012

FIG. 20

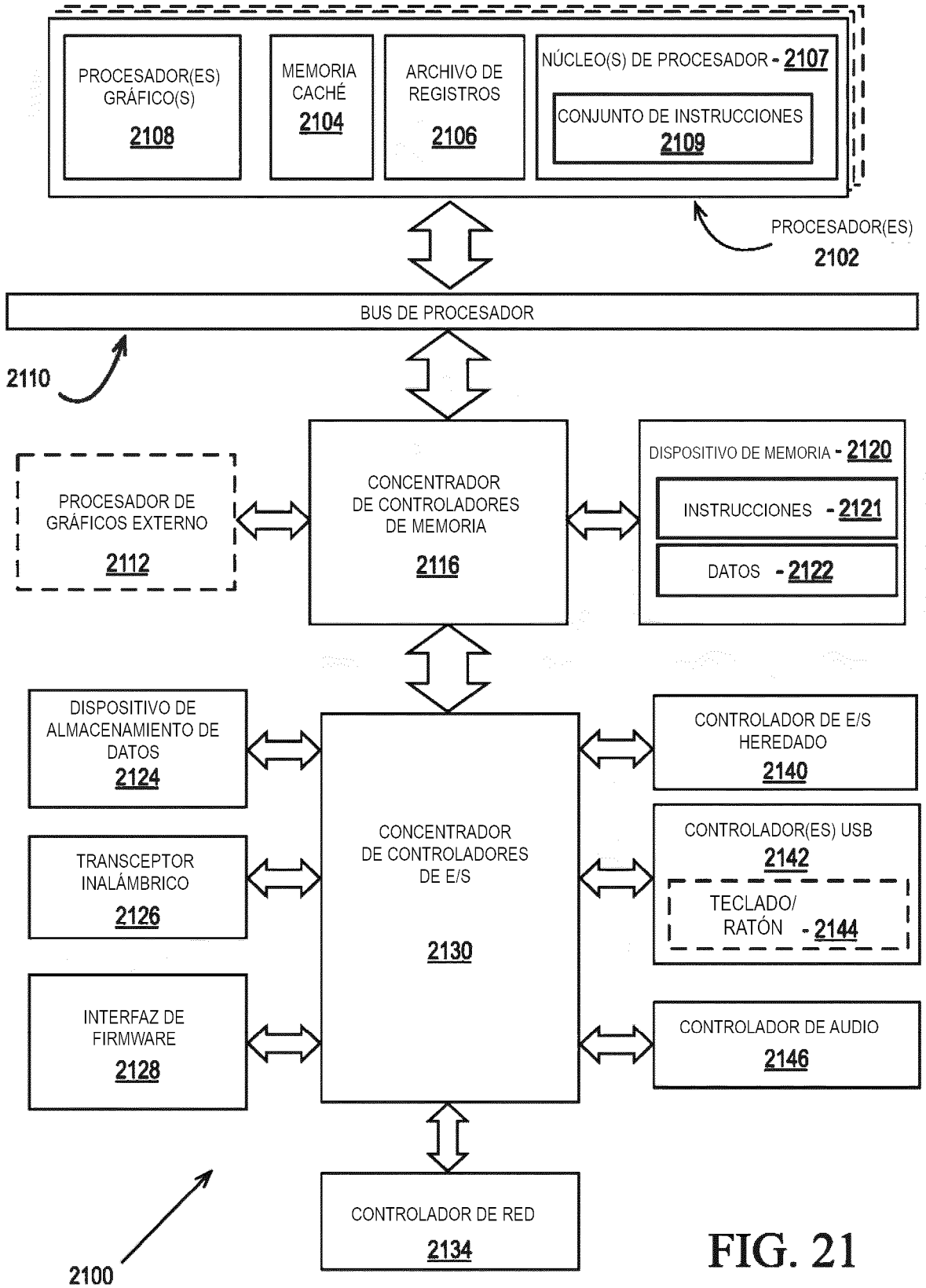

PROCESADOR(ES) GRÁFICO(S) 2108
MEMORIA CACHÉ 2104
ARCHIVO DE REGISTROS 2106
NÚCLEO(S) DE PROCESADOR - 2107
CONJUNTO DE INSTRUCCIONES 2109
PROCESADOR(ES) 2102
BUS DE PROCESADOR
2110
PROCESADOR DE GRÁFICOS EXTERNO 2112
CONCENTRADOR DE CONTROLADORES DE MEMORIA 2116
DISPOSITIVO DE MEMORIA - 2120
INSTRUCCIONES - 2121
DATOS - 2122
DISPOSITIVO DE ALMACENAMIENTO DE DATOS 2124
TRANSCEPTOR INALÁMBRICO 2126
INTERFAZ DE FIRMWARE 2128
CONCENTRADOR DE CONTROLADORES DE E/S 2130
CONTROLADOR DE E/S HEREDADO 2140
CONTROLADOR(ES) USB 2142
TECLADO/ RATÓN - 2144
CONTROLADOR DE AUDIO 2146
CONTROLADOR DE RED 2134
2100

FIG. 21

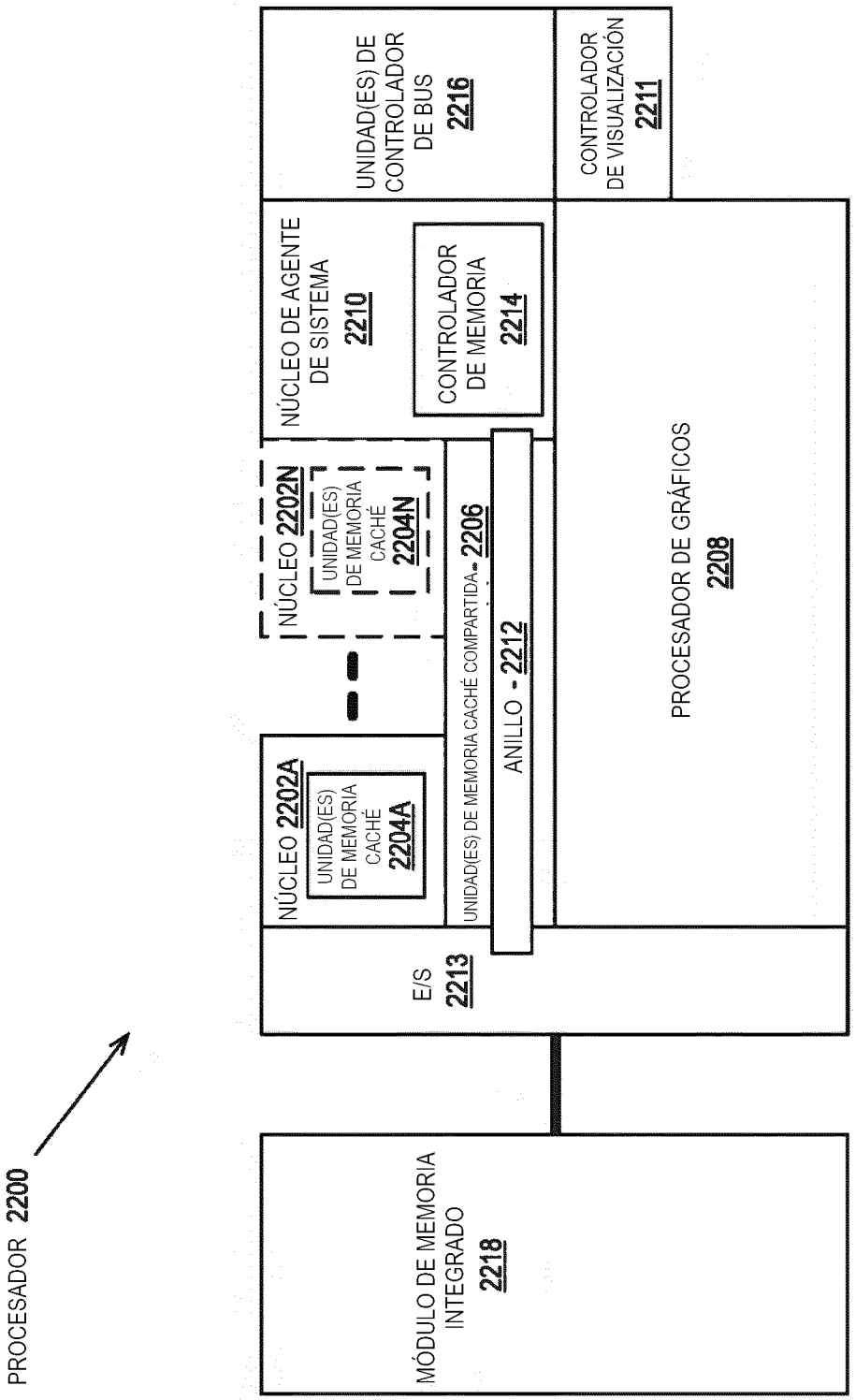

PROCESADOR 2200
MÓDULO DE MEMORIA INTEGRADO 2218
E/S 2213
NÚCLEO 2202A
UNIDAD(ES) DE MEMORIA CACHÉ 2204A
NÚCLEO 2202N
UNIDAD(ES) DE MEMORIA CACHÉ 2204N
UNIDAD(ES) DE MEMORIA CACHÉ COMPARTIDA - 2206
ANILLO - 2212
NÚCLEO DE AGENTE DE SISTEMA 2210
CONTROLADOR DE MEMORIA 2214
UNIDAD(ES) DE CONTROLADOR DE BUS 2216
CONTROLADOR DE VISUALIZACIÓN 2211
PROCESADOR DE GRÁFICOS 2208

FIG. 22

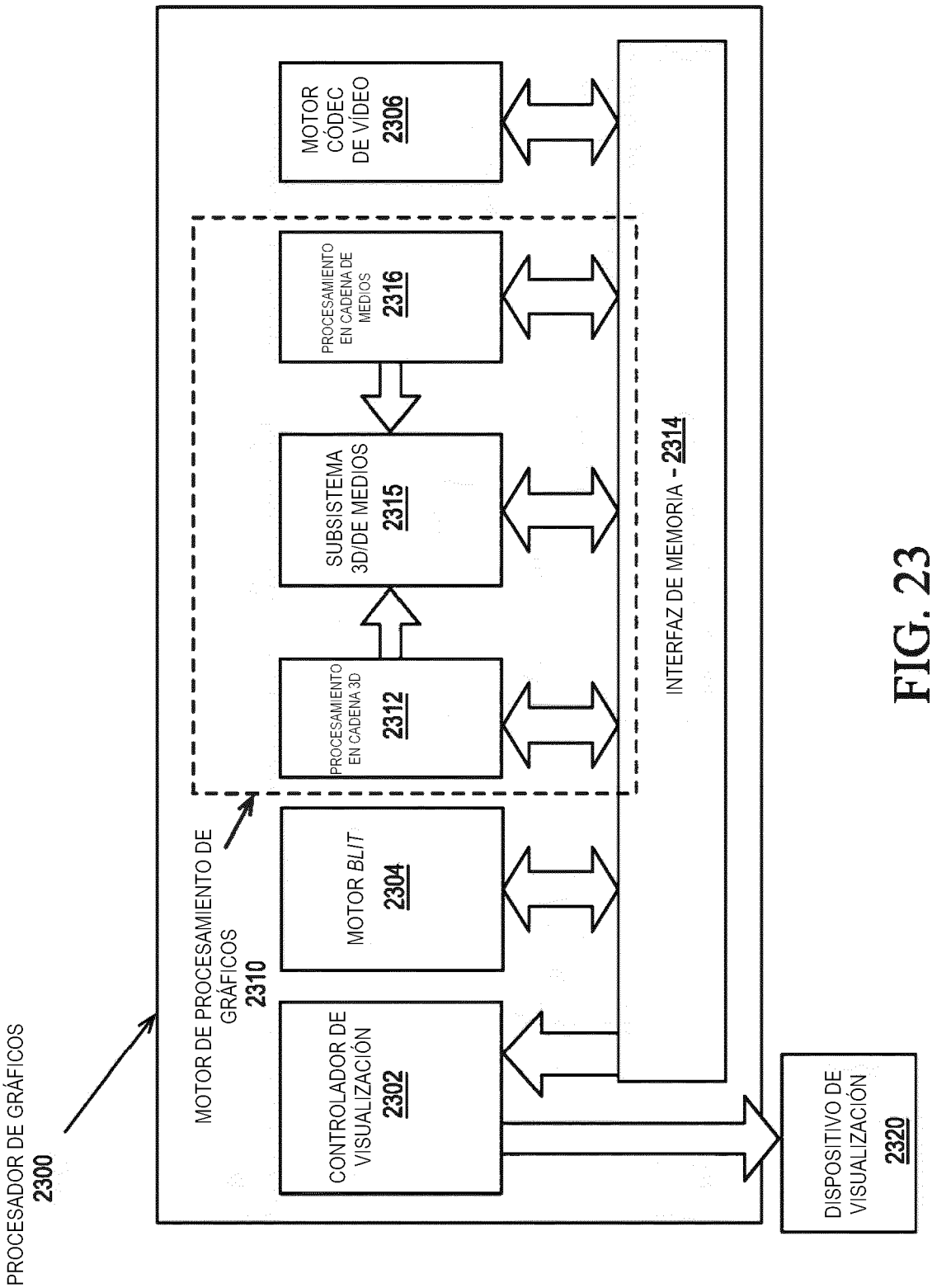

PROCESADOR DE GRÁFICOS
2300
MOTOR DE PROCESAMIENTO DE GRÁFICOS
2310
CONTROLADOR DE VISUALIZACIÓN
2302
MOTOR BLIT
2304
PROCESAMIENTO EN CADENA 3D
2312
SUBSISTEMA 3D/DE MEDIOS
2315
PROCESAMIENTO EN CADENA DE MEDIOS
2316
MOTOR CÓDEC DE VÍDEO
2306
INTERFAZ DE MEMORIA - 2314
DISPOSITIVO DE VISUALIZACIÓN
2320

FIG. 23

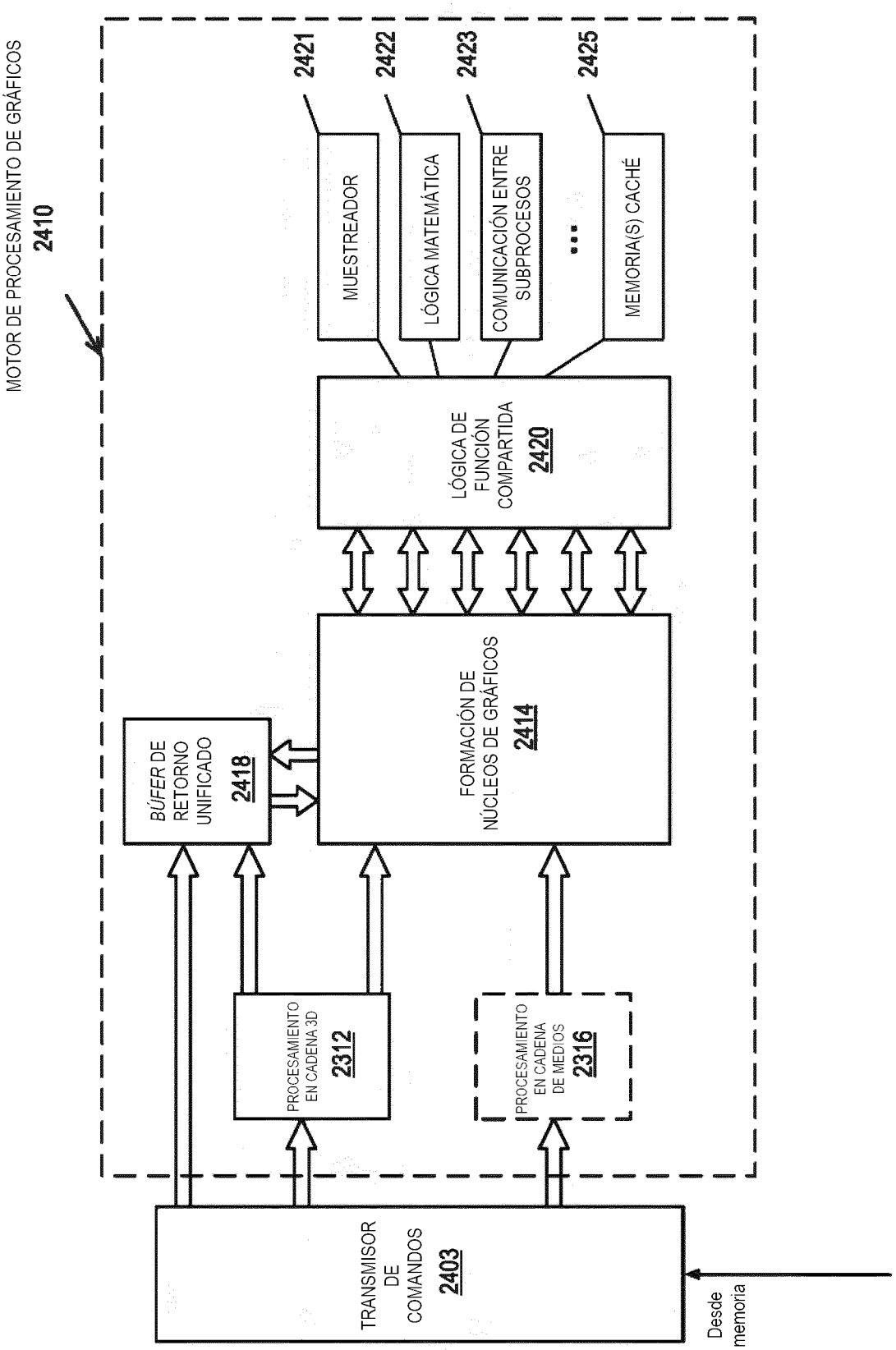

MOTOR DE PROCESAMIENTO DE GRÁFICOS
2410
2421
2422
2423
2425
MUESTREADOR
LÓGICA MATEMÁTICA
COMUNICACIÓN ENTRE SUBPROCESOS
...
MEMORIA(S) CACHÉ
LÓGICA DE FUNCIÓN COMPARTIDA
2420
FORMACIÓN DE NÚCLEOS DE GRÁFICOS
2414
BÚFER DE RETORNO UNIFICADO
2418
PROCESAMIENTO EN CADENA 3D
2312
PROCESAMIENTO EN CADENA DE MEDIOS
2316
TRANSMISOR DE COMANDOS
2403
Desde memoria

FIG. 24

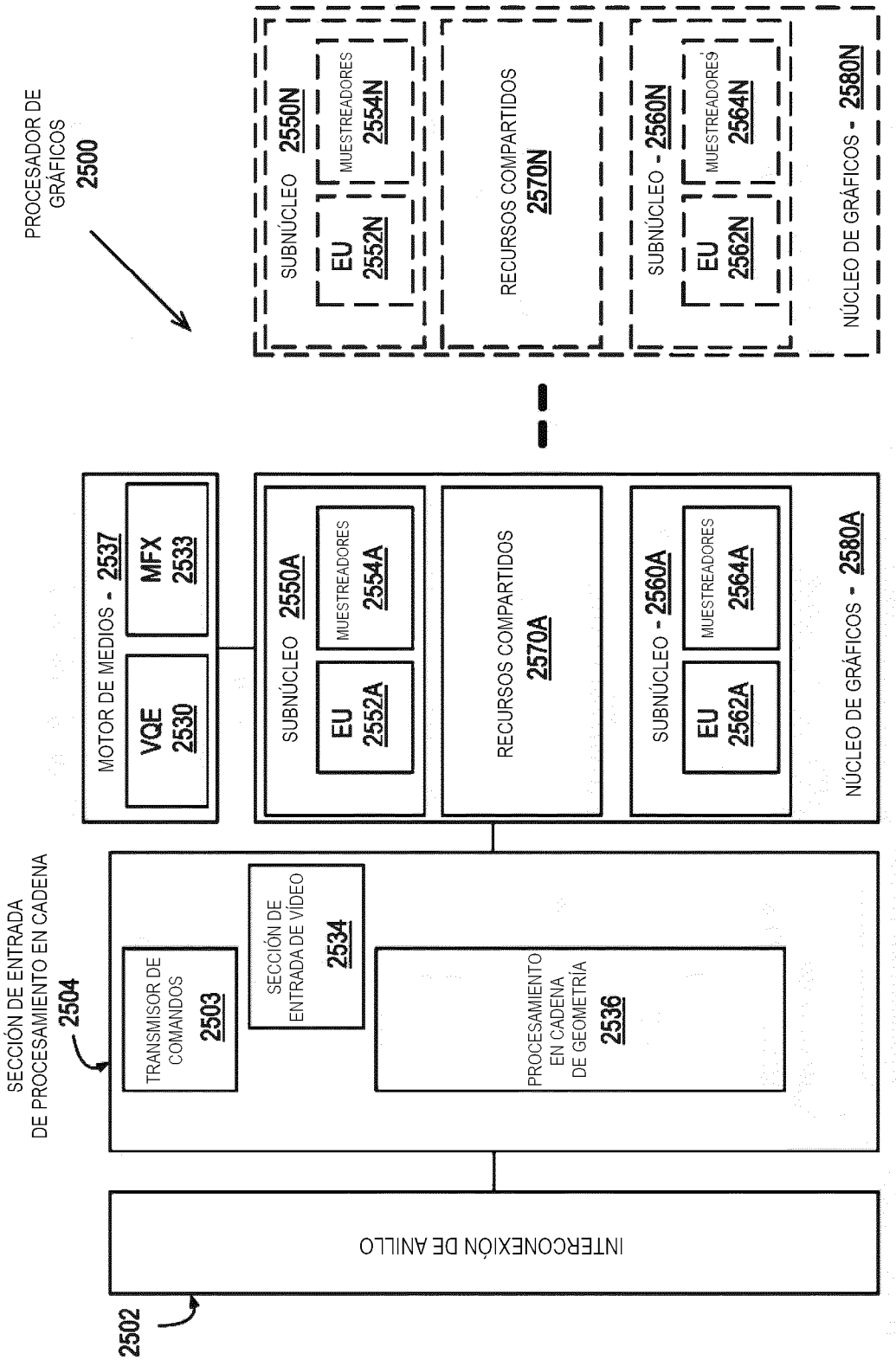

PROCESADOR DE GRÁFICOS 2500
INTERCONEXIÓN DE ANILLO
2502
SECCIÓN DE ENTRADA DE PROCESAMIENTO EN CADENA 2504
TRANSMISOR DE COMANDOS 2503
SECCIÓN DE ENTRADA DE VÍDEO 2534
PROCESAMIENTO EN CADENA DE GEOMETRÍA 2536
MOTOR DE MEDIOS - 2537
VQE 2530
MFX 2533
SUBNÚCLEO 2550A
EU 2552A
MUESTREADORES 2554A
RECURSOS COMPARTIDOS 2570A
SUBNÚCLEO - 2560A
EU 2562A
MUESTREADORES 2564A
NÚCLEO DE GRÁFICOS - 2580A
SUBNÚCLEO 2550N
EU 2552N
MUESTREADORES 2554N
RECURSOS COMPARTIDOS 2570N
SUBNÚCLEO - 2560N
EU 2562N
MUESTREADORES 2564N
NÚCLEO DE GRÁFICOS - 2580N

FIG. 25

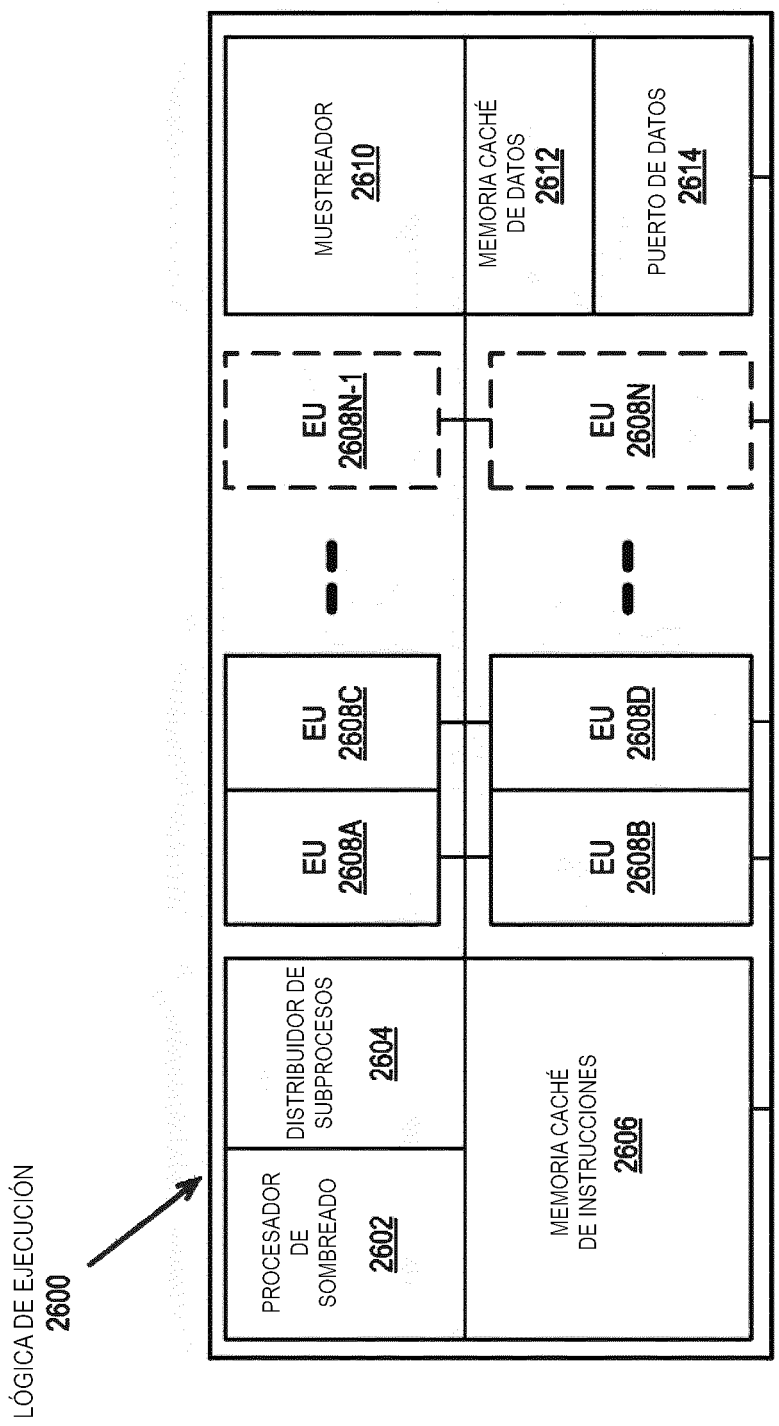

LÓGICA DE EJECUCIÓN
2600
PROCESADOR DE SOMBREADO
2602
DISTRIBUIDOR DE SUBPROCESOS
2604
MEMORIA CACHÉ DE INSTRUCCIONES
2606
EU 2608A
EU 2608B
EU 2608C
EU 2608D
EU 2608N-1
EU 2608N
MUESTREADOR
2610
MEMORIA CACHÉ DE DATOS
2612
PUERTO DE DATOS
2614

FIG. 26

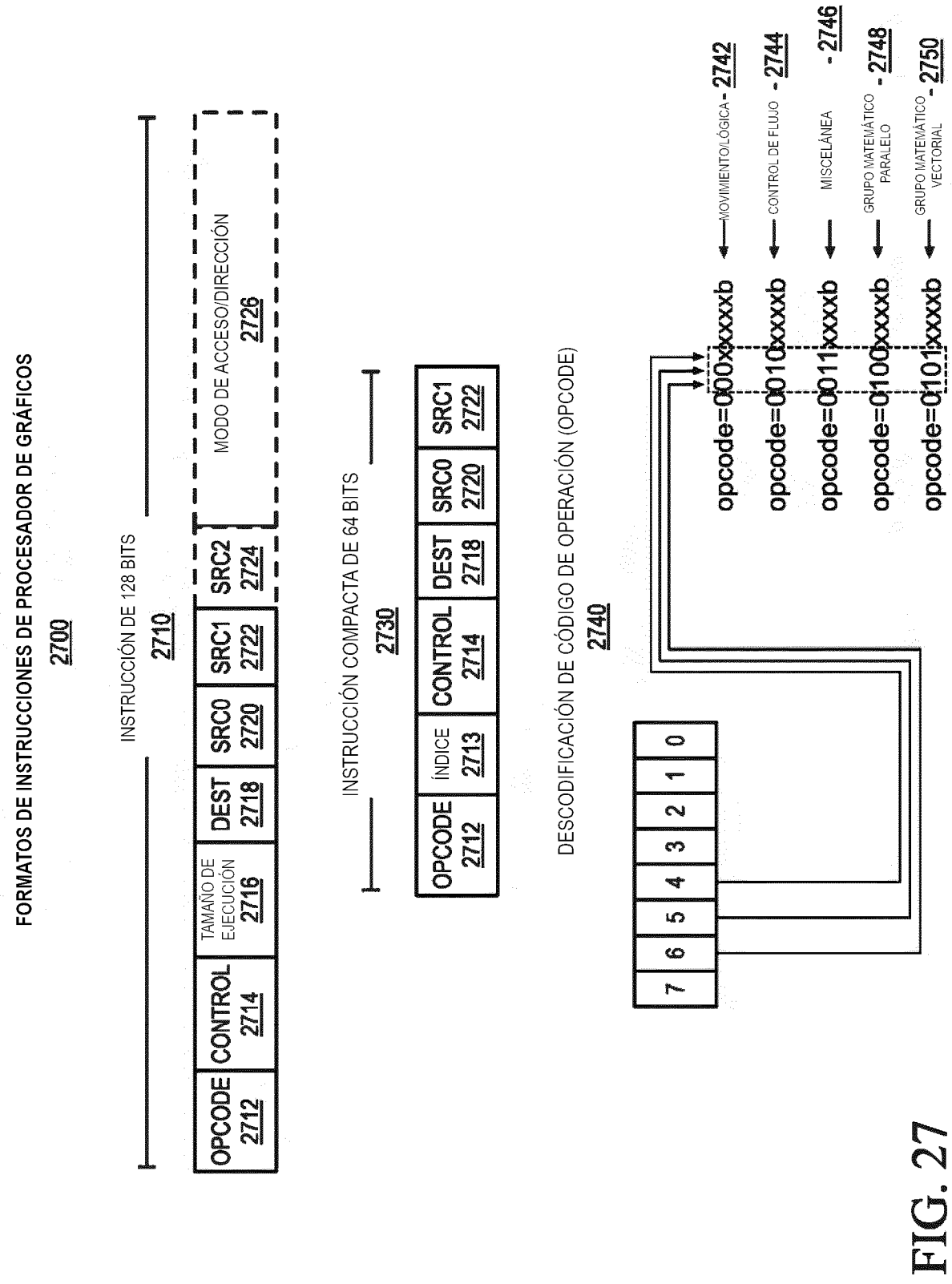

FORMATOS DE INSTRUCCIONES DE PROCESADOR DE GRÁFICOS
2700
INSTRUCCIÓN DE 128 BITS
2710
OPCODE 2712
CONTROL 2714
TAMAÑO DE EJECUCIÓN 2716
DEST 2718
SRC0 2720
SRC1 2722
SRC2 2724
MODO DE ACCESO/DIRECCIÓN 2726
INSTRUCCIÓN COMPACTA DE 64 BITS
2730
OPCODE 2712
ÍNDICE 2713
CONTROL 2714
DEST 2718
SRC0 2720
SRC1 2722
DESCODIFICACIÓN DE CÓDIGO DE OPERACIÓN (OPCODE)
2740
7 6 5 4 3 2 1 0
opcode=000xxxxb
opcode=001xxxxb
opcode=011xxxxb
opcode=100xxxxb
opcode=101xxxxb
MOVIMIENTO/LÓGICA - 2742
CONTROL DE FLUJO - 2744
MISCELÁNEA - 2746
GRUPO MATEMÁTICO PARALELO - 2748
GRUPO MATEMÁTICO VECTORIAL - 2750

FIG. 27

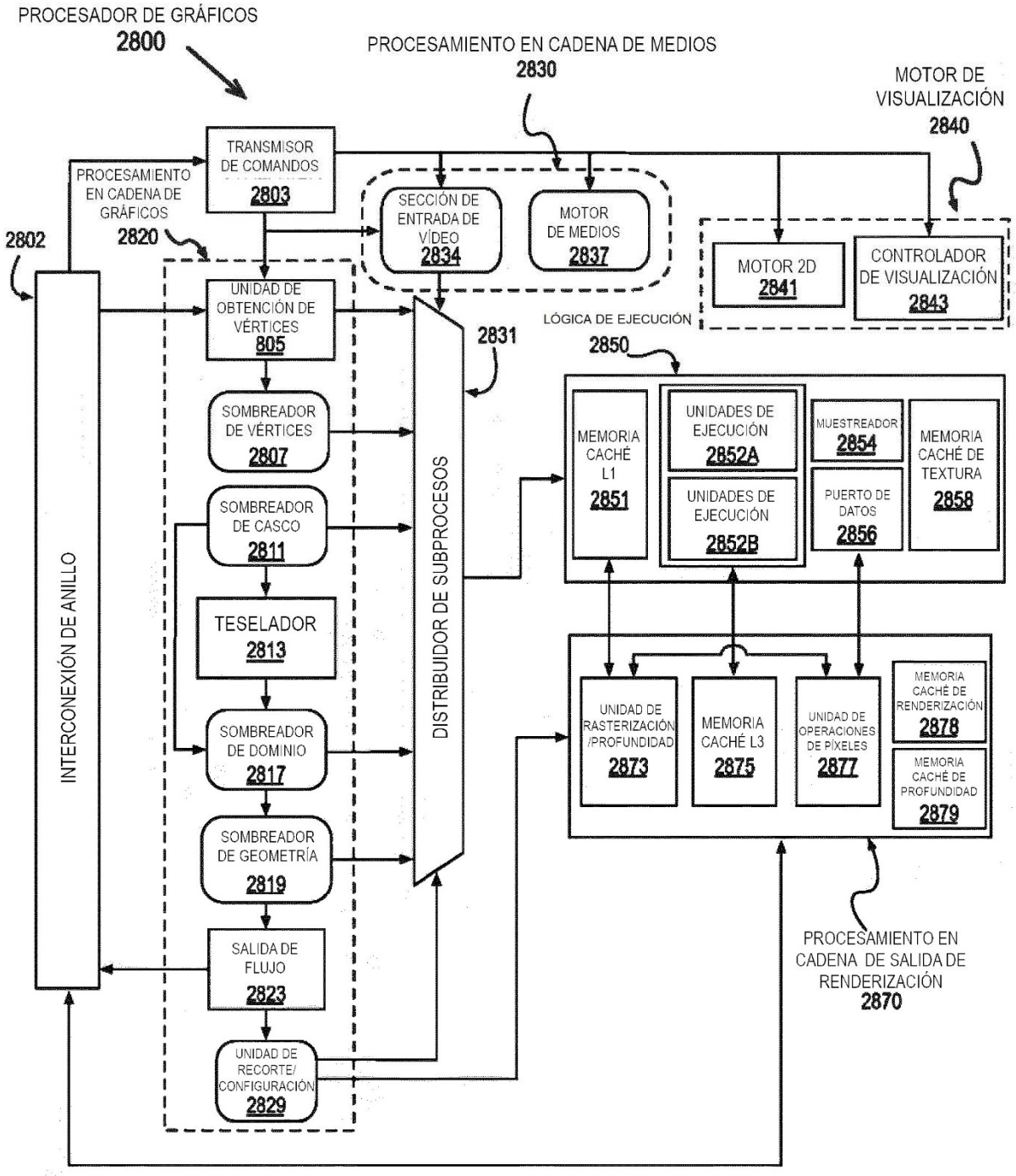

PROCESADOR DE GRÁFICOS
2800
PROCESAMIENTO EN CADENA DE MEDIOS
2830
MOTOR DE VISUALIZACIÓN
2840
TRANSMISOR DE COMANDOS
2803
PROCESAMIENTO EN CADENA DE GRÁFICOS
2820
2802
SECCIÓN DE ENTRADA DE VÍDEO
2834
MOTOR DE MEDIOS
2837
MOTOR 2D
2841
CONTROLADOR DE VISUALIZACIÓN
2843
UNIDAD DE OBTENCIÓN DE VÉRTICES
805
2831
LÓGICA DE EJECUCIÓN
2850
SOMBREADOR DE VÉRTICES
2807
SOMBREADOR DE CASCO
2811
TESELADOR
2813
SOMBREADOR DE DOMINIO
2817
SOMBREADOR DE GEOMETRÍA
2819
SALIDA DE FLUJO
2823
UNIDAD DE RECORTE/ CONFIGURACIÓN
2829
INTERCONEXIÓN DE ANILLO
DISTRIBUIDOR DE SUBPROCESOS
MEMORIA CACHÉ L1
2851
UNIDADES DE EJECUCIÓN
2852A
UNIDADES DE EJECUCIÓN
2852B
MUESTREADOR
2854
PUERTO DE DATOS
2856
MEMORIA CACHÉ DE TEXTURA
2858
UNIDAD DE RASTERIZACIÓN /PROFUNDIDAD
2873
MEMORIA CACHÉ L3
2875
UNIDAD DE OPERACIONES DE PÍXELES
2877
MEMORIA CACHÉ DE RENDERIZACIÓN
2878
MEMORIA CACHÉ DE PROFUNDIDAD
2879
PROCESAMIENTO EN CADENA DE SALIDA DE RENDERIZACIÓN
2870

FIG. 28

FIG. 29A FORMATO DE COMANDO DE PROCESADOR DE GRÁFICOS

2900

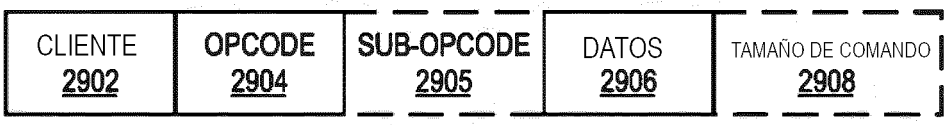
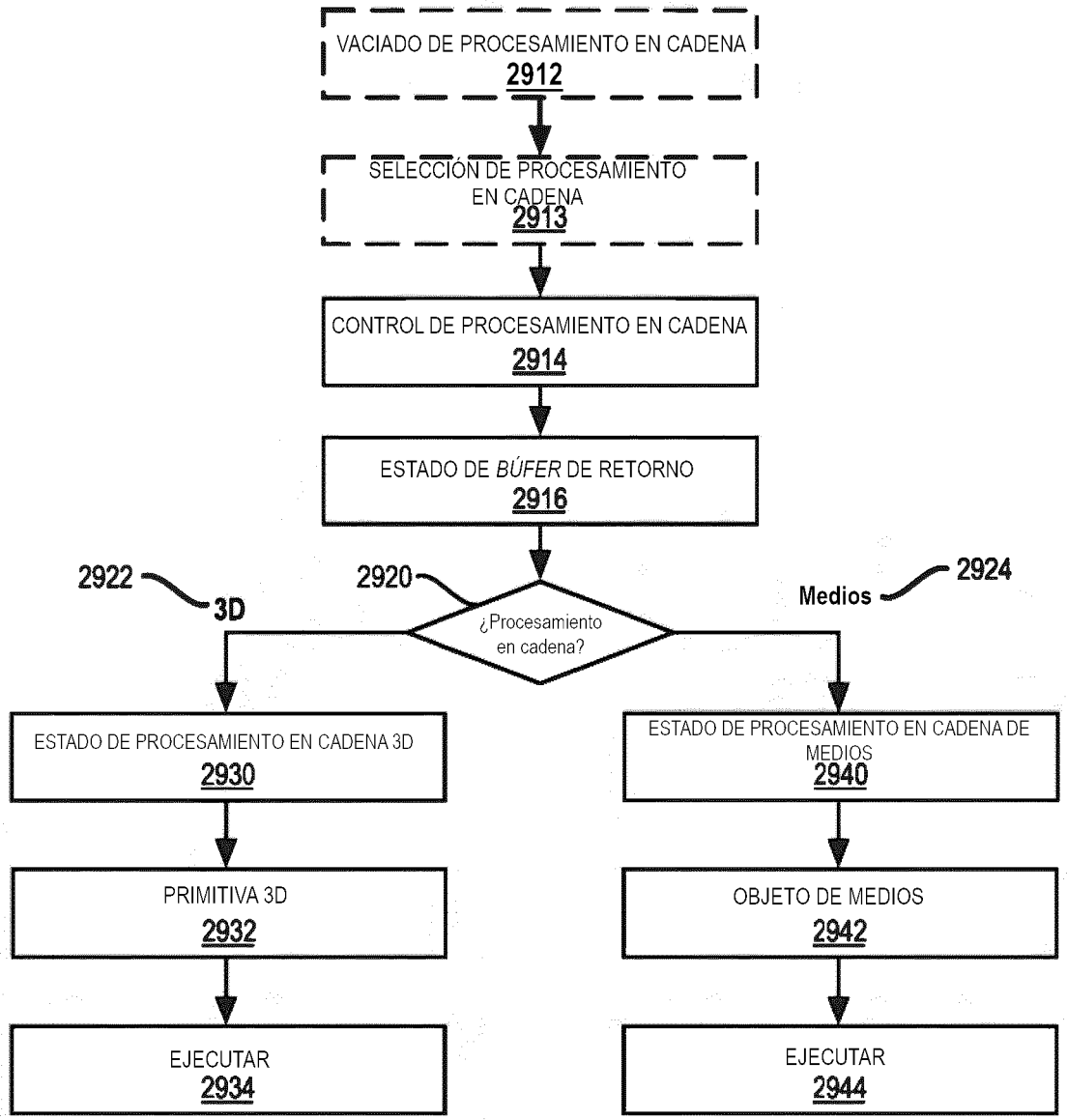


FIG. 29B SECUENCIA DE COMANDOS DE PROCESADOR DE GRÁFICOS

2910



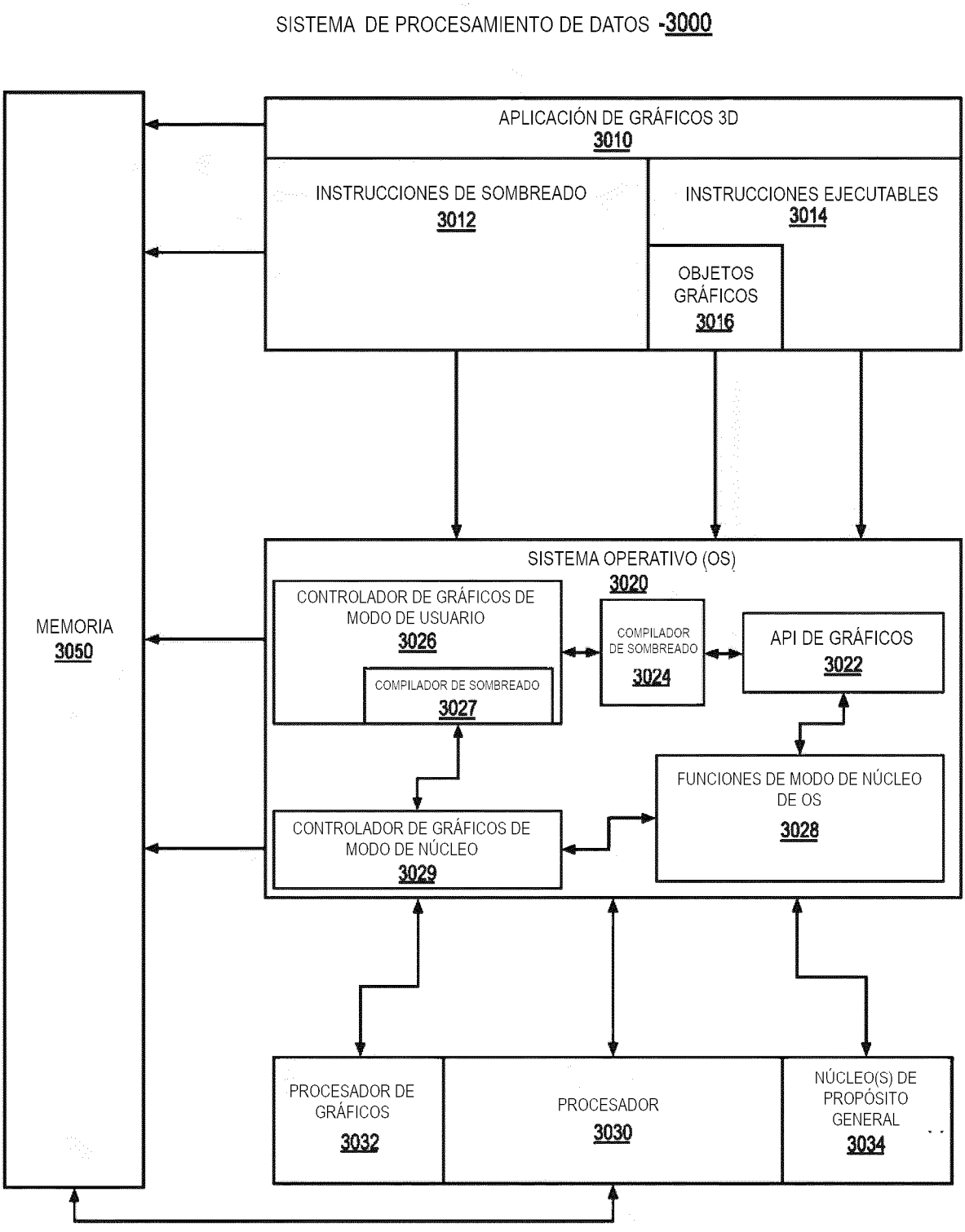

SISTEMA DE PROCESAMIENTO DE DATOS -3000
APLICACIÓN DE GRÁFICOS 3D
3010
INSTRUCCIONES DE SOMBREADO
3012
INSTRUCCIONES EJECUTABLES
3014
OBJETOS GRÁFICOS
3016
MEMORIA
3050
SISTEMA OPERATIVO (OS)
3020
CONTROLADOR DE GRÁFICOS DE MODO DE USUARIO
3026
COMPILADOR DE SOMBREADO
3027
COMPILADOR DE SOMBREADO
3024
API DE GRÁFICOS
3022
FUNCIONES DE MODO DE NÚCLEO DE OS
3028
CONTROLADOR DE GRÁFICOS DE MODO DE NÚCLEO
3029
PROCESADOR DE GRÁFICOS
3032
PROCESADOR
3030
NÚCLEO(S) DE PROPÓSITO GENERAL
3034

FIG. 30

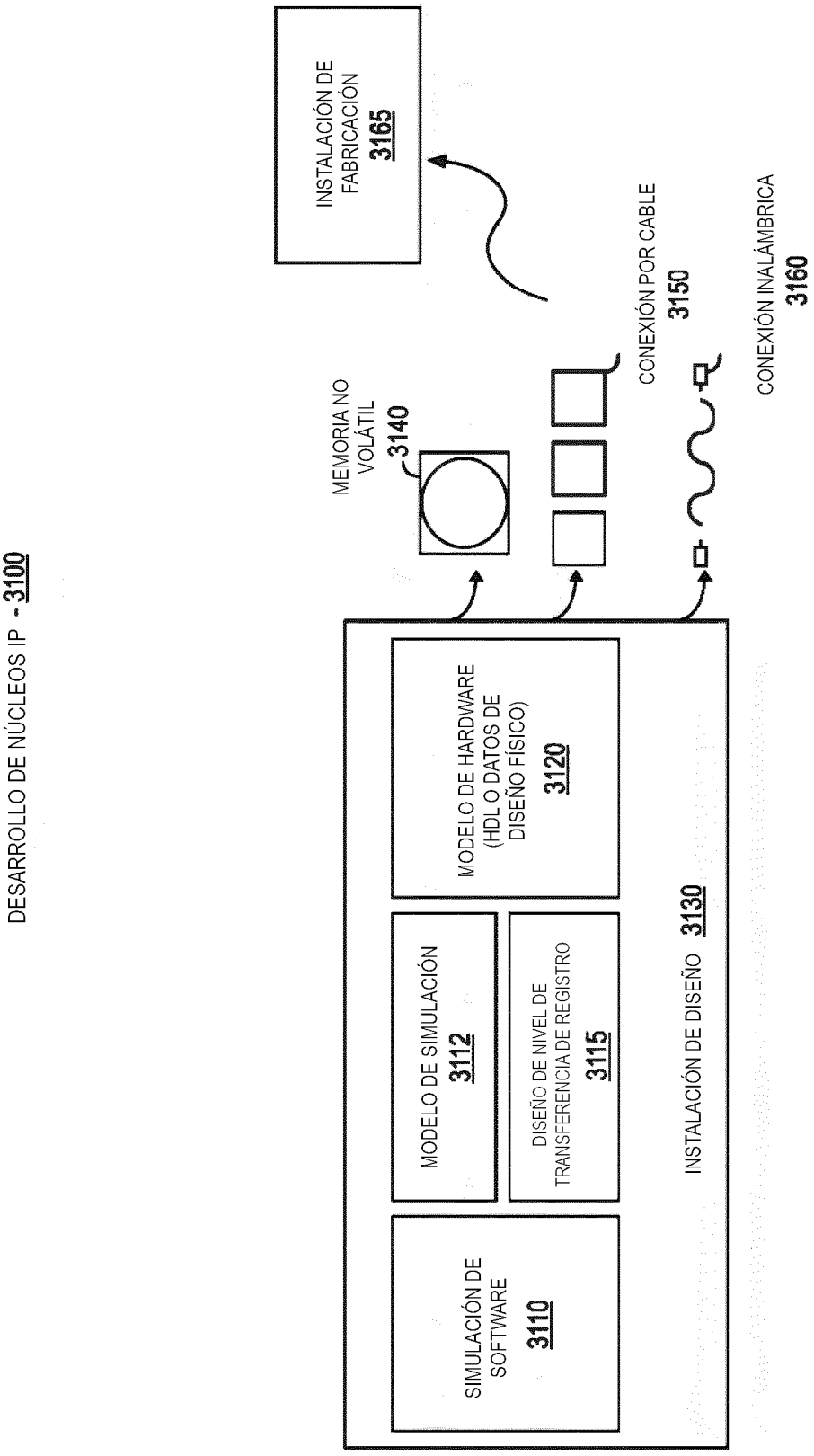

DESARROLLO DE NÚCLEOS IP - 3100
INSTALACIÓN DE FABRICACIÓN 3165
MEMORIA NO VOLÁTIL 3140
CONEXIÓN POR CABLE 3150
CONEXIÓN INALÁMBRICA 3160
SIMULACIÓN DE SOFTWARE 3110
MODELO DE SIMULACIÓN 3112
DISEÑO DE NIVEL DE TRANSFERENCIA DE REGISTRO 3115
MODELO DE HARDWARE (HDL O DATOS DE DISEÑO FÍSICO) 3120
INSTALACIÓN DE DISEÑO 3130

FIG. 31

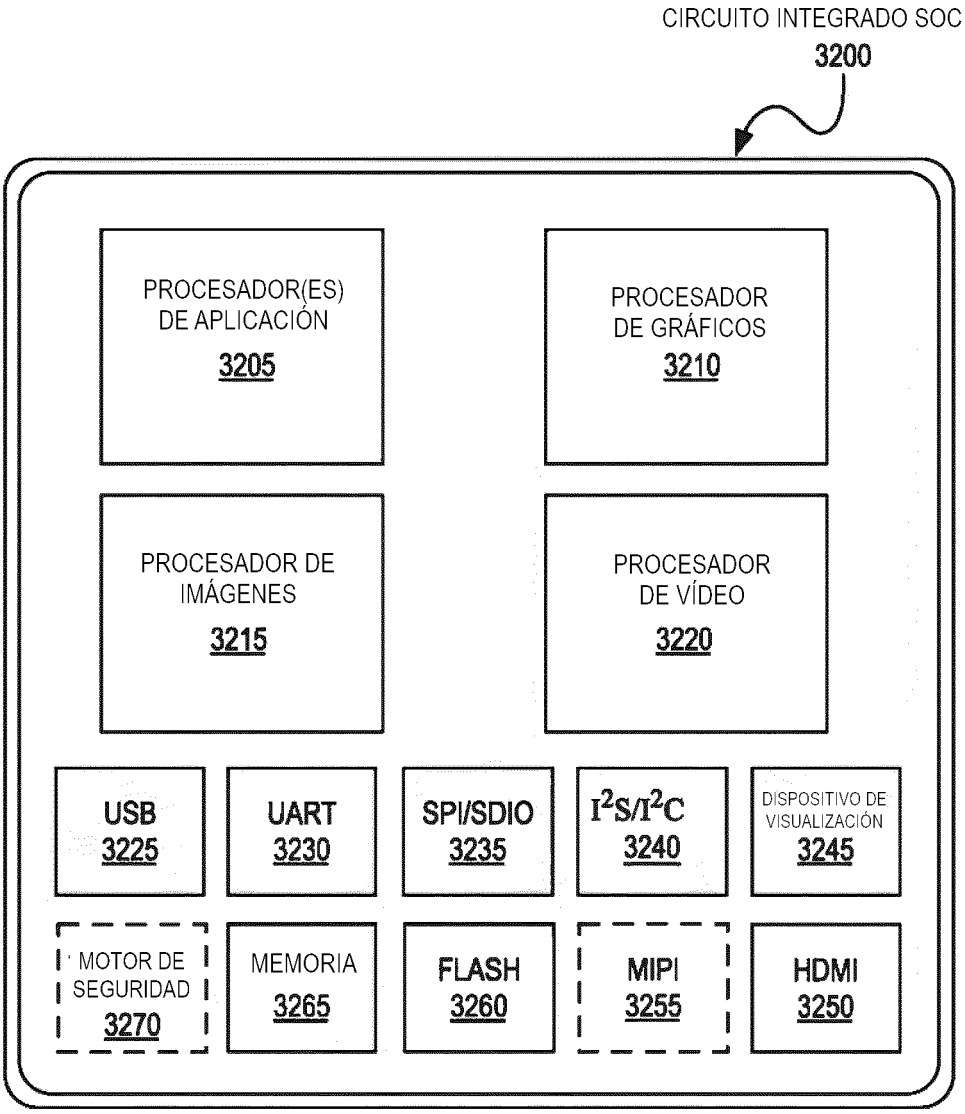

CIRCUITO INTEGRADO SOC
3200
PROCESADOR(ES) DE APLICACIÓN
3205
PROCESADOR DE GRÁFICOS
3210
PROCESADOR DE IMÁGENES
3215
PROCESADOR DE VÍDEO
3220
USB
3225
UART
3230
SPI/SDIO
3235
I2S/I2C
3240
DISPOSITIVO DE VISUALIZACIÓN
3245
MOTOR DE SEGURIDAD
3270
MEMORIA
3265
FLASH
3260
MIPI
3255
HDMI
3250

FIG. 32

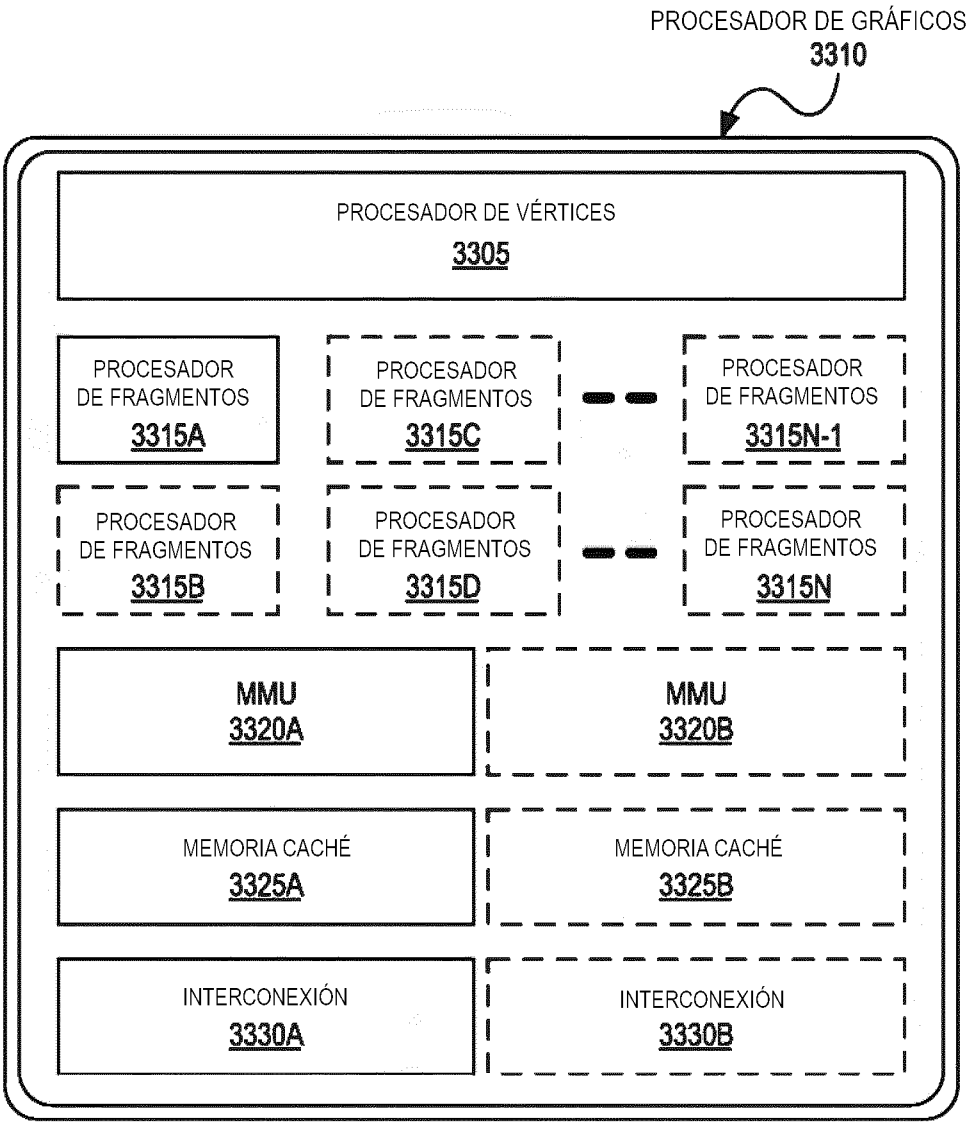

PROCESADOR DE GRÁFICOS
3310
PROCESADOR DE VÉRTICES
3305
PROCESADOR DE FRAGMENTOS
3315A
PROCESADOR DE FRAGMENTOS
3315C
PROCESADOR DE FRAGMENTOS
3315N-1
PROCESADOR DE FRAGMENTOS
3315B
PROCESADOR DE FRAGMENTOS
3315D
PROCESADOR DE FRAGMENTOS
3315N
MMU
3320A
MMU
3320B
MEMORIA CACHÉ
3325A
MEMORIA CACHÉ
3325B
INTERCONEXIÓN
3330A
INTERCONEXIÓN
3330B

FIG. 33

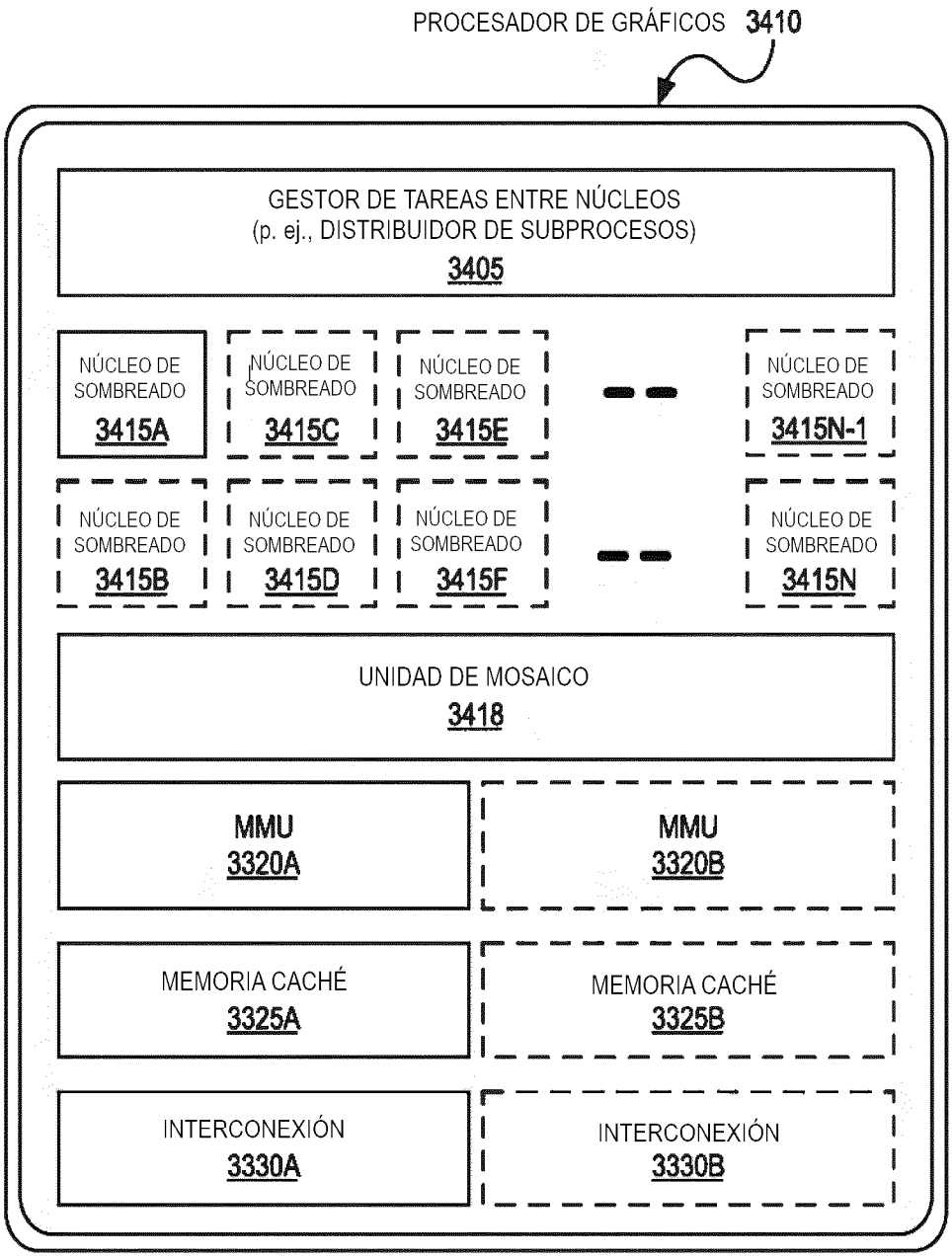

PROCESADOR DE GRÁFICOS 3410
GESTOR DE TAREAS ENTRE NÚCLEOS
(p. ej., DISTRIBUIDOR DE SUBPROCESOS)
3405
NÚCLEO DE SOMBREADO 3415A
NÚCLEO DE SOMBREADO 3415C
NÚCLEO DE SOMBREADO 3415E
NÚCLEO DE SOMBREADO 3415N-1
NÚCLEO DE SOMBREADO 3415B
NÚCLEO DE SOMBREADO 3415D
NÚCLEO DE SOMBREADO 3415F
NÚCLEO DE SOMBREADO 3415N
UNIDAD DE MOSAICO
3418
MMU 3320A
MMU 3320B
MEMORIA CACHÉ 3325A
MEMORIA CACHÉ 3325B
INTERCONEXIÓN 3330A
INTERCONEXIÓN 3330B

FIG. 34